

# **Shareware MAPIT™**

The Professional World Mapping System with  
Something for Everyone.

Version 2.0

Copyright ©1992, 1994, 1997 John B. Allison.

Allison Software  
166 Shady Lane  
Apollo, PA 15613 USA

e-mail: [jallison@kiski.net](mailto:jallison@kiski.net)

homepage: <http://home.kiski.net/~jallison/mapit.htm>

This version of the manual has been stripped of  
most graphics to reduce its size.

## **Copyrights and Trademarks**

---

**MAPIT** is a trademark of Allison Software.

**MAPIT** and its associated programs, data, and documentation are Copyright ©1992 - 1997 John B. Allison and may not be duplicated or distributed without the express written consent of the author.

Microsoft, MS, and MS-DOS are registered trademarks, Windows and Windows 95 are trademarks of Microsoft Corporation.

CompuServe is a registered trademark of CompuServe, Inc.

Hewlett-Packard, HP, and LaserJet are registered trademarks of Hewlett-Packard Company.

Pentium is a registered trademark of Intel Corporation.

---

## **MAPIT OVERVIEW**

Your Graphics Notepad on the World

**MAPIT** is an integrated global mapping and digital display system for work, school, or fun. **MAPIT** provides the breadth and depth of world mapping support necessary to meet a broad range of end-user needs. **MAPIT** — The world mapping system with something for everyone.

### CONTENTS

*MAPIT OVERVIEW* \_\_\_\_\_

**UP AND RUNNING** \_\_\_\_\_

**Packaging:** \_\_\_\_\_

**Installation:** \_\_\_\_\_

**A Quick Demo:** \_\_\_\_\_

**MAPIT and Your Computer:** \_\_\_\_\_

**Making Maps:** \_\_\_\_\_

**Important General Features:** \_\_\_\_\_

**Running MAPIT:** \_\_\_\_\_

**How Do I ...?** \_\_\_\_\_

**MAPIT Gotcha's:** \_\_\_\_\_

**Problems and Solutions:** \_\_\_\_\_

**MAPIT Tips:** \_\_\_\_\_

**MAPIT** \_\_\_\_\_

**The MAPIT Command Line:** \_\_\_\_\_

**Screen Layout:** \_\_\_\_\_

**Menu Commands:** \_\_\_\_\_

**Working with MAPIT Files:** \_\_\_\_\_

**FIGEDIT** \_\_\_\_\_

**Menu Commands:** \_\_\_\_\_

**Tips:** \_\_\_\_\_

**GEOCODE** \_\_\_\_\_

- The GEOCODE Command Line:** \_\_\_\_\_
- Definition File:** \_\_\_\_\_
- Example Output:** \_\_\_\_\_
- Example Definition File:** \_\_\_\_\_
- Advanced Considerations:** \_\_\_\_\_

***GEOTOENT*** \_\_\_\_\_

- The GEOTOENT Command Line:** \_\_\_\_\_
- Definition File:** \_\_\_\_\_
- Example Output:** \_\_\_\_\_
- Example Definition File:** \_\_\_\_\_
- Advanced Considerations:** \_\_\_\_\_

***GEOTOMP1*** \_\_\_\_\_

- The GEOTOMP1 Command Line:** \_\_\_\_\_
- Definition File:** \_\_\_\_\_
- Example Output:** \_\_\_\_\_
- Example Definition File:** \_\_\_\_\_
- Advanced Considerations:** \_\_\_\_\_

***GPS*** \_\_\_\_\_

- GPS Overview:** \_\_\_\_\_
- GPS Installation:** \_\_\_\_\_
- GPS's Menus:** \_\_\_\_\_
- Sample Output:** \_\_\_\_\_
- dB File Format:** \_\_\_\_\_
- Error:** \_\_\_\_\_
- Precision versus Accuracy:** \_\_\_\_\_
- Hardware Connections:** \_\_\_\_\_
- Supported NMEA 0183 Version 2.00 Sentences:** \_\_\_\_\_
- Problems and Solutions:** \_\_\_\_\_

***MP1TOMP3*** \_\_\_\_\_

***NMETOMP1*** \_\_\_\_\_

- The NMETOMP1 Command Line:** \_\_\_\_\_
- Example Input/Output:** \_\_\_\_\_
- Example Data Reduction Session:** \_\_\_\_\_

---

**SIMULATE** \_\_\_\_\_

The SIMULATE Command Line: \_\_\_\_\_

Script File: \_\_\_\_\_

Example Output: \_\_\_\_\_

Example Simulation Script File: \_\_\_\_\_

Demonstration Scripts: \_\_\_\_\_

Advanced Considerations: \_\_\_\_\_

**APPENDIX A – FEATURE/LAYER ASSIGNMENTS** \_\_\_\_\_

By Feature: \_\_\_\_\_

By Layer: \_\_\_\_\_

**APPENDIX C – PEN DEFINITONS** \_\_\_\_\_

**APPENDIX E – STANDARD LAT/LON NOTATION** \_\_\_\_\_

**INDEX** \_\_\_\_\_

---

---

## UP AND RUNNING

### **Packaging:**

Shareware **MAPIT 2** is distributed as four zipped files. You need only the first to test drive this product. The other files demonstrate additional data or ancillary programs.

MAPIT20A (1.4MB) - **MAPIT 2** and data good for zooms 1 to 2.

MAPIT20B (1.2MB) - other exes, examples, documentation.

MAPIT20C (0.8MB) - plate tectonics data, zooms 1 - 6.

MAPIT20D (1.6MB) - general data for zooms 2 - 6.

This data, when fully expanded, requires up to 25 MB of disk space.

### **Installation:**

To install and run **MAPIT** properly, you must preserve the file structure saved in the zip. To do this, unzip all the **MAPIT** zips using the "-d" option.

```
C:\> pkunzip MAPIT20x -d
```

Doing this to all four zips will give you the following directory structure:

```
MAPIT      - executables and data
DBFS       - point data .dbf's
DBQS       - queries and supporting .dbf's
EXAMPLES   - supporting data for examples
```

### **Punt?**

Unfortunately to get to the point where you can read this text, you may have already unzipped **MAPIT** without the -d option. Copy the four original .ZIP files to the top of your drive

```
C:> move *.zip \
(If your version of DOS doesn't support move, use copy.)
```

## 2 Up and Running

---

delete or otherwise clean up the unzipped files, then go back to the instructions above and unzip all four (or as many as you have) using the -d option. The **MAPIT** and all other directories will be created for you automatically.

Change to the **MAPIT** directory

```
C:> cd mapit
```

If you've downloaded the C and D zips, you'll need to combine the .mp3 data files together using the combine batch file.

```
C:\MAPIT> combine
```

Run **MAPIT** by entering "mapit":

```
C:\MAPIT> mapit
```

**MAPIT** will open the DEMO.MP3 file and use it as data. Furthermore, **MAPIT** will open EXTENDED.MP3 to display the Columbus Demo. When you are tired of it, delete EXTENDED.MP3. Like a dismembered star fish, it will be regenerated as a null length file the next time you run **MAPIT**.

### A Quick Demo:

**MAPIT** comes up in the Standard World View the first time you run. <Tab> to zoom in on the Gulf of Mexico. (You are tabbing backward through a two-level history.) <Tab> again to zoom in on Florida. Click to check "DISPLAY/All Features" and click "ZOOM/To Std World" to return to the beginning but with maximum features displayed. Again <Tab> twice. After you've cycled through this simple demonstration, you may want to reduce the number of Features DISPLAYed and begin experimenting by "ZOOMing/In by Corners" and using the <Esc>, <Pg Up/Dn>, and arrow keys as outlined in MAPIT.DOC.

To look up locations in MAPIT's database, select TOOLS/Database /Lookup/City Gazetteer, enter "Saint" in City, click on Lookup/More twice, select Saint Petersburg, FL from the list of matches, and click on Return & Display. MAPIT will redisplay centered on that Laotian. <Esc> the redisplay and ZOOM/To Factor x entering 300.

---

## MAPIT and Your Computer:

**MAPIT** requires a properly tuned, high-performance computer. A Pentium 166 is great. A 486/66 DX2 is marginal. The 286 instruction set is required. Although technically not required, math coprocessor capabilities are essential because of **MAPIT**'s compute-intensive nature.

You may want to add the following line to your AUTOEXEC.BAT file so that **MAPIT** knows which time zone you are in.

**SET TZ=EST5EDT**

This notifies DOS that you are in the Eastern Standard Time zone 5 hours offset from GMT and that Eastern Daylight Time is observed. Without this environmental variable, DOS assumes it is back in Redmond, Washington in the Pacific Time Zone, center of the computing world. See Appendix D for further discussion.

## *Multi-Tasking*

Today's powerful computers and advanced operating systems allow users to work among several concurrently running programs. Working simultaneously with **MAPIT** and a paint or word processing program is possible even when running **MAPIT** from within a Windows DOS session. Switch between **MAPIT** and your other program(s) by using the ALT-TAB task switching sequence. (Hold the ALT key down while clicking the TAB key until the desired program's banner comes up.) You needn't quit **MAPIT** each time you want to switch to the other program.

## *Windows 95*

**MAPIT** works well with Windows 95 and its newer, faster CD-ROM drivers and bigger buffers. Because Windows 95 automatically installs a mouse driver for normal DOS sessions, you shouldn't have to be concerned with that issue.

Adjust the (MS-DOS Prompt) Properties Box of the DOS session controlling **MAPIT** to enable Conventional and XMS memory to Auto. You may also want to set the working directory to C:\ or to your hard drive's default MAPIT directory.



## **Windows 3.1**

### **Tuning Your System**

MemMaker in DOS 6.0 or 6.2 generally does a good job of optimizing memory. **MAPIT** requires 475KB of memory. More is better. To handle its large data requirements, **MAPIT** may make use of XMS, EMS, or disk memory (in that order) for temporary virtual storage. If your system supports XMS (normally by loading HIMEM.SYS in your CONFIG.SYS), **MAPIT** does **not** require you to load EMS via EMM386.EXE.

For a significant performance boost, use disk cache software such as SMARTDrive. The disk cache must be of adequate size, at least 500K, if it is to be of any use. If you notice a lot of disk activity when *redisplaying* a portion of the world, this is an indication that your system will perform better with a larger disk cache.

Be certain that your DOS PROMPT's PIF is set for High Graphics, Full Screen, Background, and that Advanced Options include Uses High Memory Area, High Graphics, Emulate Text Mode, and Retain Video Memory.

**MAPIT** is network ready, meaning **MAPIT** opens its main database read-only. If you are installing **MAPIT** in a multi-computer networked environment and are licensed to run multiple **MAPIT** copies, consider placing the main database WORLD.MP3 on a single node on which SHARE.EXE is run. The space savings can be significant.

### **Installing a Mouse (3.1 only)**

**MAPIT requires a mouse.** Unless you have a mouse physically attached to your computer **and** have installed its driver, you will be unable to operate **MAPIT**. You can exit **MAPIT** from the keyboard by entering <CTRL>Q.

Install the mouse driver by issuing the following command at the command prompt:

```
> path\MOUSE           (e.g. C:\WINDOWS\MOUSE)
```

where *path* is the path to the directory containing the mouse driver.

Place this command in your AUTOEXEC.BAT file so that MOUSE is always installed when your system boots up.

Caution: Windows 3.x is smart and will show a mouse cursor without the mouse driver's being installed. If the cursor doesn't appear when you run **MAPIT**, <CTRL>Q out and try to edit a file with the standard DOS editor from the DOS prompt (e.g. EDIT AUTOEXEC.BAT). If you don't have a mouse cursor in EDIT, the driver isn't loaded. Try loading it from the DOS prompt as above, and if that works, add the same command near the end of your AUTOEXEC.BAT but before any call to WINdows.

## **Making Maps:**

Although **MAPIT** doesn't directly support a large variety of output devices, you can produce simple as well as publication-quality mapping output.

### ***Screen Capture***

Windows enables screen capture to the clip board via the Print Screen key. Paste (<Ctrl>V) the image into Paintbrush or Paint, standard Windows 3 and 95 applications. Change the black background to white (and other colors to their compliments) with Paintbrush's PICK/Inverse of Paint's Image/Invert Colors option. After prettying up the image in your graphics program, print it or insert it into your word processing document.

### ***.PCX Files***

A more controlled way of capturing screen images from **MAPIT** is to select the TOOLS/.PCX menu items and save the images to .PCX files. **MAPIT's** .PCX command allows you to invert the black background to white first, if you so desire. You can save the entire screen including the menu heading and footer, just the mapping area, or a selected rectangle from within the mapping area to a file whose name you choose and later import these files into your graphics or word processor by selecting the PCX filter associated with the File/Open command.

### ***HP-GL/2 Output***

Hewlett Packard's Graphics Language/2 provides the most detailed output possible. Unlike the .PCX format which is limited to the resolution of your screen, HP-GL/2 is a vector format output limited only by the resolution of **MAPIT's** underlying database and the resolution of your laser printer or pen plotting device. **MAPIT** directs this output to either your printer/plotter attached to the

PRN port or to a .HGL file. If, for instance, you want to import HP-GL/2 input into a Microsoft Word for Windows document, first produce the file from within **MAPIT**, then, from within Word, select Insert/Picture setting the List Files of Type: option to HP Graphics Language (\*.hgl).

## Important General Features:

### ***Mousing***

You can use the mouse in two different ways:

- **Dragging** — Some selections (for example, Zoom by Corners) require pressing the left button to establish the starting corner or position and, ***with the button still depressed***, dragging the cursor diagonally to the opposite corner or position to be measured ***before releasing the button***. If, when a drag is required, you press and release at the same point, the command is aborted; nothing happens.
- **Point and click** — Other activities, notably those requiring one-at-a-time point selections, look for the depression and release of the left mouse button without any intervening movement. Sometimes use of the right mouse button is required to signal the end of multiple-point entry. Sometimes the right mouse button aborts the command.

### ***Arrow Keys***

The keypad keys support simple navigation. **Pg Dn** zooms you in by a factor of 2, **Pg Up** zooms out by 2. The **arrow keys** move you over the map in the direction they point by half a screen. These keys interrupt the repaint process so that if you wish to go right a whole screen and up one half, quickly pressing the right arrow twice and the up arrow once immediately takes you there. Pressing the **Pg Dn** key three times zooms you in 8 fold.

### ***Bottom Status Line***

From left to right the bottom status line displays the current zoom factor, scale, map width at center screen, and latitude and longitude of the cursor.

---

## ***Header Status Field***

The Header Status Field three different fields of information depending upon **MAPIT**'s state. The most common display is the date/time/hour-offset. This shows the approximate (calculated) day and time and the number of hours the time of the location pointed to by the mouse is offset from your local time. See Appendix D for further discussion.

As **MAPIT** redisplay the screen, the databases **MAPIT** is reading flash up in the status field. This gives you a feel for what **MAPIT** is currently doing and where it is spending its time.

When you have the GPS Markers activated and there is active data present, **MAPIT** displays the current date/time passed through the GPS interface DBF. If you are running a simulation, you can see exactly where you are in the simulation: how it has progressed.

## ***Esc***

Pressing the **Esc** key will close a menu and will terminate the current scan of a data base if you are repainting the screen. Since **MAPIT** scans several databases, you may have to press **Esc** several times or hold it down for auto repeat to abort the entire repaint process.

## ***Tab***

Pressing the tab key toggles the display between the current and the last location/zoom (except those positions defined by Zoom/To Standard World).

When running from the CD-ROM, **MAPIT** comes up in the standard world view. If you tab once, you will go to an overview of the United States. Tab again and you will go to an overview of Europe. These two views are also stored as positions A and E respectively. (i.e., typing 'E' will take you to Europe.)

## ***Single letter or digit***

Pressing any single letter or digit restores the display to the location and zoom stored at that letter or digit. (Short for POSITION/Restore.)

## ***Left mouse button — cities, hidden text, figures***

Clicking with the left mouse button at a non-menu area of the screen, causes **MAPIT** to scan visible databases looking for hidden text, figure, and city objects to display, a potentially time-consuming process. **Esc** aborts this scan.


## ***Important Menu Functions***

The two pulldown menus new users will find most useful are the ZOOM and the DISPLAY menus. ZOOM implements navigation and repaint. DISPLAY effects the visibility of layers (features). A third menu, TOOLS, contains Database Lookup for positioning by the City Gazetteer.

## ***Enter — The Position Menu***

```
Loc: 40.2276791 N 80.2816901 W
Zoom: 1
Scale: 104,559,202
```

```
Okay
```

```
Display new loc.      
```

```
Back to MOUSE
```

```
Abort
```

**MAPIT** supports keyboard entry of exact coordinates although the operational context begins as mouse entry. Pressing the **Enter** key when **MAPIT** is prompting for mouse input with a cross hairs brings up the floating Position Menu for exact

keyboard entry of a location. Clicking on Loc: allows you to enter a latitude and/or a longitude in free format. (See Appendix E.) You can also change the zoom or scale, display the new location, and go back to mouse entry for visual positioning and a possible retry with the Position Menu for fine tuning.

The Position Menu is a very powerful navigational feature. Suppose you wanted to lay in a permanent great circle between Heathrow in London and JFK in New York. You'd use the TOOLS/Great Circles/Point to Point command. But because the accuracy by mouse entry is not sufficient at the zoom 4 level required to see both England and New York simultaneously, zoom in on the Heathrow end first, click that location with the mouse, and when the second mouse marker displays, press the **Enter** key. The Position Menu will pop up. Change the zoom level to 2 and click on Display new loc. **MAPIT** will display centered on London with New York in view. You are still in insert mode in the

Point to Point command. Position the mouse marker over western Long Island and press **Enter**. (Don't click the mouse yet!) Change the zoom to 300 and Display new loc again. Find JFK, move the mouse marker to it, and click the left mouse button. The result is an accurately-positioned great circle between Heathrow and JFK

The few places you can't enter exact data from keyboard are those associated with dynamic screen repaint while you move the mouse: dynamic great circle generation and poly-line entry, for instance.

### **MAPIT Utilities**

<b>MAPIT</b>	<b>The primary program.</b>
DBFSORT	Create an index for a new style <b>MAPIT</b> DBF.
DBFTOMP3	Convert DBF to old style .MP3 point data.
<b>FIGEDIT</b>	<b>Create and edit MAPIT figure definitions.</b>
GARTODBF	Special Garmin track file to DBF conversion.
<b>GEOCODE</b>	<b>Add lat/lon fields to a DBF based on a matching field.</b>
<b>GEOTOENT</b>	<b>Geocoded DBF to MAPIT entity DBF.</b>
<b>GEOTOMP1</b>	<b>Geocoded DBF to .MP1 line data.</b>
<b>GPS</b>	<b>Windows GPS receiving program.</b>
<b>MP1TOMP3</b>	<b>Convert .MP1 text line data to displayable binary format.</b>
MP3MOD	Modify .MP3 data utility.
MP3TODBF	Convert old style point .MP3 data to DBF.
MP3TOMP1	Convert .MP3 line data to text .MP1 format.
NETANA	Network node connecting analyzer.
NMETOMP1	Permanent track output from GPS to .MP1.
<b>SIMULATE</b>	<b>Accelerated real-time simulation of moving objects.</b>
TILEIZE	Update the tile field in a new style <b>MAPIT</b> DBF.

### **Help for MAPIT DOS Utilities**

All **MAPIT**-based DOS utilities including **MAPIT** itself support the following command line protocols from the C: prompt.

Entering / from the command line generates a complete list of calling options.

Entering /? generates on-line documentation of the program and its command line arguments.

Entering `/help` generates on-line documentation of the statement syntax of the program's definition file. Not all programs have definition files and support the `/help` option.

You can abbreviate command line options short of the point of ambiguity. **MAPIT**'s `/noe` is the same as `/noextended`, but `/no` will induce an error message listing the ambiguous possibilities as `/noextended` and `/nosavefile`

### Running MAPIT:

Shareware **MAPIT** can be accessed with

```
C:> mapit
```

which opens DEMO.MP3 as the main file and EXTENDED.MP3 as the extended or private output file. If the latter is not found, **MAPIT** creates a zero-length EXTENDED.MP3.

```
C:> mapit main /extended=x
```

opens MAIN.MP3 as the main file and X.MP3 as the extended file.

```
C:> mapit nul /noextended
```

starts **MAPIT** with no .MP3 data.

### How Do I ...?

*I want to stick pins into a map. How can **MAPIT** help me?*

You probably want to insert figures (simple line drawings) with labels and up to a screen's worth of information hidden behind each figure although simple stroked text might also do. Your first question is whether you're inserting hundreds or even thousands of entries from an already-existing database (.DBF), or if you're going to do this one-at-a-time from the keyboard.

In either case you're going to store the data in DB point entities, and you'll have to add a record to the scan table (see below). Because the CD-ROM is read-only, you must use a scan table residing on your hard drive which implies you will want to use the `mycopy/mymap` technique for starting **MAPIT** from your hard drive described on the first page of this section and change the scan table reference as outlined below.

If you're converting a lot of data from a dBASE file: You must geocode the file (see below) and convert it to a figure or other DB

entity using GEOTOENT. Finally you must reference your newly-created figure file by adding a record naming it to the currently-active scan table (see below) and make that record active. Subsequent redispays will display your new data.

If you're adding a limited amount of data from the keyboard and this is the first time you're entering data, you must create a point entity database. Use FILES/New DBF/Figure and enter a name such as "C:REDPINS" followed by a return. The General DBF Editor will display the first record which is preloaded with a square located lat = 0, lon = 0 near the coast of Africa! Escape out of this record and ignore it or move it later. Add your entities with the EDIT/DBF Entites/Insert menu. At the prompt for a DB point entity file, enter the name of the file you just created: C:REDPINS. You'll be given a choice of figure names (which you can augment with FIGEDIT). Choosing one (such as circle) brings up the plus positioning cursor which you can exactly position by entering a return or with the mouse. Locating your figure leads you to our old friend the General DBF Editor. Here you can enter Label\_Text (always end text with an Esc) and Hidden\_Text. If you have more than 40 characters of hidden text, use the Memo field. You can change many other options here, but don't alter E\_TYPE, LATITUDE, LONGITUDE, or TILE. Use the EDIT/DBF Entites/Move option to change entity location. Esc completes the entry. The marker will display on the screen but will not redisplay unless you reference its DBF (C:REDPINS) from your scan table (see below).

*What is the scan table, and how do I use and modify it?*

If you are going to display your own dBASE entities, you must reference the files in which they reside from another DBF called a scan table. Because the CD-ROM is read-only, you must add your custom records to a scan table residing on your hard drive which implies you will want to use the mycopy/mymap technique for starting **MAPIT** from your hard drive described on the first page of this section.

From FILES/Edit DBF, enter C:SCANTABL (no final "E") followed by Enter. The first record of C:SCANTABL.DBF will display. Use <Ctrl>End to go to the last record. If it is empty or unused, add the name of your DBF (e.g. C:REDPINS), add the E\_TYPE ("F" for Figures, see list under Add Scan Table Records in the **MAPIT** section) Esc-ing to terminate text, and mark ACTIVE "T" rue. Exit the editor with Esc. Use FILES/Add Scan Table Record to append a new record to the end of the DBF if there is no spare.



The scan table you have modified must be activated from within **MAPIT** by selecting FILES/Open and entering C:SCANTABL. Clicking on FILES/Scan Table DBF tells you which scan table, if any, is currently active.

*How do I use **MAPIT** to track hurricanes?*

Answer A: Let's assume that you're using labeled circle figures which you created in accordance with the instructions in question 1, "How do I stick pins into a map." As the hurricane proceeds along its track, you either want to move the figure or duplicate it changing its location, size, and possibly updating its label as it progresses. Use EDIT/DBF Entities/Move to move an existing figure. Use Duplicate and Change to make additional copies and edit them. All pretty nifty.

Answer B: You want to see the dynamic of the interaction of several tropical storms or you want to see how your hurricane moves in time. Use SIMULATE to show all this in accelerated real time. You'll have to use **MAPIT**'s moving markers with or without trails to show the progress of your storms, not quite as sexy as the circle figures, but still quite dramatic and revealing. Perhaps you can combine both duplicated figures and moving markers. You'll have to lay out a .SIM script of the hurricane's path as explained in SIMULATE.

*How do I GEOCODE data?*

Geocoding is the process of assigning latitude/longitude fields to dBASE records. **MAPIT** DB entity files are, by definition, geocoded, but not all geocoded files are DB entity files. DB entity files contain many required fields which are not necessary for geocoding. To geocode a file, you must relate a field or fields in your database with corresponding fields in a **MAPIT** database. Quite often you'll use a ZIP code or city/state/country fields for key and matching fields in geocoding. See \MAPIT20\DBFS on the CD-ROM for **MAPIT** databases against which to geocode. Use the **MAPIT** GEOCODE utility to do the work.

*How do I create a **MAPIT** DB entity file from my dBASE IV file?*

Geocoding is the first step in creating a DB entity file from your own data. Each record must have an associated latitude/longitude. That's the hard part. Use the **MAPIT** utility GEOTOENT to add the fields necessary to convert your geocoded file to a particular **MAPIT** entity format.

*How do I visualize which colors go with which color numbers?*

For those of us who can't quite picture magenta, mauve, or color number 11, use the EDIT/Layers/Change Color/Color (or Layer) to display a new color or the color on another layer. Since the Color option actually changes the color on that layer, it's safest to change layers remembering that layer 121 uses, by default, color 1, layer 122, color 2, etc.

*How can my child make maps for school?*

**MAPIT** is the perfect source of real data for students to use in producing maps for school or just for fun. Probably the easiest method is to create .PCX files from within **MAPIT** and to import them into Paint of Paintbrush. If you need to produce more colorful output, use the flood fill option of Paint to color oceans and lakes blue. If you don't have a color printer, the lighter colors will give you more satisfactory contrasting colors. Light yellow, for instance, serves well as water contrasting against land. Choose the roller symbol for flood fill. Non-solid colors are produced by dithering and won't unflood, so check your flood areas first for leaks with a solid color. (Leaks are caused by breaks in borders or lines which allows the flood color to spread beyond its intended boundary. Use Zoom to find and patch leaks.) Children love this. Turn off the **MAPIT** grid before screen capture to make fills of large areas easier.

*What do all those files hold that seem to come along with **MAPIT**? Which may I delete?*

.mp3	— Main and private files holding line and old style point entities. WORLD.MP3 is the main data file.
.fig	— File containing the user's custom figures. Normally this file is user-created by FIGEDIT. Without it, figures won't be displayed. Defaults to STD.FIG.
.fnt	— File defining stroked fonts. Without it, stroked text doesn't work. Defaults to STD.FNT.
.pen	— File overriding <b>MAPIT</b> 's internal HP-GL/2 plotter settings. See STD.PEN.
.sav	— File holding <b>MAPIT</b> 's final state to

restart **MAPIT** where it left off.  
Defaults to MAPIT.SA. **Recreated if deleted.**

.shw

- File holding a shadow image of similarly-named large .mp3 file. **Recreated if deleted.**

*What are the old versus the new style point entities. Which should I use?*

**MAPIT** point entities come in two styles: old, which reside in .mp3 files along with line data, and new, which are specialized dBASE IV files. MP3TODBF and DBFTOMP3 convert entities from one format to the other. In general the new style point entities are more robust and flexible than the old. It is, however, easier to save data *ad hoc* to your private .mp3 if you are running “map myfile” or “mymap myfile” and do not have a DB entity file created and entered in your scan table. On the other hand, you should set up a DB entity file in the new style if you are serious about manipulating your data. See the first question, “How can I stick pins into a map” for details.

*How do I export line data for use in another system?*

If the data to be exported is from **MAPIT**'s main database (D:\MAPIT20\WORLD.MP3) or from your own private .mp3 file, open that file as the main database with a new extended database (e.g. map new). Use the EDIT/Copy command to copy all visible entities to your extended database. (To capture the fullest depth of **MAPIT** data, you must be at zoom 24 or beyond.) Outside of **MAPIT**, use MP3TOMP1 to copy and convert the data in your extended database from .mp3 to .mp1 ASCII lat/lon format for import into another system. Also see MP3MOD for copying .mp3 data within well-defined extents from the C:> prompt rather than from with **MAPIT**.

*In DOS, there's this confusing distinction between file names C:\xxx and C:\xxx. What does that leading slash do?*

The leading slash says, “Go to the top of the drive.” If your current directory on drive C: is \mydir (you changed directory there with the command “cd \mydir”) and you ask for a directory listing of c:\xxx, DOS will look for the file xxx on drive C: relative to C:'s current directory and will return, if it exists, C:\mydir\xxx. If you ask for c:\xxx, DOS will try to return C:\xxx as found in the root directory. The slash implies an absolute path while no slash

implies a path relative to that drives current directory. **MAPIT** quite often uses relative addressing so that it doesn't have to know the details of the directory structure you choose to implement on your hard drives.

This leads to a pet peeve. The DOS Prompt under Windows is often set up to begin its life in C:\windows where enough garbage and general confusion reside to confuse the most hearty soul. In your W3.1 PIF editor or by right clicking on your W95's MS-DOS Prompt's tool bar icon and left clicking on Properties, change the Working Directory to a simple "C:\\" to take you to the root of your C: drive at program invocation. Of course if you have a DOS Prompt dedicated to a particular task (such as **MAPIT**), you'll want your Working Directory to reflect that choice (e.g. D:\ for **MAPIT** on your CD-ROM or C:\xxx for **MAPIT** from your hard drive).

*I want to make a really large map for a display at school, and all we have is a little 8 ½ by 11 printer.*

Get out the scissors and tape and set up **MAPIT** at the zoom level you wish to print your map. You're going to make a series of maps covering the entire area in which you're interested and, with your scissors and tape, cut and paste them together. Because **MAPIT** is a Mercator projection, you can piece these separately printed maps together indefinitely, at least in theory.

The hard part is producing map sections which fill up a whole sheet of paper out of your printer. Its easiest if you have a printer which can accept HP-GL/2 directly or a drawing program which accepts it as input and supports your laser printer. Working with Paint in Windows 95 is a little more difficult but workable once you have it figured out. You'll have to export inverted .PCX files of the entire map area or, from within **MAPIT**, use Print Screen to cut and a <Ctrl>v to paste directly into Print. You will have had to expand the background to approximately twice the size and then stretched the picture 200% in both the horizontal and vertical directions before printing in landscape mode. Perhaps you have an easier method or a better software package for outputting a full-page image.

The key and easiest part is producing the adjacent images from within **MAPIT**. After selecting your zoom and outputting the first image, hit an arrow key twice to move either up, down, or sideways before outputting your next image. Keep doing this until you've covered your area of interest while task switching with a

<Ctrl>Tab between **MAPIT** and your drawing program for each new image. At a scale of 1:1,000,000, a paper map of the equator would stretch some 125 feet. How many sheets of paper in 125 feet?

*I heard about Baku, Azerbaijan on the radio but am a terrible speller. How do I look it up?*

Normally you'd use the TOOLS/Database Lookup/City Gazetteer query to look up a city whose name you could spell. With so many cities whose names begin "Ba", you'd be better off using the /Cities in.... query and entering "Az" for the country and "Ba" for the city. The seven entries immediately appear.

*My granddad served on Los Negros in the Pacific during WWII. It must be pretty small. I've never found it on a map. How do I look it up?*

Use the TOOLS/Database Lookup/Placenames query. You'll find when you turn on DISPLAY/Boundaries/Political Text that Los Negros is a little horse shoe-shaped island with a total length of about 15 miles, part of Manus, Papua New Guinea in the Admiralty Islands, the Bismarck Archipelago.

*How do I make scatter diagrams of cities over large continental areas?*

To accelerate display, **MAPIT** separates cities by size in a series of dBASE files accessed via the scan table and ignores them until you zoom in to some maximum scale. Use FILES/Edit DBF to open your hard disk version of SCANTABL.DBF, look for the files holding cities of the sizes you're interested in, and change their MAX\_SCALE fields from the 4,000,000 or whatever to 0 or some other small number. In DISPLAY/Cities & Landmarks, toggle "Symbol & Name" to "Symbol only". The next time you redisplay the screen, **MAPIT** will display all your cities as circles. If you are zoomed out for a broad view, this redisplay will take a long time.

*I'm plotting a map, and city names overlie each other. What can I do?*

There is an obscure field in the city record named TXT\_OFFSET which defaults to "E". Enter the first letter of **A**bove, **E**ven, or **B**elow to change the position of the name text to lie to the right of and above, even, or below the level of the city's marker. Edit individual cities using EDIT/DBF Entities/Change and clicking in the middle of their symbols. Here again you cannot change data

on the CD-ROM but must make, change, and reference a copy on your hard drive.

## MAPIT Gotcha's:

Don't forget to TILEIZE, either from within **MAPIT** for small dBASE files or from the utility for large files, after you've created or made manual changes to the lat/lon fields. Without the proper tile number associated with a latitude/longitude pair, the entity simply will not display.

Don't forget to delete the scan table .MDX index file after you've made changes to the sort PRIORITY field. **MAPIT** will re-create a correct index on startup.

If **MAPIT** has problems opening a .DBF after you've created it and **MAPIT** or a utility died in the process and you know that the file is there and is of the correct format, rebooting may clear the problem. Access problems can haunt a DBF like the ceaselessly roaming spirits of the undead after the untimely demise of a dBASE program.

Blow away the .SHW shadow file when you've made changes to (lengthened or changed header zoom via MP3MOD) .MP3 files. The old shadow file will prevent the changes from becoming apparent. **MAPIT** automatically creates a new shadow file when it doesn't find the old one for large .MP3 files.

DB entities can be inserted into inactive databases (or into databases whose scan table is inactive). The new entity will display upon insertion but will not display on repaint.

There is a subtle distinction between a null entry in the primary key field of a database lookup and a blank. A null in the City field of City Gazetteer will find all cities starting with the first record. A blank will find none. Both blank and null match all entries on secondary keys.

## Problems and Solutions:

The following are problems you may encounter while using **MAPIT**, and their possible causes/solutions:

1. *No stroked text appears. The latitude/longitude numbers are missing.*

## 18 Up and Running

---

- A. The font file `std.fnt` which holds stroked text character definitions is not in the user's directory.
  - B. Stroked text which is too small to read at a given zoom is not displayed.
  - C. The current zoom factor does not fall within the text's min/max zoom values.
2. *The city markers display, but no city names.*
- A. Under the menu DISPLAY/Cities & Landmarks, the Symbol & Name option must be selected.
  - B. The absence of a font file will also cause the names to be absent.
3. *Figures I have previously entered are not visible.*
- A. The figures file `std.fig`, which holds figures' definitions, is not in the user's directory.
  - B. The current zoom factor does not fall within the figure's min/max zoom values.
4. *Graphic elements I have added in a previous session have disappeared.*

The graphic elements you entered are written to the private database `EXTENDED.MP3`. That same file must be in your current directory. Use the *DIR* command to check `EXTENDED.MP3`'s length and date for reasonableness.

5. *Error — Lost sync in mid file: "file\_name"*

The `.MP3` data file *file\_name* has become corrupted. Nothing beyond the point of corruption can be read. Replace it with a backup file or try to EDIT/Copy as much data from it as possible. (See Private Database Management.)

6. *Data I try to highlight for copy, modification, or deletion won't highlight although it displays. This is so frustrating!*

Data you are going to change must be in the extended database. Although you can view data in **MAPIT**'s main database, it is read only and cannot be changed. (See **MAPIT**'s /delete switch for the one exception.)

7. *Line data I entered while running **MAPIT** from the CD with "map", not specifying any extended file, appears to be entered but disappears upon redisplay.*

You have discovered the beauty of the /noextended option, the nul file, the bit bucket, write only media, if you will. **MAPIT** constructs temporary graphics displaying your lines or old-style point entities as you enter them and as it write the permanent data to the nul file. Upon repaint, the nul file, which always returns an immediate End-of-File condition, a nul length, tells the redisplay code there is nothing there to display. Beautiful, isn't it?

## **MAPIT Tips:**

**MAPIT** is busy scanning the databases when the cursor disappears between repaints. When it becomes visible again, the redisplay is finished and **MAPIT** is ready to accept command input. Remember, you can cut short each database scan by pressing the Esc key — once for the main database and again, if necessary, for your private database. (Usually the main database scan is much longer.)

Use the cross hairs displayed by POSITION/Center for more accurate lat/long display in the status line while measuring positions of objects. Cancel when finished with the right mouse button.

Use DOS's NUL file name if you need an empty file to read from or a bit bucket to write to. For example, if you wish to view or operate on the private database file EXTENDED.MP3 file by itself without possible interference from the data in the main database WORLD.MP3, open **MAPIT** using the NUL file as the main database name:

```
C:> mapit nul
```



## MAPIT

### The MAPIT Command Line:

In general, start **MAPIT** from the command line by entering

```
MAPIT [main_data_file] [/options]
```

where

*main\_data\_file* — The primary .mp3 format data file opened read-only. Defaults to WORLD.MP3 if not specified.

options

*/extended=private\_data\_file* — A private .mp3 format data file opened for update with user data or changes. If none exists, a zero length file is created. Defaults to EXTENDED.MP3 if not specified.

*/noextended* — No extended file looked for or created.

*/figure=figure\_file* — File containing the user's custom figures.. Normally this file is user-created by FIGEDIT. Without it, figures won't be displayed. Defaults to STD.FIG if not specified.

*/font=font\_file* — File defining stroked fonts. Without it, stroked text doesn't work. Defaults to STD.FNT if not specified.

*/save=save\_file* — The file holding **MAPIT**'s final state to restart **MAPIT** where it left off. Defaults to MAPIT.SAV if not specified. Recreated if deleted.

*/nosavefile* — No .SAV file produced (read only environment).

*/pendefinition=filename\_name* — Override **MAPIT**'s internal HP-GL/2 plotter settings. Defaults to a file ending in .PEN.

*/delete* — Allows the **MAPIT** menu selection EDIT/Copy & Delete to work by opening *main\_mp3\_data* for update.

**MAPIT** also creates and maintains a .SHW shadow file for each large .MP3 database it opens. **MAPIT** will create a new one if you

delete it. If you alter a large .MP3 file by concatenating another to it, delete the corresponding shadow file to make all of the data in the new .MP3 file visible. (See **Working with MAPIT Files.**)

See MAP.BAT and MYMAP.BAT for descriptions of sample batch startup files for **MAPIT**.

## Screen Layout:

**MAPIT** uses a mouse-actuated pull-down menu structure in which commands are grouped by similar function along the top of the display.

The bottom of the screen holds **MAPIT's** status line, which displays the zoom factor, the approximate scale and distance across the center of the screen, and the latitude/longitude of the mouse pointer. Sometimes instructions, prompts, or information are written to the status line. The Header Status Field, top and center, is described above.

FILES	Controls ScanTable access and Exit
ZOOM	Redisplays the screen and controls zooming.
POSITION	Controls the viewer's location.
TOOLS	Controls great circle calculation, database lookup, GPS markers, and plotting functions.
EDIT	Controls layer color assignments and DBF and .MP3 entity modification and duplication.
DISPLAY	Controls mapping features visibility and format.

## Menu Commands:

### **FILES**

Exiting **MAPIT** and advanced operations controlling the selection of databases for display.

### **Scan Table DBF**

dBASE IV databases (DBF's) have been added to supplement the main (WORLD.MP3) and extended (/extended=) .MP3 databases and are designed to hold point type data (figures, stroked text, hidden text, and cities). While .MP3 files still support point data, their primary function is now limited to line data. The scan table is a dBASE file holding a collection of entity file names which, in turn, hold **MAPIT** point entity data for display. Think of a scan table as a directory of entity tables. Only one scan table can be

active at a time, but you may have many unopened and therefore unused scan tables existing in your computer's file system. On the other hand, you need not have any scan table defined. By default, **MAPIT** looks for the scan table *scantabl.dbf* (no "e" in *scantabl*). If not found, **MAPIT** ignores its absence. Scan tables must have one or more entries. The entries refer to **MAPIT** entity tables which describe entities for display on your screen. This directory/file type hierarchy makes for a very open-ended, flexible, and powerful entity display system.

Scan table entries can be either active or inactive. They must be active to be searched and displayed. If active, the named entity table must exist. Its availability can also be limited by display scale, a technique designed to improve redisplay efficiency.

Scan Table DBF displays the name of the current scan table. Click or hit any key to go on.

## **Open**

Open requests the name of a scan table to open as the current scan table. Use the complete path name if it is not in the current directory. The old scan table is automatically closed.

## **Close**

Close closes the current scan table but does not open a replacement. All **MAPIT** redisplay now takes place from the main and extended .mp3 files.

## **New DBF**

New DBF is a special command used to create a new **MAPIT** dBASE file containing a single record. You choose the type of database to create: a scan table or one of several entity table databases. **MAPIT** creates databases of minimal field count and with required field names. Its possible for you to use more complex databases with your own extra fields added. Currently only entity tables containing point entities are supported. Lines must still reside in .mp3 files.

## **Add Scan Table Record**

Use this command to add additional scan table records beyond the first one. You can control the order in which entity tables are searched and displayed by adjusting the value of the PRIORITY field, a numeric two-digit field sorted in ascending order. The default priority is 20.

You must set *ACTIVE True* if you want the referenced entity database to be scanned. *E\_TYPE*, the entity type of the referenced entity tables must be set according to the table below. *E\_TYPE* is case insensitive.

*CATEGORY* defines an *ad hoc* entity type which, in the future, will allow dynamic inclusion/exclusion of entire entity files just as Political Boundaries are a predefined category of .mp3 files. *Not yet implemented.* *F\_COLOR* refers to the a forced color for all entities in this entity file. A value of -1 signifies no forced color. *DESCRIPTION* is descriptive comment for documentation purposes only. *FILE\_NAME* is, of course, the path name of the entity file to be scanned and displayed. *FILE\_TYPE* must be 'E' for **MAPIT**

E TYPE	MAPIT Entity
'C'	city
'F'	figure
'H'	hidden text
'M'	pcx map
'S'	stroked text

entity files. *MAX\_SCALE* refers to the maximum scale above which this entity file is not scanned. *MIN\_SCALE* refers to the minimum scale below which this entity file is not scanned. Both these values are compared with the current display scale. For instance, the Standard

Display centered on the prime meridian at the equator and showing the entire world gives a scale of approximately 1:157,000,000 on a 10 inch wide monitor. Zooming in reduces that scale shown near the lower left corner of your display. You can specify that entity tables will not be used until the current scale is less than or equal to a *MAX\_SCALE* value and won't be available beyond (less than) the value of *MIN\_SCALE*. You can turn this feature off by not changing the default values from 0, a flag to ignore these values.

### **Edit DBF**

Edit DBF is a generalized dBASE editor which should work on just about any reasonable database. It is designed, however, to facilitate changing fields in **MAPIT** scan table and entity records. The fields listed, their order, and data types are dependent on the definition of the database being edited. The example show here is from a scan table database. One should realize, however, that because this is a generalized editor and because no tag (key) files are opened, changes made to a key field will have no immediate effect upon record ordering. Changing the *PRIORITY* field in scan table records or the *TILE* field in entity files will have no immediate ordering effect. You can, however, force a reordering by deleting the corresponding .MDX (index) file and running **MAPIT** again. (*scantabl.mdx* is the index for *scantabl.dbf*.) Finding the index file

missing, **MAPIT** will recreate it on the fly, and everything, including the newly changed record, will be ordered properly.

You can change a field by clicking it with you left mouse button. The type and length of field appear on the bottom left of the display. Text fields are a little tricky in that they are not terminated with the Enter key but by Esc. Returning inserts an implicit <CR><LF> sequence into the string. Entering a city name, for instance, with an Enter in it will force all following text to the next line. You can enter multi-line character fields in this manner. Although the text may disappear as you enter more, re-editing it brings up the whole field formatted to the proper width and depth.

You can skip to the next record with the down arrow key. Up arrow takes you back one. <CTRL>Home takes you to the first physical record and <CTRL>End, to the last. While Ins *doesn't* insert a record (to avoid ordering problems), <CTRL>Del marks a record for delete. Footer information displays in red to indicate this. The record can be undeleted by keying Del by itself. You must compact the database to physically purge deleted record, an operation not supported within **MAPIT**. Records marked for deletion will not display but are changeable in order to get them back in case you change your mind. All changes you make to fields take effect immediately!

Another instance of this editor is tied to the EDIT/DBF Entities/Change and is designed specifically for editing DBF entity records.

## **Reset DBF Tiles**

**MAPIT** tiles are numbered 5 x 5 degree geographic areas designed to limit data search and speed repaint. This option assigns the tile field the correct tile number for a location. Externally created DBF's must make the tile field the key. You should manually delete the associated .MDX index file to force the creation of a new index. Mismatched tile/loc combinations or misordered tiles will prevent the display of the related entities. If you are working with large DBF's (several megabytes or larger), use the external program **TILEIZE** to assign the tile numbers and complete the job much more quickly.

## **Multi-Record Change**

A general DBF editor to change a field in all records to a particular value. Let's you preview the first *n* records you will change.

---

## **Exit MAPIT**

Terminates **MAPIT**. Alternately you can key <CTRL>Q to quit or exit from the system (if, for instance, you don't have a mouse).

Upon exit, **MAPIT** creates the save file `mapit.sav` reflecting changes in the system, most notably your last location and zoom factor. When **MAPIT** starts up again, the systems resumes where you left off. If a save file is damaged or deleted, a new one is created the next time you exit. Of course your old locations and settings stored in the original save file are lost.

Another file **MAPIT** may create is the shadow file, `data_file.shw`. A shadow file holds an image of the region data for a particular `.mp3` database file. Loading the shadow file is much faster than scanning the regions of a large database at startup. Shadow files aren't created for small databases. If a shadow file is damaged or deleted, a new one is created the next time you exit. Your first startup may be delayed while the databases' regions are scanned.

## **ZOOM**

Zooming is a method of drawing closer to or farther away from the detail of the portion of the display you are viewing. The initial zoom factor of 1 displays what is herein called the Standard World: the display centered on 0 latitude, 0 longitude with visibility from 180 degrees west to 180 degrees east. You can use one of a number of different methods to zoom in to see more detail, or once you are in, to zoom out for a broader, less detailed view.

Zooming in is necessary not only to see the additional undulations of a rugged coast, for instance, but to view new objects revealed only at the greater detail. To prevent screen clutter, many objects — major and minor rivers, national borders, state and provincial borders, and cities— are intentionally not visible except at higher zoom factors. When you zoom in too far, instead of seeing the smooth, natural-looking curves of coastlines, you see the underlying vectors making up the database. The maximum zoom factor supported by the data is dependent on the size and detail of the **MAPIT** database you purchased. Higher zoom factors demand much larger data files.

## **Redisplay**

This command clears the screen before repainting it with the current data. Use this function if you want to clear off old or temporary data.

## **Overwrite**

This command does not clear the screen, but paints over it with current data. You can use this function in conjunction with changing the visibility of certain objects to accentuate certain mapping features. For instance, if you are examining an area in which a border follows a small river and the border in question has been overdrawn by the river, you can accentuate the border without losing all small rivers by turning off small river visibility (DISPLAY/Rivers) and then choosing ZOOM/Overwrite. Because the Rivers display is toggled off, the rivers will not be redrawn after the overwrite and the border will be visible. Because you did not clear the screen first, other Rivers will remain displayed from the prior redisplay.

### **(Kill PCX Map)**

Turn off the currently-turned on PCX Map. Redisplaying or overwriting after selecting this item kills a PCX Map. Needed only after displaying a PCX Map.

### **In by Corners**

This is the most common means of zooming in. First choose one corner of a proposed new window by pressing the left mouse button and, with it still depressed, drag the mouse to the diagonally opposite corner of the proposed viewing window.

**MAPIT** adjusts either the width or height of the proposed window to maintain your screen's aspect ratio while guaranteeing that the area you asked to view (at a minimum) will be visible. Zooming In by Corners also changes the center of the display.

### **In by X Factor**

This command prompts you to enter a factor by which to zoom *into* the display while maintaining the current center. If your current zoom factor is 3, zooming in by a factor of 2 will result in a new zoom factor of 6.

### **Out by X Factor**

Here you can enter a factor by which to zoom *out* from the display while maintaining the current center. If your current zoom factor is 9, zooming out by a factor of 2 will result in a new factor of 4.5 (although the status line will display 4).

## **To Factor X**

If you know the exact factor to which you want to zoom, you can enter it with this command.

## **To Scale X**

Scale refers to the mapping scale, the ratio of the width of the display to the width of the scene displayed. One inch on a ten inch display equals 2,485.7 miles at the equator for a scale of 1:157,492,884.

scale	latitude	display width	zoom
9,000,000	0	1421 mi.	17
9,000,000	83 N	1421 mi.	2

While zoom factors are easy to work in and think about, professional cartographers think in terms of scale. Scale is latitude independent. Whether you're working at the equator or at 83 degrees north in Greenland, equal scales give, by definition, equal fields of view.

## **To Std World**

Standard World is the whole the world's entire width across the screen centered on latitude 0, longitude 0 and displayed at zoom factor 1 — the view **MAPIT** shows when it first started up. If you are ever lost, you can come back to this point.

## **POSITION**

**MAPIT** allows you to navigate over the face of the globe by ZOOM and POSITION. ZOOM controls the zoom factor (the width of the displayed area), and POSITION controls the center of the displayed area.

You can change the center of the display by several means. One means is ZOOM/In by Corners; with this technique, however, your choice of a new center is limited to the current viewing area. The POSITION commands offer explicit alternative methods.

**MAPIT** tries to maintain a constant zoom factor while changing location. This means that as one moves north or south, the scale will change.

## **Center**

Clicking the left mouse button after choosing Center repositions the screen's center to the location just clicked. The scope of the display (zoom factor) remains the same, but you can move up to a



half-screen's distance from the old center. Use this technique to center the display on some point of interest before zooming in for a close-up look. (By-X-Factor zooms are accomplished on the current center.)

Begrudgingly politically correct, **MAPIT** supports displays centered just about anywhere. You can, for instance, center the display at 180 degrees (still at zoom 1) to measure the shortest distance between Boston and Australia.

## ***Drag***

You can drag any point on the screen to be re-displayed at any other point. If, for instance, a feature near the bottom of the display is the topmost object you wish to see (that is, your interest is in features below it), press the left mouse button while over that near-the-bottom feature, drag to a point near the top of the screen, and release the left mouse button. Use this technique to move as much as a screen's distance from the old center.

## ***Last (TAB)***

Selecting the Last menu option (or pressing the tab key from the main menu) repositions the display to the last location/zoom combination. Selecting Last again toggles the display back to what originally had been the current location/zoom. Using the TAB key is a fast way of toggling between these two views. The only exception to this method is that the Standard World location is not included in this Last scheme because you can easily get to them by re-invoking Zoom/To Std World.

## ***Save as***

Save as is a method of permanently storing a location/zoom combination as a single letter or digit for later access. For instance, if you save the location of Bermuda as "H" (for Hamilton, its capital city), you'll be able to get back to that view of Bermuda from any location by Restoring H or entering H at the main menu. You can save up to 36 of these single character locations.

## ***Restore***

Restore is the formal menu-selected method of returning to any previously Save as location/zoom factor. The fast method is, of course, just entering the single letter or digit in which the target view is saved. Restoring to an unsaved letter results in no action.

---

## **TOOLS**

### **Temp. Distance**

Measure great circle distances between multiple points. (Position Menu entry not available.) Curves produced are in temporary graphics only and will disappear after a redisplay. See Great Circles below in this menu for creating permanent distance curves.

Temp. Distance computes the great-circle distance from one point to another while dynamically displaying both great-circle and constant-compass-bearing (rhumb) lines. For short distances, great-circle distances approximate rhumb lines which you would draw on a Mercator map with a ruler. Great-circle distances, however, are the physically shortest distances on a globe (as measured by standard crows). Great circles appear as curved lines on a Mercator projection, with the notable exceptions of the equator and all north-south running (meridian) lines. Radio waves follow great circles as did Charles Lindbergh.

To measure distance, position the cursor over the location from which you want to start measuring, press the left mouse button, and drag (with the button still depressed) to the place at which you want to end the measurement. Double-clicking halts measurement. **MAPIT** displays the result on the footer status line in statute miles, nautical miles, or kilometers depending on your current system of measurement. (See the DISPLAY menu.)

If you need to measure multi-leg distances, measure the first leg as noted above; at the end of that leg release the left mouse button. Then press it again before moving on to the next leg. You can back up (delete legs) with the right mouse button.

The To: and From: angles displayed right after the distance on the footer status line are the bearings from the starting point to the destination and from the destination back to the starting point. You might expect these bearings to have a simple reciprocal relationship: one's being equal to the other plus or minus 180 degrees. They do only for short distances. If you were going to fly your airplane or point your antenna, these angles would indicate the initial course for you to fly

The excess length of the rhumb over the great circle distance is dynamically displayed in top center status field along with the To bearing of the rhumb course. The From bearing is the To's reciprocal in this case.

You can measure great circle distances more than half way around the world. When your great circle flips over both poles, your track is more half way around the world and you are measuring the longer part of the great circle. The letter *L* will appear in the footer following the From angle, and the distance will be greater than 12,000 miles, the approximate half circumference of the earth.

## **Temp. Circle**

Dynamically enter a temporary circle of a given radius about a point. (Position Menu entry not available.) Curves produced are in temporary graphics only and will disappear after a redisplay. See Great Circles below in this menu for creating permanent circle curves.

Temp. Circle draws a circle around a geographic point of your choosing. As you draw it, its radius and the enclosed area are displayed on the lower left corner of the footer status line.

While actually a simple circle on a sphere, the range curve can be complex when projected onto a flat map. As the curve passes over one of the poles in the Mercator projection, it transforms into a sinusoidal open curve, which then, as it passes over the second pole, reforms back into a closed curve only to disappear as small circle on the opposite side of the world from its starting point. This strange behavior is really the path a ring would take as it grows from a single point on a sphere encompassing half the world before converging to a point on the other side of the earth.

To prove to yourself that all points on a given circle are equidistant from the center, use the Temp. Distance function to connect the center to a number of randomly chosen points on the range curve noting their measured distances.

You can cancel a Temp. Circle function in mid-command by depressing the mouse's right button.

## **Great Circles**

### **Permanent:**

Permanent/temporary toggle for all items in this menu.  
Permanent objects are stored in the extended file.

### **Distance:**

Dynamic (Multi-point) - Position Menu entry not available.  
Point to Point -

**Line of Sight:**

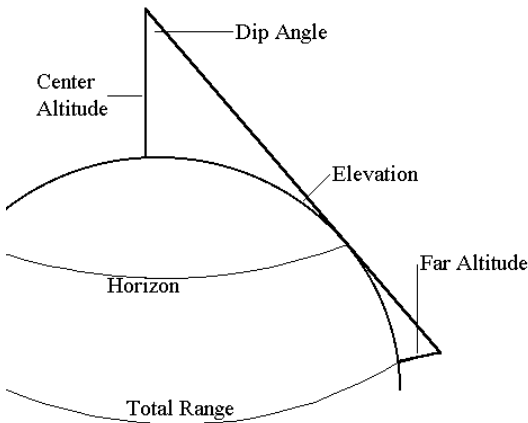
at Angle, From Location - What you see along a heading from a known point.

at Angle, To Location - Line on which you stand when looking at an angle toward a known point. Typical of coastal piloting or RDF work.

**Complete Great Circle**

at Angle, Through Point -

through Two Points -

**Line of Sight****Range Circles:**

Dynamic Circle - Position Menu entry not available.

Two Points -

Center & Range -

Line of Sight - Range from central elevated point to a far elevated object over the horizon or optionally limited to a dip angle or elevation. Use for satellite receiver range, air reconnaissance, strong light navigation.

## ***Database Lookup***

**MAPIT** supports database lookup of a number of different items related to cities, countries, states, US ZIP and telephone exchanges, geographic placenames, airports, and US towers. Once an item is located, you can go to its geographical position.

All queries share the following in common:

One or more fields in which to enter information. If you enter partial data, **MAPIT** will match the characters you have entered as well as any following as though requested with an implicit trailing '\*'. Entering "Oak" matches Oak, Oakdale, and Oakmont. In cases with multiple fields, the first field is a direct index into the related database and is fast. Following fields eliminate entries from the primary search which don't match these secondary fields by scanning through all matching primary keys, a potentially inefficient and time-consuming process for broad primary-key searches. For this reason, there are a number of different related queries using different primary keys. Some are faster for one thing, others for another. For example, you could look up all the cities beginning with the letter "S" in a particular country by using the "City Gazetteer" whose first field (primary key) is City and whose second field is Country. That process would be inefficient in that the query would scan through all cities in the world beginning with the letter "S" in order to find those in the country of your interest. A more efficient approach would be to use the "Cities in...." query whose primary key is Country and last key is City. Temper this warning, however, with the knowledge that the US accounts for some 75% of all cities in the database. If you're working with US cities, the difference between queries might not be as great in this case.

City	Country	State	County
swansea			
Swansea	Australia	New South Wales	
Swansea	Australia	Tasmania	
Swansea	United Kingdom	Wales	West Glamorgan
Swansea	United States	Arizona	Yuma
Swansea	United States	California	Inyo
Swansea	United States	Illinois	Saint Clair
Swansea	United States	Massachusetts	Bristol
Swansea	United States	South Carolina	Lexington
Swansea (Historica	United States	Nevada	White Pine
Swansea Center	United States	Massachusetts	Bristol

Click on city to select.

 Lookup/More Return & Display Restart CancelFinished Total found: 10

Please note. Country names may not be as you think. England is listed as a "state" in the country of United Kingdom!

As you enter the query menu, the mode you are in appears in the left footer: "Select modifiers and lookup". Click with the left mouse button over one of the top query fields to select that field for data entry. (Clicking with the right mouse at this point escapes you out of the query menu.) The query is case-independent. Enter a return or a tab to complete the field. You may also click either mouse key to end the entry. Click with the left to select the next field. When you are finished entering all query fields, click the "Lookup/More" field near the lower right section of your screen. (If you didn't end your last entry with a return, you'll have to click twice.) **MAPIT** performs the search displaying the results in the middle of the screen. As it finds matches, it updates the counter in the right footer. When the query is completely satisfied, the message "Finished" appears in the left footer. If there is more data to be displayed, the message is "More" and you must continue to click on "Lookup/More" to display the next page of data. At any time you can re-enter new data in the query fields, but for the lookup to act on a fresh primary key, you must click on the "Restart" field to return to the "Select modifiers and lookup" mode. After that you can select the "Lookup/More" field. Results from new queries are appended to old results. If you highlight any displayed result and select the "Return & Display" field near the bottom left of your screen, the query will end and your screen will be centered on the location associated with that entry using the current zoom factor. You can abort a query by selecting the "Cancel" field or by clicking the right mouse button. Please note that if you try to highlight an area of the results screen where there are no results, **MAPIT** will

display the error message “DB read failure on record 0.” Click or return to clear. You’ve done no harm.

## Available Database Searches

Title	Description
City Gazetteer	Alphabetical list of all cities.
Cities in....	List cities by country/state/county
Cities by State	List cities by state/county/country
Phone/ZIP’s by City	List US phones/ZIP’s by city and state. Phone numbers are area codes plus exchanges (up to 6 digits) with no blanks, dashes, or parentheses.
US Cities by ZIP	List phones/cities by ZIP and county.
US Cities by Phone	List ZIP’s/cities by phone.
Country Gazetteer	List countries of the world with International Dialing Codes.
State (Abbr.) in....	List states/(US abbreviations) by country.
Counties in....	List counties by country/state.
Placenames	List geographic placenames.
Airports	List many airports of the world.
Towers (US)	List US fire and radio/television towers.

## File Maintenance

Each database query has its own maintenance screen controlling database name, index, and query program. Note that the paths are relative (do not reference an absolute path beginning with the drive’s root) to the subdirectory “dbqs” where all query data is stored. There should be no need to make changes.

## Plot HP-GL/2

Output high-quality graphics to an HP LaserJet III (or better) laser printer, a plotter, or to an HP-GL/2 [or HP-GL] graphics file for import into an external graphics program or word processor. To change the default pen assignments, line type or width assignments, or plot borders or lat/long grids assignments, see the example pen definition file `std.pen`. Change a copy and invoke **MAPIT** with the command line calling parameter `/pendefinition=my_file`. See Appendix C: Pen Definitions for a table of colors and a listing of `std.pen`.

---

The Landscape/Portrait choice refer, of course, to the plots orientation: horizontal or vertical and the Page Width and Height to the plot's dimensions in inches.

Text Scale refers to the relative size of plotted text versus other graphics objects and defaults to a value of 1.0X. Changing the Text Scale to 2.0 doubles the size of the text.

Borders refers to the heaviness of the lines which default to the 0.35 mm minimum on the LaserJet III. If you specify a lesser value, the minimum is used. Notice that Borders default to 0.5 mm being slightly heavier than default lines. You can also toggle borders, which frame the plot, on and off.

Detail is a relative value (defaulting to 1.0X) indicating the detail of the data sent to your plotter. A detail of 1.0X will ship vector data in the detail normally displayed on your screen. **MAPIT** invokes dynamic line segment pruning during display to speed graphics repaint and to minimize clutter caused by many small single-pixel entities. While this works well for the relatively course environment of your display, plotting devices commonly support much finer resolution where the extra detail may be welcome. Increasing the detail to 2.0X, 16X, or FULL causes **MAPIT** to pass more detailed information to your plotter.

If you are experiencing memory overflows on rasterizing (laser printer) plotting devices, reducing the detail to 0.7 or 0.5 may drop the output vector count to the point where entire plot fits within the device's memory. (Not plotting cities and text also helps.) Naturally the quality of the plot may suffer noticeably.

Deep Scan is a toggle indicating the depth of database scan to be used in retrieving plotting data. Without it, a plot uses the same portion of the database used by the display. Below zoom 24, several less-detailed versions of the data are accessed to expedite repaint. Selecting Deep Scan forces **MAPIT** to reach deeper into its database and retrieve the most detailed version for plotting. If you're already at or below zoom 24, Deep Scan will have no additional effect. .

The Printer/Plotter versus File choice selects whether the output is sent to your system's PRN device or to a file. If you select a file, **MAPIT** prompts you for a file name where the default file name extension is .hgl. Three other choices indicate whether this must be a new file, may be an old file which will be overwritten with the new data, or may be an old file to which the new data is appended. This final choice allows you to batch a number of plots into one file assuming your plotting device supports multiple plots.



PCL Header indicates whether you are plotting to a Hewlett Packard PCL 5 (Printer Control Language) supporting device such as an HP LaserJet laser printer. If your output will eventually go to a word processing document or to a pen plotter, the PCL Header probably is not desirable. Its purpose is to switch the printer from text to plotting mode.

The Plot Full Map/Plot Selected Area menu choices begin the plotting process. In either case the selected area is guaranteed to be plotted with more added to the sides or to the top and bottom depending upon the selected aspect ratio (Landscape/Portrait and plot dimensions) versus the chosen area. Plot Full Map, of course, selects your entire screen, a Landscape conformal area. Plot Selected Area gives you the chance to select the corners of a rectangle defining your plotting area. Your original chosen area is outlined in red while the recalculated (to fit the aspect ratio) area is outlined in yellow. You are given a final chance of aborting or continuing the plot after the actual plot area is outlined.

Abort is the most direct way of aborting the plotting process before you begin.

### ***Plot PCX***

With Plot PCX, you can output colorful .PCX screen shot files directly without the need for screen capture by an external program.

Invert temporarily switches black and white map colors without changing other colors during creation of the output .PCX file to lighten the effect of the otherwise black background.

You must choose a File name for output. The default name extension is .PCX, a format read by many paint programs including Windows' Paintbrush. The overwrite/new only choice indicates the action to be taken if the named file already exists.

You have the choice of plotting the Full Screen which includes the menu and status lines, the Full Map, or a Selected Area. With the latter choice you must select the corners of a rectangle defining the area to plot. The fact that .PCX screen dumps are limited to the resolution of your screen, by selecting only a portion of your screen with this latter choice you may be substantially reducing the resolution of the captured area as well as reducing the size of the output file. In other words, best results (and largest files) are obtained by selecting as large an area to plot as possible.

You may still need to use external screen capture such as Print Screen from within a Windows DOS box (see Making Maps/Screen Capture) if you need to show such **MAPIT** details as a pointing cursor or the menu system in use.

## **GPS/Markers**

Display real-time moving markers communicating with **MAPIT** through a dBASE IV interface file. **MAPIT** cooperates with **GPS**, an external 16 bit Windows program which updates file GPS.DBF with live data from the NMEA 0183 output cable of a GPS receiver. **SIMULATE**, a DOS-based simulation program included with **MAPIT**, also drives markers. You can use your own external program to update one or more database records and monitor the positions by moving markers. The square markers may be labeled showing a name, heading, speed, and time, as well as displaying an arrow indicating direction of motion. The frequency of database polling to detect changes is variable with lower limits in the sub-second range. The display can be tied to follow one marker if it moves out of view.

Unlike entities which are stored in the .MP3 database and are limited to a locational accuracy of approximately +/-100 feet, the placement accuracy of moving markers is limited only by the precision of the data passed through the interface database.

*Active* must be selected for **MAPIT** to read the database specified under *Interface DBF*. If the user exits **MAPIT** with a database open, upon restarting, **MAPIT** tries to open the same database with the same display conditions. Note that while *Active* in the context of this menu refers to **MAPIT**'s attempt to open and process an interface database. In order to be displayed, each record in said database must also have its "active" field set true by its controlling external program.

*Min zoom* is a threshold value indicating at what zoom level markers will become visible. The default value of 0 implies that markers will always be visible. (At the Standard World display, the zoom is 1.)

*Color* is the default color number for displaying markers when their corresponding database records don't specify a particular value (*i.e.* have a value of -1). The default default color number is 4 or red.

*Scan delay* provides a lower bound for the delay between subsequent reads of the database and the display of updated markers. The default is once a second. When **MAPIT** is idle

(you're not in a command or menu), **MAPIT** will read the database and update marker locations no more frequently than once a *Scan delay*. Database processing and marker display times are dependent upon the number of records read from the database and on the amount of information displayed with each marker and cannot be guaranteed. If you specify a one second delay, that delay will be introduced between each redisplay of the markers. Working in **MAPIT** will stop the redisplay until your activity is over. You can set the delay down to 0.01 seconds, actually to one clock tick, or 1/18th of a second. The flicker rate of the mouse icon may increase as you shorten the delay.

You can auto track or follow one marker icon. If a marker moves off the screen, **MAPIT** will move the display window to follow it. *Auto track* is a dual-function menu item. It toggles auto tracking on and off, and if you don't select the default entry with a simple return, accepts input of a new marker name. The default marker name is "GPS", but you can enter any marker name from the current database. Be careful. **MAPIT** doesn't allow you to select a name not in the current database and will revert to the old name.

*Marker trails* retains the temporary graphics of the square icons showing the position of past locations of markers. Because these trails are retained in temporary graphics only and are not saved as permanent entities, they disappear the next time the screen is redrawn.

*Display name* toggles the name line of moving marker icons. A name may be up to 20 characters.

*Display speed* toggles the speed/heading line of moving marker icons. The units of speed are determined by the **GPS** program or other program which feeds the interface database. The value stored in the speed field is used directly. **MAPIT** attempts no conversion to other units.

*Display time* toggles the time line of moving marker icons. This field indicates when the database-updating program last updated each record. In fact, **MAPIT** will update a marker only if its active field is true and if the time is different from the time the marker was last displayed. Note that the time displayed by GPS units may be Universal Time (GMT) and may not reflect your local time zone.

## **Register PCX**

**MAPIT** supports the display of single-screen .PCX files overlaying regular **MAPIT** entities. The purpose of this feature is to support using maps of very small areas as backgrounds for **MAPIT**

operations. If, for example, you wish to display the boundaries of a farm or a state park copied from another document, **MAPIT** must know where to place this detailed overlay and at what scale. This process is called registering and involves matching two (distant) points on the .PCX overlay with two points or lat/lon locations in **MAPIT**. This menu option supports that registration process. Once registered, a .DBF holding registration information can be called and displayed by a type 'M' scan table record. The .PCX file must be a 16 color image, ideally the type created by the old Windows 3.1 PaintBrush program. **MAPIT** displays .PCX files using its own color palate and may thus change the colors of the .PCX file. 256 color files don't map properly and are unusable. Convert them first to 16 colors using programs such as Paint Shop Pro.

Work from top to bottom in the menu and pay attention to instructions in the footer.

*Display PCX* accepts the file name of the .PCX you are registering and displays it.

*Align PCX* is the critical command in this sequence. It prompts first for a point on the PCX which you must pick with the mouse and then asks for the corresponding point on the map which you may enter with the mouse or by position menu. (A return.) It does this for two points. **MAPIT** will display the registration you've just entered for you to check. You can repeat this step any number of times. Registration implicitly establishes a scale which **MAPIT** will return to every time you access this PCX overlay.

*Name map* is the text by which a registered PCX will be known by in **MAPIT**. It will be displayed below the PCX map symbol (the letter "M" enclosed within a rectangle).

*Register in* queries you for the name of map entity DBF which will hold references to this and perhaps other PCX maps. If no file exists, **MAPIT** asks if it should create it. Give the name of a file on a writeable drive (e.g. C:pcxfiles) and not the readonly CD-ROM.

The file you created or added to in *Register in* must be referenced in your scan table for your PCX maps to display. See FILES/Add Scan Table Record. The entity type field E\_TYPE must be set to "M".

## **EDIT**

Move, duplicate, change, insert, and delete objects or entities in various **MAPIT** databases.

### **DBF Entities**

Move, duplicate, change, insert, or delete .DBF point entities.

#### **Move**

This command uses a selection sequence common to many other commands in **MAPIT**. Select an entity by positioning the cross hairs over the origin of the DBF entity you wish to move. The origin of DBF entities are commonly a point in the center, the user-defined origin point of figures, or a point at the lower left corner of the first character in text. Click with the left mouse button to make the selection or with the right to abort. If you miss with the left click and don't select any entity, you can click again immediately. Your selected entity will become highlighted and a message in the footer will display the action and entity type. (See the type table in Add Scan Table Record.) A floating menu asks, "Correct entity?" Answer "Y" or "N" to proceed or not. An affirmative response returns control to the cross hairs. Move and click with the left button to choose a new location, with the right to abort the process, or use the Enter key to bring up a keyboard-controlled floating menu for exact positioning. This sequence is typical of selecting DBF entities.

The move command changes the location of DBF entities.

#### **Duplicate**

Duplicate uses the common selection sequence to replicate the original entity in a new location. The original entity can be duplicated any number of times and can be located by either left mouse-click positioning or by keyboard entry. The new entities are stored at the end of the original entity's data file.

#### **Change**

Change invokes the common dBASE Editor described in FILES/Edit DBF automatically opening that record in the dBASE file corresponding to the selected entity. The dBASE Editor edits the entity record as a general DBF record and without any special intelligence. Therefore you should avoid changing the key field (TILE) or the LATITUDE/LONGITUDE fields from which the TILE's value is calculated. (Small changes within the same 5 degree by 5 degree tile are okay. Use the Move command which updates these

fields properly.) An improperly assigned TILE number may allow it's entity to display in certain circumstances, but not in zoomed situations. Also you won't be able to select the entity with the Change command to change it back. (Use the FILES/Edit DBF command to access and fix the errant record.) You can use the Change command to determine in which file an entity is stored. The file name and record number are displayed in the footer.

The database scan caused by the Change command displays a small dot at all entities' origins in the highlight color in the 5 by 5 tile in which you click. If you aren't sure exactly where an entity's origin is, clicking near it with the Change command will cause it to be highlighted. Click near enough of course, and you'll select the entity itself, which is generally the point!

Change is useful in that it allows access to entities marked for deletion permitting you to undelete them. (<CTRL>Del and Del) Because deleted items aren't normally displayed, use the click-in-the-same-tile technique described above to display the origins of deleted entities. Clicking the origin of a deleted entity will display it temporarily to allowing you to determine if you've got the entity you want.

If two entities overlie each other exactly, you can access only the top one. If you want the bottom one, mark the top one for deletion and then use the Move command to move the now-exposed bottom entity slightly to allow direct access to it. Select and undelete the old top entity.

### ***Insert***

Insert asks you to enter an existing database name to which to add a new **MAPIT** DBF entity. If you want to create a new database, use the FILES/New DBF menu item to create the database and its first record. Insert determines the type of entity you are adding by accessing the first record of the database. You are prompted to select a location for this entity using the cross hairs or keyboard entry. Finally the Insert command invokes the dBASE Editor to allow you to personalize the new entity further.

### ***Delete***

The Delete command is a short-cut way to select and delete DBF entities. After you have selected an entity with the mouse-controlled cross hair, Delete displays both the entity's file name and record number and asks you if you are certain you wish to delete it. You can always undelete it with the Change command by entering the Del key from within the DBF Editor.

## Layers

A layer is really nothing more than a number assigned to each **MAPIT** entity. Color and visibility are associated with these numbers or layers. **MAPIT** supports up to 256 layers. Layers 0 through 119 are reserved for the main **MAPIT** database. Layers 120 through 255 are left for user input. By default, objects you enter are placed on the Current Layer and are displayed in that layer's color.

The sub menu displayed by the Layers command shows at a glance the Current Layer, its color, and whether all, some, or none of the layers are visible.

### Current Layer

Use the Current Layer command to reset the Current Layer to a different value. All entities defined after this time will be assigned the new Current Layer. Already-defined entities will retain the Current Layer assignment in effect when they were created. Entering a return with no other value after display of the default value [nn] is a do-nothing action which retains the current Current Layer.

Layer 120 is the default Current Layer.

### Color

Color is a display menu item showing the color assigned to the Current Layer. One color is assigned to each layer.

### Change Color

Change Color displays yet another sub menu which you can use to display and change the color of any layer. Assigning new colors to layers allows you to control the colors with which entities are displayed. You could, for instance, turn all rivers from blue to red by reassigning the layers on which rivers are defined to color number 4, red. You can assign any color to any layer you wish. Objects on that layer will be shown in that color the next time they are displayed if they themselves default to the layer color (have a color of -1).

Clicking on Layer changes the focus to another layer and displays its color swatch. Use Color to assign a new color number. *All to default* is dangerous. With one click you can undo all your past color reassignments.

<i>Color</i>	<i>Color Name</i>	<i>Color</i>	<i>Color Name</i>
--------------	-------------------	--------------	-------------------

<i>Number</i>		<i>Number</i>	
0	black	8	gray
1	blue	9	light blue
2	green	10	light green
3	cyan	11	light cyan
4	red	12	light red
5	magenta	13	light magenta
6	brown	14	yellow
7	white	15	bright white

There are 16 colors, 14 of which are assigned to layers on a rotating basis. Colors 1 through 14 are assigned to layers. **MAPIT** uses Black to overwrite and delete and Bright White to highlight during object selections.

See Appendix A for a complete layer color assignment chart.

## ***Visibility***

All layers begin life in an active state but can be turned off to make their entities invisible for display purposes and immune from any EDIT commands. Turning off a layer does not remove from the screen already-displayed objects that reside on that layer. They will be missing on the next repaint. This technique is how the DISPLAY toggles work. Similarly, you can Copy & Delete only objects which are on "on" layers. Turning off a layer or range of layers is like read/write-protecting them.

### ***All visible***

Clicking All visible turns on all layers.

### ***Make visible***

Make visible allows you to turn on a single layer or a range of layers.

### ***Make invisible***

Make invisible allows you to turn off a single layer or a range of layers.

### ***All invisible***

Clicking All invisible turns off all layers.



## ***Private File***

Display the name of the current private or extended file, the file to which all line or .mp3 entity insertions are stored. Use the calling argument `/extended=file_name` when starting MAPIT to assign a different private file.

Note: Menu items below this point refer to objects stored in the extended .MP3 file. Use DBF Entities above for the new style .DBF point entities. Lines must, of course, always be saved in .MP3 files.

## ***Insert***

Insert objects into the extended database on the current layer and displayed in the color of that layer. The extended data file is updated at the completion of each entity's entry reducing data loss because of power failure.

With the exception of line and xline, the entities described in this section are point entities which can also be held in .DBF entities described above.

### **city**

Enter a city by filling in the information requested by the entry screen. **MAPIT** will initially ask for a name and a location before continuing to the entry screen. The Min and Max Zoom Factor are those zoom factors for which the city first appears and then disappears. Zero means it never disappears. Above, Even, and Below refer to the choice of placing the city's name exactly to the right of the marker or offset above or below to reduce conflicts with nearby cities. A forced color is a color number which takes precedence over the default color of the assigned layer. Replace inserts the new city data at the initial requested location while Move allows you to reposition the city which you entering by clicking with the left mouse button. Abort, of course, aborts the entire entry process.

Displaying cities of all sizes, all the time, would clutter the screen. Therefore cities are made visible by zoom factor according to size:

Population	Minimum zoom (US)	Minimum zoom (elsewhere)	Font	Marker
> 500,000 or national	10	10	1	double circle

capital				
> 50,000	30	30	1	circle
> 5,000	300	50	1	small circle
> 0	750	150	4	small circle

### **figure**

The figure option is a powerful means of replicating miniature simple line drawings throughout the displayed map. You must define figures before using them (see FIGEDIT), storing them in the file `std.fig` and inserting each figure by name in a map's database. Multiple instances of a figure may be inserted. The actual figure is not included in the database, only a reference to it by name. For the figure to be displayed properly, the figure file you used while inserting your figures must be present. You can, of course, globally change a figure by replacing it in the figure file with another of the same name or using an alternative figure file with similarly named figures.

Choosing the Insert/figure menu option brings up a window containing a list of figures in your figures file. As you move the mouse over the name, the name of the current figure is displayed in the status line at the bottom of the screen. Depress the left mouse button to choose a figure or the right one to abort. Cross hairs appear to help you in the initial positioning of the figure while the changing latitude/longitude is visible in the right half of the status line. Pressing the left button positions the figure and opens the menu pictured above.

Use this final menu to change the figure's Scale (size), Layer, Forced Color if any, and to Save as is, Move, or Duplicate. Duplicate means to place additional copies of the figure at other locations with the left mouse button finishing with the right button.

### **line**

You can enter multisegmented lines of any length. Hold the left mouse button down and notice the temporary "rubber band graphics" tying the starting point of the line segment to the current mouse position. As you move the cursor, notice that the latitude and longitude on the status line are continuously updated to aid in accurate line positioning. Releasing the left button fixes the last line segment.

If, while you are close to the end of the last segment (the release point), you press the left button again, you'll begin a new segment connected to the last. On the other hand, if you click the left

button while you are a distance from the end of the last segment, the line is ended, the data is stored, and the temporary graphics are replaced with a permanent line.

If you click the right mouse button while inserting a line, one segment of the temporary line is deleted for each press and release. If you try to back up beyond the start of the first line segment, the entire EDIT/Insert/line command is aborted. No data is stored.

There are limits of approximately 100 feet (somewhat less at higher latitudes), to the precision of data entered into the database. If you try to enter data with more precision than this, the temporary graphics will look good but will not be permanently reflected in the database, as your first repaint will demonstrate.

### ***xline***

Xline refers to exact lines which can be placed more precisely via the keyboard.

### **stroked text**

Stroked text is drawn directly over a portion of **MAPIT**'s display. The program first asks you to choose a temporary location on which the text will be drawn as you compose and change it. Choose a location which will give you adequate room to see and deal with your text; you can change to the permanent location later.

As you enter the text from the keyboard, the stroked text will appear in default font and size at the temporary location. Special characters you have mapped to your keyboard can be used, even though they may not be displayed properly in the default font. (Function Keys F1 through F12 are mapped to ASCII characters 20 through 31.) Entering a return takes you to the start of the next line under the first character of text you entered. You can correct text by backspacing to the incorrect letter(s) and reentering. Press **Esc** to exit the text entry mode.

Next, the program displays an option menu with which you can change the attributes of your text or edit the text itself by choosing Text. In the edit mode, your current position is at the end of the text string you've previously entered.

On this menu, Min and Max Zoom Factor refer to the zoom factors at which the text first appears and disappears as you increase zoom. If you leave the factors at zero, the text will first appear when it grows to a legible size in the case of Map Relative text

(see below for discussion of the Map Relative text entry mode). Remember that, for text to be visible, its origin (the insertion point) must be visible. The text may disappear as you zoom in, unless its origin is exactly in the center of your window. This happens because the origin has fallen outside the window.

Seven fonts are supplied in the **MAPIT** file std.fnt; the system capability is ten:

Font 1 is the default. If the std.fnt file is missing at program startup, stroked text will not display. If you call for a font that has not been loaded, the system uses font 1 in its place. Font 1 is the simplest and displays fastest. The more complex a font (that is, the higher its number), the longer it takes for a given text string to display.

The two modes for stroked text entry are Map Relative and Screen Relative. Text which is Map Relative scales with the size of the map display as you zoom in and out. If you zoom out far enough, it becomes too small to be seen. Screen Relative text, on the other hand, is always the same size relative to the screen no matter what the zoom factor. It never grows or shrinks.

Size is an arbitrary number that is proportional to the size of the text on the screen and inversely related to the current zoom factor. The system suggests a text size for which the newly entered text will appear the same height on the screen independent of zoom. Consequently, the greater the zoom factor, the smaller the suggested text size. If you enter Map Relative text size of 0.2 at one zoom value and 0.2 at another zoom, the text will be the same height in both cases. The maximum size for Map Relative text is 10. Screen Relative text will always display at the entered size independent of zoom.

Not all sizes of Map Relative text are legal. This fact is most noticeable for large sizes and results from the encoding mechanism  $10/n$  where  $n$  is a positive integer. A value of  $n = 1$  yields a text size of 10, 2 yields 5, 3, 3.333, and so on. A Map Relative text size of 8, for instance, is impossible to attain, its  $n$  integer falling between 1 and 2 (1.25 exactly). This shortcoming is not as significant for smaller text sizes.

Stroked text can be entered at any angle divisible by 2. If you try to enter text at 45 degrees, for instance, **MAPIT** rounds that value to 46 degrees and then draws the text rotated around its origin at that angle. Zero degrees is horizontal (right reading) and 90 degrees is vertical (up reading).

You can force the text to be placed on a layer different from the default layer and to have a forced color if so desired.

Finally, you can either Abort the text entry, Save the text as is, Move the text to another location, or choose one or more additional locations. When you choose locations, the cursor changes to cross hairs for more accurate placement and the status bar displays the exact latitude/longitude of the cursor. You can insert the text multiple times in different locations by clicking the left mouse button. Click the right button to abort/finish the operation.

### **hidden text**

Hidden text hides behind a small triangle until you bring it forward by clicking the triangle with the left mouse button. When you enter hidden text, the entire screen is filled with a non scrolling window on which you can type any text as you would with a simple editor. The program uses the default screen font. Word wrap is automatic. The **Enter** character is the paragraph delimiter. As in stroked text, press **Esc** to terminate text entry. (You can return to edit hidden text if necessary.)

After you have terminated text entry, the program displays an option menu with which you can change the layout of your text:

Min Zoom Factor is, again, the zoom factor at which the hidden text marker first becomes visible. This value defaults to about 2/3 of the zoom factor in effect when the hidden text is entered. A value of 0 or 1 assures that the marker is always visible. (Hidden text markers are Screen Relative; they are of constant size and always visible except as limited by Min Zoom Factor.)

Window Width and Height display two values each. The first is the current value and the second is the computer's suggested setting. There is interaction between width and height. The computer calculates the minimum width based on the longest line and the screen width, and given that width, calculates the required height. If you enter a new width or height, the computer recalculates its suggested values. You must then change the other parameter (height or width) to the suggested setting, or reset your previous entry. Sometimes several iterations are necessary to produce the window you want. At any time you can View/Edit Text to review the current arrangement or to edit the text. View/Edit (Autosize) resets actual values to **MAPIT's** preferences.

In the absence of any overriding standards or artistic opinions, you will probably want to use wide short windows; they are easier to

read than narrow tall ones. When you view hidden text, **MAPIT** opens whatever size window you have created and positions it as close to the marker as possible.

Choose Location(s) or Abort as you would for Stroked Text to insert zero or more instances of this hidden text.

## **Selecting**

Several of the following EDIT commands share the concept of selecting objects on which to perform an operation. Selecting consists of drawing a rectangle on the screen using the mouse. The database[s] are scanned for objects falling within that rectangle; when found, the objects are highlighted in bright white. (The database scan can take a significant time.) At the end of the highlight process, **MAPIT** asks the user if he wishes to continue. The user responds affirmatively with **Y** or **y** (yes), or negatively with any other input. The command either continues or aborts. The yellow selection rectangle remains to indicate the extent of the operation.

To abort any selected option at the start, click on and off without moving the mouse. This creates a null-sized selection rectangle.

## **Change**

Change .MP3 objects once they have been inserted into the extended database. If you try to select an entity on your screen and it doesn't highlight, it either is not a .MP3 entity or it is not in the extended file where it can be changed.

### **city**

Position the mouse-controlled cross hairs over the city to be change and click the left button. A menu similar to the Insert/city menu provides control.

### **figure**

You must select a figure by including its origin in a selection rectangle (see above). Once selected, a menu similar to the Insert/figure menu provides control. Note: If you change an existing figure's property such as color and then choose Duplicate, the color of the original will not be permanently changed but will upon repaint display as it was originally. The duplicate figure, however, will permanently display the new changed color. If you want to change the original, you must Replace it.

**line**

To change a line, you must include at least one of the line of interest's nodes within the selection rectangle. **MAPIT** highlights the line when it is selected and a menu displaying the Layer, Forced Color, and the commands Replace, Move, and Abort appears. Replace applies any layer or color changes to the existing line. Move permits movement of each of the line's nodes to a new location (within the approximate limit of 7 degrees from the line's first node). The cross hairs jump to the selected line's first node if visible.

When the cross hairs are close to any node, a rectangular box appears around that node's position: you can move the selected node by depressing the left mouse button, dragging it to a new location and releasing the button. You can do this for any number of nodes in a single line. Click the left mouse button outside a selection box to complete the change to the line. Clicking the left button aborts the line change.

You cannot change a line in the main database. (If you really want to, make a backup of the main database and open **MAPIT** with a nul main database and /extended=world. All changes will then be made to what is normally the main database. Be careful!)

**stroked text**

Select stroked text by including its origin in a selection rectangle. Recall that the origin of stroked text is at the baseline and left of the first letter. Once selected, a menu similar to the Insert/stroked text menu provides control.

**hidden text**

Position the mouse-controlled cross hairs over the hidden text to be change and click the left button. A menu similar to the Insert/hidden text menu provides control.

**Delete**

Deletes the selected objects from the extended but not from the main database. Note that deleting while you are zoomed in very close may eliminate not only visible objects, but those which are invisible at your current zoom factor. Deletion does not shrink the database.

---

## ***Assign to Layer***

Move selected objects in the extended database from their current layers and assign them to a new layer. Object data are not moved. Both visible and invisible objects are reassigned.

## ***Copy***

Copy selected objects from the main to the extended database. If obviously visible objects are not selected, they may well be in the extended database already. Objects cannot be duplicated in the same database. Both visible and invisible objects are copied.

The **MAPIT** copy function *does not* preserve original regions. (See the discussion of regions under **Working with MAPIT Files/Regions**.)

## ***Copy & Delete***

Copy selected objects from the main to the extended database and then deletes the originals. The purpose of this command is to transfer data once and only once from one file to another. You must invoke **MAPIT** with the /delete command line argument because the main database is normally opened read-only. If obviously visible objects are not selected, they may well be in the extended database already. Both visible and invisible objects are moved.

## ***DISPLAY***

Control the display of **MAPIT** features Clicking on selections toggles on and off the layers on which these features are stored.

## ***Grid***

Toggle on and off latitude/longitude markings. Turning off the grid sets previously grid-colored pixels black. These black pixels may obscure object detail until the next repaint or overwrite.

## ***All Features***

Displays or changes the status of which layer features are active. '?' indicates some are and some aren't. A check indicates all are. Cycles through the choices. Selecting features for display is accomplished in the following menus or by turning layers on or off directly in the EDIT/Layers menu.

The following major feature types call menus controlling the display of specific features/layers. Selecting all features may,



depending on the zoom factor, slow down screen repaint and result in a jumble of confusing lines. Not all features are available at every zoom level. Small rivers become visible at zoom 24 and beyond. (Major breaks are at zoom 2, 6, and 24 for line (.MP3) entities.) Point entities such as cities and text are also phased in at different scales. Some cities become visible at 1:4,000,000, some at 1:2,000,000, and the smallest at 1:1,000,000. The thresholds for point entities are set in the scan table which you may modify. For instance, you could set all cities to be visible at any scale and in the Cities & Landmarks menu turn off the text to show only the symbols. Displaying at a zoom of 1 would give an indication of population density around the world although the screen would take quite a while to paint as **MAPIT** processed some 200,000 cities.

***Boundaries:***

***Rivers:***

***Roads:***

***Railroads:***

***Utilities:***

***Transportation Structures:***

Please note that airports are controlled by Transportation Text under Transportation Structures. Data near England, Japan, and India is missing entirely and is unavailable.

***Cities & Landmarks:***

***Ocean Features:***

The oceanographic contours are, in general, 1:3,000,000 scale while the ones marked as approximate are much less detailed. Plate Boundaries are self-explanatory. Fracture Zones are lines of weakness in the earth's crust that cross the diverging sea floor boundaries at right angles forming short areas where plates grind past one another. Magnetic Lineations are more mysterious but appear to be small local fields induced by the magnetic minerals contained in the spreading volcanic rock masses comprising much of the ocean basin floor.

**Land Cover:****Units of Measure**

Cycle through Statute Miles, Nautical Miles, and Metric. Used in the display of distances.

**Lat/Long Format**

Cycle through Degrees, Minutes, Seconds, and Decimal Seconds. Used in the right footer to display the location of the mouse.

**Screen Width**

Measured in inches to calculate the scale displayed in the footer.

**Version**

Particulars about this product and its operation. Virtual memory refers to the amount of conventional memory dedicated to supporting virtual memory (XMS, EMS, or disk) activity. Free refers to the actual amount of conventional memory unused by **MAPIT** but necessary for proper operation during certain commands. This figure never grows too large, the excess being allocated to virtual memory. The number and size of regions reflects the amount of data in your main and extended databases.

**Working with MAPIT Files:****File Names and Conventions**

**MAPIT**'s mapping data is stored in special binary files having the extension .MP3. These files have a highly structured internal format designed to speed the display of data on the screen. These files also have the useful property that two or more .MP3 files can be concatenated together to form a new larger file. (See the example below of using the DOS *COPY* command with the */B* switch.) If the internal structure of a .MP3 file becomes corrupted, **MAPIT** responds with an error message stating that it has lost sync during a file read. No data beyond that point is available to the user. The file must be replaced.

**MAPIT** uses two default file names for storing .MP3 data: **WORLD.MP3** for its primary database and **EXTENDED.MP3** for the current private database. You can run **MAPIT** against other primary and extended databases by explicitly specifying their names as you invoke **MAPIT**:

```
C:> MAPIT alt_db /extended=private
```

**MAPIT** expects alternative databases to have the .MP3 extension unless you explicitly invoke them with other extensions.

When you enter data into **MAPIT**, the data is stored in the private file EXTENDED.MP3. You should use private databases when storing your own data or transferring it to others. The smaller file size makes these actions much more efficient.

You can combine several of your private files with the binary *COPY* command:

```
C:> COPY /B FILE_1.MP3+FILE_2.MP3+... BIG.MP3
```

or

```
C:> COPY /B FILE*.MP3 BIG.MP3
```

Remember that, once files are combined, they can't be separated. Don't delete the originals until you are sure everything is all right. And don't combine your private data with the primary **MAPIT** database WORLD.MP3.

## ***Private Database Management***

As mentioned above, **MAPIT** uses the default file name EXTENDED.MP3 for storing your private database. Suppose you want to make and support two private databases, each to be distributed to a different set of users. The first involves the Philippines and the second Hawaii. Obviously you don't want to use the default private database name EXTENDED.MP3. When you're working with the Philippines data, start **MAPIT** using:

```
C:> MAPIT /EXTENDED=PHIL
```

and when working with Hawaii, start **MAPIT** using:

```
C:> MAPIT /EXTENDED=HAWAII
```

Now you have two separate .MP3 files, each with its own data. If you or your users want to view that data, you can do so directly (without the benefit of **MAPIT**'s database) by entering

```
C:> MAPIT PHIL
```

More than likely, you will want to use the backdrop of **MAPIT**'s database in the normal way. If in the future you have a user who wants to use both the Hawaii and Philippines data at the same time, combine them with the binary *COPY* command:

```
C:> COPY /B PHIL.MP3+HAWAII.MP3 COMBO.MP3
```

---

## Regions

If **MAPIT** had to traverse the entire 250MB WORLD.MP3 database every time it re-displayed an image, screen repaints would be extremely slow. The primary method **MAPIT** uses to speed the display from large databases is to divide the data into regions. Regions are special database header records whose extents are defined by the mapping data following them.

Each .MP3 must begin with a region header. When you add objects to **MAPIT**, the first record written to a new EXTENDED.MP3 file is a region header. As you add more and more data to the personal database, the data is appended to the end of the file while the region header is updated to reflect the new data. If, for example, all the data you enter is centered in the Indian subcontinent, the total extent of this new data updates the region header. When India is not on the screen, the EXTENDED file is not even scanned for possible displayable data. Its region header indicates that none of its data is visible.

Region headers are created automatically, but you must be aware of them if you try to import or enter *large amounts* of data. The only way for you to start a new region header is to begin a new EXTENDED.MP3 file. In other words, rename the old EXTENDED.MP3 file to force the creation of a new EXTENDED file when you enter more data. When you finish, concatenate the two files together to get a single file with two region headers.

This only becomes an issue when your files grow so large that a linear scan begins to slow down **MAPIT**'s overall performance. Then you want to divide your large single-region file into one with several non overlapping regions.

How can you create multiple regions from a large single-region file? (See the utility MP3MOD for an alternate method to that outlined here.) Rename EXTENDED.MP3, make a backup, and then open it as **MAPIT**'s primary data file remembering the /delete switch. Be sure there is no EXTENDED file. Copy/Delete region-sized areas from the primary to the secondary file. (Copies are **always** from the primary to the EXTENDED file.) Exit **MAPIT** and rename EXTENDED.MP3 — perhaps to X1.MP3. Repeat these steps until all data has been copied and deleted from the original primary file. Finally, combine the separate files using the binary *COPY* command:

```
C:> COPY /B X*.MP3 TOTAL.MP3
```

The final result is a file with manageable regions. **MAPIT's** WORLD.MP3 database consists of a great many regions split on 5 x 5 degree tile boundaries, carefully constructed and designed to give reasonable performance in spite of the vast amounts of data involved. If you try to copy a sizable portion of this data to your private database, all copied data will be stuffed into one 'super region' in EXTENDED.MP3. Performance when accessing your private database

## **FIGEDIT**

FIGEDIT is the figures creation/editing program used to supply figures for **MAPIT**. Figures are small drawings that you can insert one or more times in your maps. The figures are stored in files ending in the extension .fig. At startup, **MAPIT** looks for the file STD.FIG and tries to resolve references to figures names from that file.

FIGEDIT is an editor of memory-resident objects. Run FIGEDIT without any command arguments. When it starts up, there are no entities in the current figure and the current figure is nameless.

FIGEDIT displays a menu along the top of screen from which you can choose items. The status bar at the bottom of the screen shows the default figure scale, entity count, figure title, and file name.

### **Menu Commands:**

#### ***FILE***

##### ***New***

Creates a new file to hold figures data.

##### ***Open***

Opens a file for input/output to FIGEDIT. The names of all figures contained in the newly opened file is listed for informational purposes. Enter return or click the left mouse button to proceed.

##### ***Close***

Saves and closes an open figures file.

##### ***Exit***

Saves and closes an open figures file if there is one and exits from the FIGEDIT.

#### ***FIGURES***

The Figures menu controls the loading and display of figure objects found within an open figures file.

## ***Redraw Full***

Redraws the figure currently loaded into memory to be redrawn at full size, the size in which editing is done.

## ***Display at Scale***

Display the current figure at the default scale displayed in the status line.

## ***New***

Begin a new in-memory figure.

## ***Load***

Load a figure from those in the current figure file. Click the left mouse button over your choice of names on the screen.

## ***Save***

Save any changes made to the current in-memory figure to the current figure file.

## ***Save as***

Save the current in-memory figure under a new name in the current figure file.

## ***Remove***

Delete a figure from the current figure file. The deleted figure is loaded into memory giving you a chance to save it in another figure file or under another name.

## ***INSERT***

One builds figures by entering objects from the INSERT menu. As construction continues, a tally of the entity count is displayed at the bottom of the screen. There is an entity count limit of 100. The more complex objects consume more entities, building to that limit.

As you insert objects, they are displayed on the screen at 100-percent size. Remember that the final display size will be much smaller. The default is 10 percent. Detail will disappear or will clutter the final figure.

## ***Line***

Insert multiple straight-line segments by clicking and dragging with the left mouse button. Double click the left button to end, the right button to back up a segment.

## ***Rectangle***

Insert rectangles by specifying the opposite diagonal corners of the rectangle you wish to enter.

## ***Arc***

Enter the center and the radius of the arc. FIGEDIT draws a complete temporary circle and asks you to enter start and stop angles. Arcs are constructed *counter-clockwise*. The start and stop angles are points relative to the circle's center (not necessarily on the circumference) anywhere on the screen. Enter the start angle first, followed by the stop angle. The circle will disappear and the arc will appear in its place.

## ***Circle***

Enter the center and the radius of the circle.

## ***Ellipse***

Insert ellipses by specifying the opposite diagonal corners of an enclosing rectangle. You can enter only horizontal or vertical ellipses.

## ***CHANGE***

### ***Origin***

Each figure has an origin. When you insert a figure into a map, you actually specify where the origin of the figure will be placed. The rest of the figure is displayed relative to the origin. FIGEDIT marks the origin with a blue cross. Its default position is the screen's center. You can change that location to any place on the screen.

### ***Scale***

Normally figures are not displayed in maps at the scale in which they were drawn, but as smaller images. The default scale is 10 percent, but you can change that to any value you want.



## **Delete**

Delete allows you to remove entities, but you must be careful. Move to the end of a line, to the radius of a circle or arc, or to the major or minor radius of an ellipse and press the left mouse button. A small square appears if you've chosen an object, and the object type currently being considered appears in the status line. After affirming the deletion, the entity remains until the next repaint. If more than one entity occupies a spot, the first entity laid down is chosen. Be aware that an arc can be selected for deletion anywhere on its radius, even on the on the clipped (non displayed) portion! Getting a deletion square in the middle of nowhere means that *something* is being deleted!

### **Tips:**

You needn't be in a figure or file to begin editing.

You can transfer figures from one file to another:

- Open a file
- Load a figure
- Save-as to another or new file.

The key to good figures is to *keep them simple*. Detail which looks good at full screen is clutter at 10 percent. (Here's where artistic talent is invaluable!)

Like .mp3 files, .fig files can be concatenated with the binary copy.

## **GEOCODE**

This **MAPIT** utility geocodes or adds latitude/longitude information to records of a dBASE IV file. An input file must be run against a key file to yield the output file. All these files must be in dBASE IV format.

### **The GEOCODE Command Line:**

Usage:

```
GEOCODE [input_file] [output_file] [/options...]
```

where

- input\_file* — an optional dBASE IV file holding the records to be geocoded. Assumes a .DBF extension.
- output\_file* — an optional dBASE IV database name in which the geocoded data will be stored. There are normally more fields in the *output\_file* than in the *input\_file*. If the */inplace* option is used, the *output\_file* must not be specified. Extensions supplied.

options:

- /inplace* — Update the *input\_dbf*. Output fields must already exist.
- /lookup\_dbf =* — a dBASE IV file holding the latitude/longitude and a key field. Assumes a .DBF extension
- /key =* — the primary key field name. *e.g.* ZIP
- /k\_length =* — the match limit on the primary key.
- /definition =* — a .DEF file holding definitions which may augment or replace those given here as command line arguments. Command line arguments take precedence over statements in the definition file.
- /append* — add newly geocoded data to the end of the *output\_file*.
- /overwrite* — delete any data in an existing *output\_file* before adding the newly geocoded data. By default, an error message is issued if the *output\_file* already exists and neither

	the /append nor the /overwrite option is given.
/matched	— transfer records with 1 or more matches in the <i>lookup_dbf</i> .
/unique	— transfer records with exactly 1 match in the <i>lookup_dbf</i> .
/first	— don't search beyond the first match in the <i>lookup_dbf</i> . If there are mutiple, the geocoded result is the average of all matches unless this option is used.
/max_range=	— include only records within this range.
/min_range=	— include only records beyond this range..
/units=	— min/max range units: miles, nmiles, kilometers, km, yards, feet, ft
/center=	— calculate the distance from this location to the geocoded distance for each record. No space allowed between latitude and longitude unless the string is delimited by quotation marks. e.g. /center=40:30.84N79:36.94W
/version	— display program's version.
/?	— display this output.

## Definition File:

**GEOCODE**'s command line arguments are so numerous that you can place some or all of them within a definition file having the extension .DEF. This file is free format on a line-by-line basis and uses the ';' character as a comment. Everything to the right of the comment character is ignored. The order of statements is not important. Certain combinations are illegal. Many of the following commands can also be entered as command line arguments. For instance, **UNIQUE**: corresponds to **/unique**. Selections made at the command line override those made in a definition file allowing you to have a general definition which you can override easily at program invocation. Reference your definition file from the command line with the /definition option as in

```
C:\> geocode /definition=my_def
```

## Specify the input file:

```
INPUT DBF: file_name  
INPUT MDX: file_name ; optional
```

INPUT TAG: tag\_name ; not supported

INPUT DBF is the database you want to geocode. In general, it makes no difference if the input file has an index (.MDX file). If you specify an INPUT MDX and an OUTPUT MDX, **GEOCODE** will try to create a similar index for the output file.

### ***Specify the lookup file:***

LOOKUP DBF: file\_name  
 LOOKUP MDX: file\_name ; defaults to LOOKUP DBF  
 LOOKUP TAG: tag\_name ; defaults to "DEF\_TAG"  
 LOOKUP EXP: ; needed if no .MDX exists for  
 dBASE\_Expression LOOKUP DBF.

The LOOKUP DBF must have an index file/key. The MDX file\_name defaults to the DBF file\_name and the tag\_name defaults to "DEF\_TAG" unless otherwise specified. If no LOOKUP MDX exists for a DBF, you must specify a dBASE\_Expression which, in general, consists of one or more field names connected with operators. You can define multiple tag/expression combinations in any one MDX but can use only one-at-a-time. This puts several indexes into one MDX file. The following, for example, is legal:

LOOKUP DBF: zipphone  
 LOOKUP MDX: zipphone  
 LOOKUP TAG: def\_tag ; 1<sup>st</sup> tag  
 LOOKUP EXP: upper(zip)  
 LOOKUP TAG: city\_state ; 2<sup>nd</sup> tag  
 LOOKUP EXP: upper(city + state\_code)  
 LOOKUP TAG: county\_city ; 3<sup>rd</sup> tag  
 LOOKUP EXP: county + city

If your LOOKUP DBF already has a defined MDX, there is no need to go to all this trouble. **GEOCODE** will select from the tags/expressions available.

### ***Specify the output file:***

OUTPUT DBF: file\_name  
 OUTPUT MDX: file\_name ; optional  
 OUTPUT TAG: tag\_name ; not supported  
 OUTPUT EXP: ; not supported.  
 dBASE\_Expression  
 INPLACE: ; updates INPUT DBF. All  
 output fields must be defined.  
 OVERWRITE OUTPUT: ; overwrite, append, or error msg.  
 APPEND OUTPUT:

LAT: lookup\_field output\_field ; latitude field names in the lookup and output files.  
LON: lookup\_field output\_field ; longitude field names in the lookup and output files.  
RECORD DISTANCE: output\_field ; save the great circle distance from CENTER.  
RECORD MATCH: output\_field ; save the match count. 0 or greater.  
MATCHED ONLY: ; matched 1 or more times  
UNIQUE: ; matched exactly once.

The OUTPUT DBF is a copy of the INPUT DBF but may contain more fields and fewer records depending on matching and selection criteria. The INPLACE option updates the original if the original has enough fields. The fields of unmatched records will not be changed if MATCHED ONLY or UNIQUE is specified. MATCHED ONLY and UNIQUE act on the number of records in the lookup database matching the input record. (See KEY below for what defines a match.) If you specify neither, all records from the input file, matched or not, will be transferred. You can specify that the number of matches be recorded in the output database with the RECORD MATCH statement. The existence of the output database will generate an error message and halt further processing unless you indicate your desire to either OVERWRITE or APPEND to the output database. The LAT and LON statements map the fields containing the latitude and longitude from the lookup database to the output database, the ultimate purpose of geocoding being to transfer this data to matching records. The RECORD DISTANCE statement saves the great circle distance from each geocoded record to the location specified in the CENTER statement (See below).

### ***Specify the selection options:***

KEY: lookup\_field input\_field [length] ; matches key fields in the lookup and input files. Multiple statements are permitted.  
FIRST MATCH: ; stop lookup for this input record after the first match.  
MATCH ONLY value IN FIELD input\_field [length] ; match a constant value.  
SKIP value IN FIELD input\_field [length] ; skip a constant value.  
CENTER: lat/lon\_string ; the location from which you want to measure distances.

MAX RANGE: distance	Expects the standard free-format lat/lon string. ; records beyond this range don't match.
MIN RANGE: distance	; records within this range don't match.
UNITS: units	; the units applied to the ranges above. These must be spelled out completely (as opposed to the command line arguments version) and include miles, nmiles, kilometers, km, yards, feet, and ft. Defaults to miles.

You must specify at least one KEY statement to correlate fields in the lookup and the input files. The optional length argument limits the test for a match to the first *n* characters of the fields. Use this, for example, when matching a mailing list containing 9 character zipcodes to a lookup database of 5 character zips by setting length to 5. You may have multiple KEY statements all of which must be satisfied for a record to be matched. The first statement, however, is the only one for which an indexed lookup is performed in the lookup database. If FIRST MATCH is not used, subsequent records are read from the lookup database until the first KEY no longer matches. Closely related to the KEY statement which compares fields in the input and lookup databases are the MATCH ONLY and the SKIP statements. These statements select or reject records based on matching a constant value against a field in the input database. You may have only one of each of these statements. An example would be to MATCH ONLY on zipcodes having the first three digits equal to "153" or all records whose state is "PA".

The CENTER statement establishes a single location from which the great circle distances to each matched record can be calculated. Use the MAX/MIN RANGE statements to further exclude otherwise-qualifying records and optionally record those distances with the RECORD DISTANCE statement discussed above. The UNITS statement indicates the units of the values in the MAX/MIN RANGE statements. If absent, the default is statute miles.

## Example Output:

The following example is created by copying the lookup database ZIPPHONE.DBF from the **MAPIT** delivery CD-ROM and found in directory \MAPIT20\DBQS. When you geocode XLIST.DBF against it using the DEMO.DEF definition file (both found in \MAPIT20\EXAMPLES\GEOCODE), **GEOCODE** will create the index file ZIPPHONE.MDX at the time of the first execution and will produce the following output if executed as follows:

```
C:> geocode /def=demo

64 records geocoded, 68 transferred out of 68.

min distance =      9.63 miles
avg distance =   435.32 miles
max distance =  2199.89 miles
from location lat = 40.5833333, lon =-79.5833333
```

The output will be saved in the new file TEMP.DBF. In this example all records, both matched and unmatched, are transferred. You can transfer only the matched records to another database by overriding the definition file specifications from the command line.

```
C:> geocode /def=demo xlist another /matched
```

This time the output will read:

```
64 records geocoded, 64 transferred out of 68.
```

and the geocoded records will be stored in the file ANOTHER.DBF. Make changes to the definition file and explore different option combinations. Some combinations may generate error messages which should be self-explanatory.

## Example Definition File:

```
; demo.def - Demonstration Geocode definition
;

;      lookup      input  [length]
KEY:   zip         ZIP    5
key:   state_code state  1    ; avoid non-US
                                ; addresses

record match:  geomatched ; record match count in this
                                ; field
```

```

;unique:                ; = 1
;matched only:         ; > 0
                        ; 0 or greater if not
                        ; specified

overwrite output:      ; Gives error msg if one of
;append output:        ; these is NOT defined and
                        ; output file exists.

LOOKUP DBF:            ZIPPHONE
lookup mdx:            zipphone ; Defaults to dbf name.
lookup tag:            def_tag  ; Defaults to def_tag.
lookup exp:            upper(zip); Needed if no mdx exists for
                        ; lookup dbf.

;lookup tag:           2nd_tag
;lookup exp:           upper(city + state_code)
;lookup tag:           third
;lookup exp:           county + city

INPUT DBF:             XLIST     ; File to be geocoded.
;input mdx:            xlist

OUTPUT DBF:            TEMP
; Inplace:

record distance:      dist
center: 40:35N 79:35w
;Max Range: 1000
;Min Range: 100
UNITS: miles

first match:

; match only PA in field STATE
;skip 15066 in field Zip 2

```

## Advanced Considerations:

The database ZIPPHONE.DBF we used in the preceding example may be good for casual geocoding but, as it stands, lacks the control and efficiency those who would do more might desire. It holds a combination of zipcode and telephone exchange information, over 105,000 records, many of which are duplicated from one point of view or the other. To make it truly useful and efficient for your operation, you will want to import ZIPPHONE into a database manager such as Microsoft's ACCESS, delete the



unnneeded fields, remove duplicate and blank key records, and save the database as a dBASE IV image under a different name.

If, for example, you are going to be working with 3 digit zipcode centroids, you could generate those centroids by removing the telephone data completely and reducing the width of the ZIP field to 3 digits. Remove the duplicate records so that each record is unique. Under any three digit zipcode there will be a number of different records. Within ACCESS, you could calculate each 3 digit combination's average lat/lon by database programming or you could export the database to serve as an initial lookup database and, still within ACCESS, create a smaller database with unique three digit zipcodes and null lat/lon fields. Export this second database as your input database and use **GEOCODE** to find the average location of each record by **not** using the /FIRST option in the geocoding process. Your new output database will represent the true centroid of 3-digit zipcodes and could be used as your new lookup table when geocoding future work. If you are geocoding a large amount of data, a condensed table holding a single record for each matching key is more efficient than averaging over multiple lookup records on the fly for each input record. In a non-condensed table, the /FIRST option will give misleading results unless all equal-key lat/lon data are duplicates.

## GEOTOENT

This **MAPIT** utility converts geocoded dBASE IV files to **MAPIT** entity files. **MAPIT** entity files are standard dBASE IV files containing certain field names required by **MAPIT** to display the entities properly. **MAPIT** entities stored in dBASE files are point entities: they exist at single points and have no inherent dimensions or extents. Lines, on the other hand, stored in .mp3 files, have length and consist of many points. The point entities are figures, stroked text, hidden text, cities, and pcx maps.

### The **GEOTOENT** Command Line:

Usage:

```
GEOTOENT [input_file] [output_file] [/options...]
```

where

- `input_file` — an optional dBASE IV file holding the records to be converted. Assumes a .DBF extension.
- `output_file` — an optional dBASE IV database name in which the **MAPIT** entity data will be stored. Assumes a .DBF extension.

options:

- `/definition=` — a .DEF file holding definitions which may augment or replace those given here as command line arguments. Command line arguments take precedence over statements in the definition file.
- `/append` — add newly converted data to the end of the `output_file`.
- `/overwrite` — delete any data in an existing `output_file` before adding the newly converted data. By default, an error message is issued if the `output_file` already exists and neither the `/append` nor the `/overwrite` option is given.
- `/type=` — create this type of **MAPIT** entity. Use `figure`, `stroked_text`, `hidden_text`, `city`, or `pcx_map`.
- `/help` — display a list of legal definition file statements.
- `/version` — display program's version.

/? — display this output.

## Definition File:

**GEOTOENT**'s command line arguments are so numerous and complex that you can place some or all of them within a definition file having the extension `.DEF`. This file is free format on a line-by-line basis and uses the `';` character as a comment. Everything to the right of the comment character is ignored. The order of statements is not important. Certain combinations are illegal. Some of the following commands can also be entered as command line arguments. For instance, **APPEND OUTPUT:** corresponds to `/append`. Selections made at the command line override those made in a definition file allowing you to have a general definition which you can override easily at program invocation. Reference your definition file from the command line with the `/definition` option as in

```
C:\> geotoent /definition=my_def
```

## Specify the input file:

INPUT DBF: `file_name`

INPUT DBF is the database you want to convert. In general, it makes no difference if the input file has an index (`.MDX` file). The input file must be a geocoded file, *i.e.* it must contain fields containing latitude and longitude data.

## Specify the output file:

OUTPUT DBF: `file_name`

OVERWRITE OUTPUT: ; overwrite, append, or error msg.

APPEND OUTPUT:

ENTITY TYPE: *type* ; create the output dbf in the specified format choosing from *figure, stroked\_text, hidden\_text, city, or pcx\_map*. Defaults to *figure* if not specified.

The OUTPUT DBF is a copy of the INPUT DBF augmented to contain those fields necessary to define the selected ENTITY TYPE. The existence of the output database will generate an error message and halt further processing unless you indicate your desire to either OVERWRITE or to APPEND to the output database.

**Modify fields:**

EXCLUDE field	; Don't transfer this field.
RENAME field1 TO field2	; Change the field's name.
field = fields strings...	; Load an output field with data from multiple input fields and strings. Input field: <i>n</i> limits assignment to the first <i>n</i> characters.
LOAD field WITH constant	; Load a field with a constant.
LOAD field1 WITH constant IF field2 OP constant	; Load a field with a constant if another field matches certain criteria.
LOAD field1 WITH constant IF field2 OP constant WIDTH number	; Load a field with a constant if the first <i>n</i> characters of field2 matches certain criteria.
LOAD field1 WITH constant IF field2 OP constant1 AND OP constant2	; Add a second operator and constant.
LOAD field1 WITH constant IF field2 OP const1 AND OP const2 WIDTH number	; Two operators with constants and a WIDTH applied to field2.

EXCLUDE excludes input fields from being replicated in the output database. RENAME changes the name of an input field in the output database. The assignment character = specifies that the fields and strings on the right should be concatenated and saved in the field name on the left. Strings are characters between quotation marks and may include blanks. Specify a newline character with \n which is converted to the carriage return-line feed sequence for proper display in **MAPIT** and in dBASE programs. The contents of fields are transferred by naming the field. Trailing blanks are eliminated. You can truncate a field by appending the : character followed by a maximum width. For example:

```
hiddentext = "Office location:\n" city " , " State " " ZIP:5
```

LOAD loads a field with constant text. The text may not have imbedded blanks. Copying the data can be conditional upon the value of a second field's meeting certain criteria specified by one of the following operators: =, <, <=, >, >=, <>. For example:

```
LOAD labelcolor WITH 4 IF zip >= 15000 AND < 16000 WIDTH 5
```

Comparisons of numeric data are done by first converting the fields to floating point representation. Character fields are compared lexically. The preceding example is somewhat confusing because it is a lexical (character-by-character) comparison of what might appear to be numeric data but isn't. **GEOTOENT** issues a warning message when numeric fields are truncated by the WIDTH operator. You may not get what you expect depending upon how the numeric data is internally held in the field.

### Example Output:

**GEOTOENT** will produce the following output when run against the TEMP.DBF data file and TEST4.DEF definition file found on the **MAPIT** CD-ROM under the \MAPIT20\EXAMPLES\GEOTOENT directory

```
C:> geotoent /def=test4

Creating Figures Output.
Excluding field GEOMATCHED
Renaming field DIST to DISTANCE

Creating database index "OUT"

Transferring data...

Re-indexing output database.
68 records transferred.
```

This HP-GL/2 plot was created by running MYMAP.BAT (See **MAPIT BATCH FILES**) from a directory on the C: drive with local scantable C:SCANTABL altered to include an Active record referencing figure entity (E\_TYPE "F") database C:OUT.DBF. Note that the geocoded circles are proportional in size to their distance from Pittsburgh and are colored in the **MAPIT** original based on distance. The figure labels contain text from the distance field.

### Example Definition File:

```
; test4.def - Example GEOTOENT definition file.

INPUT DBF: temp
OUTPUT DBF: out

OVERWRITE OUTPUT:
```

```

ENTITY TYPE: figure

EXCLUDE geomatched
RENAME dist to distance

label_text = dist

LOAD fig_scale WITH 1 IF dist <= 10
LOAD fig_scale WITH 2 IF dist > 10 AND >= 50
LOAD fig_scale WITH 3 IF dist > 50 AND >= 100
LOAD fig_scale WITH 4 IF dist > 100 AND >= 200
LOAD fig_scale WITH 5 IF dist > 200 AND >= 300
LOAD fig_scale WITH 6 IF dist > 300 AND >= 500
LOAD fig_scale WITH 7 IF dist > 500 AND >= 800
LOAD fig_scale WITH 8 IF dist > 800 AND >= 1200
LOAD fig_scale WITH 9 IF dist > 1200 AND >= 2000
LOAD fig_scale WITH 10 IF dist > 2000

LOAD labelcolor WITH 3 IF dist > 200
LOAD labelcolor WITH 5 IF dist > 500

LOAD labelcolor WITH 4 IF dist >
  2000

;LOAD fig_name WITH CIRCLE IF zip = 15 WIDTH 2
LOAD fig_name WITH CIRCLE

HIDDENTEXT = CITY "\n"STATE ", " ZIP:5

```

## Advanced Considerations:

The primary purpose of **GEOENTOENT** is to add and initialize fields from an existing dBASE file and produce a new dbf in one of **MAPIT**'s entity formats. When you reference this entity dbf by adding it to the scan table dbf (see FILES/Add Scan Table Record or FILES/Edit DBF), **MAPIT** will display your data the next time it displays data (scans the databases) via the ZOOM/Redisplay or ZOOM/Overwrite command. You could, by hand with a database manager such as ACCESS, do the very things this program does. The LOAD, RENAME, EXCLUDE, and assign functions could all be accomplished in your database manager. However the advantages of **GEOENTOENT** are operational ease and efficiency. If you have complicated functions to perform, you may have to fall back on ACCESS or its equivalent. But **GEOENTOENT** will place you far along the path toward accomplishing your goal.

You may have to experiment to work out the details of complicated queries. **GEOTOENT** issues warning and error messages at run time if you make mistakes and may refuse to transfer data. A few general rules will help. Field names referenced as the source of data refer to fields in the input file. Although a field may be EXCLUDED (not transferred) or RENAMED, its data is available for use in LOAD or assignment statements at run time. Destination fields, however, must exist in the output dbf.

You can truncate data in a field by assigning it to itself with a width parameter. The data is truncated although the field is not shortened.

```
zip = zip:5
```

**MAPIT** entities require the existence of the fields LATITUDE and LONGITUDE. If your source dbf contained the data stored in the fields LAT and LON, you would fulfill **MAPIT**'s requirements by renaming the fields.

```
RENAME lat TO latitude  
RENAME lon TO longitude
```

## **GEOTOMP1**

This **MAPIT** utility converts geocoded dBASE IV files to **MAPIT** .mp1 files, ASCII files containing multiple lines of latitude/longitude pairs. The **MAPIT** utility MP1TOMP3, in turn, transforms these latitude/longitude pairs into **MAPIT** .mp3 line entities. Depending upon your request, the lines can describe circles, squares, and rectangles around points in your original dBASE file, connect points with straight lines and great circle arcs, and connect other points to the original points by straight lines and great circles.

### **The GEOTOMP1 Command Line:**

Usage:

```
GEOTOMP1 [input_file] [output_file] [/options...]
```

where

- `input_file` — an optional dBASE IV file holding the records to be converted. Assumes a .DBF extension.
- `output_file` — an optional file name in which **MAPIT** line data will be stored as ASCII lat/lon pairs. Assumes a .MP1 extension.

options:

- `/definition=` — a .DEF file holding definitions which may augment those given here as command line arguments. Command line arguments take precedence over statements in the definition file.
- `/append` — add newly converted data to the end of the *output\_file*.
- `/overwrite` — delete any data in an existing *output\_file* before adding the newly converted data. By default, an error message is issued if the *output\_file* already exists and neither the `/append` nor the `/overwrite` option is given.
- `/ location=` — the external location from which to draw straight lines or great circles to individual records. Expects the standard free-format lat/lon string. No space allowed between



- latitude and longitude unless the string is delimited by quotation marks. e.g.  
/center=40:30.84N79:36.94W
- /help — display a list of legal definition file statements.
- /version — display program's version.
- /? — display this output.

## Definition File:

**GEOTOMP1**'s command line arguments are so numerous and complex that you can place some or all of them within a definition file having the extension .DEF. This file is free format on a line-by-line basis and uses the ';' character as a comment. Everything to the right of the comment character is ignored. The order of statements is not important. Certain combinations are illegal. Some of the following commands can also be entered as command line arguments. For instance, **APPEND OUTPUT** corresponds to **/append**. Selections made at the command line override those made in a definition file allowing you to have a general definition which you can override easily at program invocation. Reference your definition file from the command line with the **/definition** option as in

```
C:\> GEOTOMP1 /definition=my_def
```

## Specify the input file:

INPUT DBF: file\_name

INPUT DBF is the database from which you want to generate lines. In general, it makes no difference if the input file has an index (.MDX file). The input file must be a geocoded file, *i.e.* it must contain fields containing latitude and longitude data.

## Specify the output file:

OUTPUT FILE: file\_name

OVERWRITE OUTPUT ; overwrite, append, or error msg.  
APPEND OUTPUT

The OUTPUT FILE contains a series of line definitions generated from specifications in the DEFINITION FILE. If an OUTPUT FILE of the same name already exists, you can choose to OVERWRITE OUTPUT, which deletes it, APPEND OUTPUT, which adds the new data to its end, or have processing cease with an error message, the default action, which protects the original file contents.

**Specify fields and locations:**

LATITUDE: field_name	; Name of the field holding the record's latitude. Defaults to "LATITUDE".
LONGITUDE: field_name	; Name of the field holding the record's longitude. Defaults to "LONGITUDE".
INTERNAL LATITUDE: field_name	; Field containing the latitude of an internal location.
INTERNAL LONGITUDE: field_name	; Field containing the longitude of an internal location.
EXTERNAL LOCATION: lat/lon_string	; An external loc. as a standard free-format lat/lon string. May contain a blank between lat lon. <i>e.g.</i> 40:30.84N 79:36.94W

LATITUDE: and LONGITUDE: specify the field names holding the geocoded latitude and longitude data for each record. Their default values are "LATITUDE" and "LONGITUDE" respectively. INTERNAL LATITUDE: and INTERNAL LONGITUDE: specify the field names of internal locations for each record. The internal location can be used for drawing a line to the record's geocoded location. Similarly the EXTERNAL LOCATION: specifies a single location from which lines can be drawn to each record's geocoded location. You can specify or override an EXTERNAL LOCATION: statement with the **/location=** command line argument.

**Specify marker instructions:**

COMMENT: fields_names and quoted_strings	; Precede a record in the OUTPUT FILE with a comment consisting of quoted strings and data from the named fields.
CIRCLE radius_value [if_clause]	; Output a CIRCLE of a given <b>radius</b> if certain optional conditions are met.
SQUARE width_value [if_clause]	; Output a SQUARE of a given <b>width</b> if certain optional conditions are met.
RECTANGLE WIDTH width_	; Output a RECTANGLE of a

<i>value</i> [ <i>if_clause</i> ]	given <b>width</b> if certain optional conditions are met.
RECTANGLE HEIGHT <i>height_value</i> [ <i>if_clause</i> ]	; Output a SQUARE of a given <b>height</b> if certain optional conditions are met.
PLUS <i>width_value</i> [ <i>if_clause</i> ]	; Output a PLUS '+' character of a given <b>width</b> if certain optional conditions are met.
X <i>width_value</i> [ <i>if_clause</i> ]	; Output an 'X' marker of a given <b>width</b> if certain optional conditions are met.

COMMENT: specifies that the following fields and strings be concatenated and written as a comment to the OUTPUT FILE. Comments in .mp1 files are any text following the ';' character through the end of the line. Strings are characters between quotation marks and may include blanks and \n newline characters which continue the comment to the next line.. You can truncate a field by appending the ':' character followed by a maximum width. For example:

COMMENT: CITY " , " STATE " " ZIP\_CODE:5 " , \nUnited States"

The CIRCLE and other marker statements cause those markers to be written to the OUTPUT FILE as sequences of straight line segments described by latitude/longitude pairs. The size of those markers is defined by a value followed by an optional unit. Note that the circle is defined in terms of its RADIUS while the other markers, by their WIDTH or HEIGHT dimensions. Both a RECTANGLE WIDTH and a RECTANGLE HEIGHT statements are needed to define each RECTANGLE.

### ***Specify line instructions:***

STRAIGHT LINE INTERNAL [ <i>if_clause</i> ]	; Draw a straight line between a location specified within a record and the record's geocoded location.
STRAIGHT LINE EXTERNAL [ <i>if_clause</i> ]	; Draw straight lines between a single location specified outside a record and all records' geocoded locations.
STRAIGHT LINE CONNECTOR [ <i>if_clause</i> ]	; Draw a straight line between each record's geocoded location and that of the following record.
GREAT CIRCLE INTERNAL	; Draw a great circle between a

[if_clause]	location specified within a record and the record's geocoded location.
GREAT CIRCLE EXTERNAL [if_clause]	; Draw great circles between a single location specified outside a record and all records' geocoded locations.
GREAT CIRCLE CONNECTOR [if_clause]	; Draw a great circle between each record's geocoded location and that of the following record.

These optionally conditional expressions draw either straight lines or great circles connecting the geocoded locations of records, connecting an internal location within each record to its geocoded location, or connecting a single externally-specified location to the geocoded location of each record.

**Subdefinitions:**

value	numeric_field_name, number
unit	miles, nmiles, kilometers, km, meters, yards, feet, ft
if_clause	IF field_name OP constant [AND OP constant] [WIDTH number]
OP	=, <, <=, >, >=, <>

Outputting data can be dependent on an optional *if\_clause* and the value in a field's meeting certain criteria as specified above. For example:

```
CIRCLE 5 miles IF zip >= 15000 AND < 16000 WIDTH 5
```

Comparisons of numeric data are done by first converting the fields to floating point representation. Character fields are compared lexically. The preceding example is somewhat confusing because it is a lexical (character-by-character) comparison of what might appear to be numeric data but isn't.

**GEOTOMP1** issues a warning message when numeric fields are truncated by the WIDTH operator. You may not get what you expect depending upon how the numeric data is internally held in the field.

**Example Output:**

**GEOTOMP1** will product the following output when run against the CAPITALS.DBF data file and TEST1.DEF definition file found on the **MAPIT** CD-ROM under the \MAPIT20\EXAMPLES\GEOTOMP1 directory. Output to the screen:

```
C:> geotomp1 /def=test1

Overwriting
Transferring data...
412 records processed out of 1071.

C:> mp1tomp3 out /color=4 /over
C:> mymap out
```

**Test1 Output:**  
**Great Circles from Washington DC to National Capitals**  
**Straight Lines from National Capitals to State Capitals**  
**Squares and Circles around National and State Capitals**  
**Test1 Output: Detail of Brazil**  
**Test2 Output of Cards:**  
**Straight Line Connectors between Records**  
**25 mile Squares**  
**Test3 Output of Cards:**  
**Great Circles from External Point to Nodes**  
**25 mile Squares**

Test1 output to the file OUT.MP1:

```
; Washington, District of Columbia, United
States of America
39.6191469 -76.0965859
39.6191469 -77.9767541
38.1708531 -77.9577757
38.1708531 -76.1155643
39.6191469 -76.0965859

38.8950000 -77.0366700
38.8950000 -77.0366700

; Taipei, , Taiwan
25.7680369 122.3013261
25.7680369 120.6931139
24.3197431 120.7025557
24.3197431 122.2918843
25.7680369 122.3013261

25.4059635 121.4972200
.
25.4059635 121.4972200

38.8950000 -77.0366700
```

```
25.0438900 121.4972200
```

```
25.0438900 121.4972200
```

```
25.0438900 121.4972200
```

```
; Seoul, Soul-t'ukpyolsi, Korea, Republic of
```

```
38.2872069 127.8995204
```

```
38.2872069 126.0543596
```

```
36.8389131 126.0721232
```

```
36.8389131 127.8817568
```

```
38.2872069 127.8995204
```

*etc.*

## Example Definition File:

```
; test1.def - examples of geotomp1 options.
```

```
; Draw Great Circles from Washington DC to national capitals.
```

```
; Draw straight lines from national capitals to state capitals.
```

```
; Draw 100 mile width squares around national capitals.
```

```
; Draw 25 mile radius circles around state capitals.
```

```
input dbf: capitals
```

```
output file: out
```

```
overwrite output
```

```
Square 100 IF natnl_cap = T
```

```
CIRCLE 25 miles IF state_cap = t
```

```
GREAT CIRCLE EXTERNAL IF natnl_cap = t
```

```
STRAIGHT LINE INTERNAL IF state_cap = t
```

```
EXTERNAL LOCATION: 38.895 -77.03667 ; Loc of Washington, DC
```

```
INTERNAL LATITUDE: lat ; Natl capital loc assigned beforehand
```

```
INTERNAL LONGITUDE: lon ; in database.
```

```
COMMENT: city_name " , "PROVINCE " , " country
```

## Advanced Considerations:

**GEOTOMP1** differs from **GEOTOENT** in that the former converts discrete points recorded in dBASE files to series of line data depicting markers around said points or the connection of these points with other locations. **GEOTOENT** converts dBASE data to **MAPIT** dbf entities. **MAPIT** dbf entities can, in general, be either map or screen relative. .MP1 line entities are of fixed size and are,

therefore, always map relative. You must specify before hand the radius of a circle in miles or kilometers, and it will always be such.

**GEOTOMP1** allows the user to attach multiple lines and markers to each point in a single pass. You could draw a circle and square around each point and draw a straight lines and great circles from internal or external points while connecting consecutive data points with straight lines and great circles. There is, however, beauty in simplicity.

---

## **GPS**

Allison Software's **MAPIT** GPS Utilities Package is a set of IBM PC-compatible programs designed to capture NMEA 0183 data from a Global Positioning System receiver transforming it into mapping coordinate data and saving it to computer disk for display and use by **MAPIT**. The Global Positioning System (GPS) consists of 24 satellites maintained by the US Department of Defense. Signals transmitted from these satellites are encoded with precise location and timing information which enable users with GPS receivers to determine their location anywhere in the world with 100 meter accuracy.

### **GPS Overview:**

**MAPIT** GPS supports the capture of real-time positioning data from a GPS receiver's NMEA output cable through a computer's serial COM port. GPS converts the data from its original NMEA 0183 sentence format to a series of latitude/longitude/time fixes which the user can view in real-time on the screen, can be sent to an intermediate dB file for concurrent access by **MAPIT**, and saved in a log file. GPS is a Windows program and must be installed under Microsoft Windows 3.1 or Windows 95 on an IBM PC-compatible computer. It expects GPS output in the form of NMEA 0183 Version 2 sentences supplied at 4800 baud to serial COM port 1 - 4. (See **Hardware Connections** for port pin configuration.)

GPS notifies the user when the data signal is acquired, dropped, or missing; optionally logs incoming data to the display or to an ASCII log file; and parses NMEA sentences collecting the results as fixes for display, logging, and updating of the first record of a dB IV-compliant data base for real-time access by **MAPIT**. Additionally, supported and unsupported NMEA sentences can be displayed and logged and the computer's system clock reset to agree with the time supplied by the GPS signal. GPS updates the file system's FAT (File Access Table) about once a minute to reduce data loss due to computer battery failure or system crash.

GPS continues to operate when minimized to an icon.



## GPS Installation:

Being a Windows program, you must install GPS from within Windows so that its icon appears in the program group of your choice.

From within Windows 3.1 first select the program group you want the GPS icon to reside in by clicking anywhere in it. When selected, the program group's top banner is darkened. From the Program Manager, click on File and then select New from the menu. The New Program Object window appears. Select Program Item and OK.

The Program Item Properties dialog box appears. Enter "MAPIT GPS" as the Description, "gps" as the Command Line, and the path where GPS.EXE resides as the Working Directory. (Note CB5.DLL must also reside in the working directory with GPS.EXE or in the \WINDOWS\SYSTEM subdirectory.) Click OK and GPS's icon will be installed in your chosen program group. Execute GPS by double clicking this icon.

## GPS's Menus:

File	Accesses files containing settings.
DBF	Accesses database files.
Log	Accesses log files.
Transfers	Controls GPS input and logs output information.
Settings	Controls data port, fix spacing, and smoothing.
Help	Displays About screen.

### **File:**

The files accessed by this menu record or restore the GPS DBF, Log, Transfers, and Settings options. Opening an existing file recreates the state GPS was in when the File was created. Files have the default extension .GPS.

**New** - Resets the program settings state back to conditions existing at program start.

**Open** - Opens an existing program settings state file and loads the settings into the program, closes any old DBF or Log files, and opens requested ones.

**Save** - Saves the program settings state in an existing file.

**Save As** - Saves the program settings state in a new file.

**Exit** - Terminates the GPS program and closes all files.

### ***DBF:***

The first record of the database file opened by this menu is kept up-to-date with the current fix generated by GPS when the system is active. Files have the default extension .DBF. The name of an active dB file is listed in the menu bar.

**Open** - Opens an existing DBF for continual update with the current fix.

**Save As** - Creates a new DBF and opens it for continual update with the current fix.

**Close** - Closes the currently open DBF.

### ***Log:***

The files accessed by this menu hold information displayed on the main window during program execution. Files have the default extension .LOG. The name of an active Log file is listed in the menu bar.

**Open** - Opens an existing Log for output of newly displayed information. The information is appended at the file's end.

**Save As** - Creates a new Log and opens it for output of newly displayed information. If you force output to an existing file, the file is first truncated

**Close** - Closes the currently open Log.

### ***Transfers:***

Controls input from the GPS unit, the display of information on the screen (and to the Log file when one is open), and the synchronization of the computer's clock with the time encoded in the received signal.

**Receive GPS** - Accepts GPS input data from the selected serial com port when checked. Logs an error message to the display (and optional Log file) if data stops for a ten second period. (Expected nominal data rate is one fix consisting of one or more NMEA sentences no more than once a second.)

**GPS input** - Outputs raw data to the display (and optional Log file) when checked. The data is compatible with .NME format and can therefore be used as input to NMETOMP1.

**Position output** - Outputs processed fix data to the display (and optional Log file) when checked. A fix is output after the first sentence of the next fix is received. Fix output is in a format compatible with .MP1 format and therefore can be used as input to MP1TOMP3.

**Supported** - Outputs supported (recognized and therefore translated) raw NMEA 0183 data sentences to the display (and optional Log file) when checked.

**Unsupported** - Outputs unsupported (unrecognized and therefore untranslated) raw NMEA 0183 data sentences to the display (and optional Log file) when checked.

**Errors** - Outputs the raw data of any recognized sentences which are obviously in error to the display (and optional Log file) when checked. This condition most commonly happens at program startup when only the last part of the first sentence is received.

**Reset Clock** - When checked, updates the computer's system clock to reflect the minutes and seconds as they're delivered real-time from the GPS hardware. The hour is not changed in order to protect the local time offset from the Universal Time delivered by the GPS except when the corrected minutes advance or retreat over the hour mark. Even that exception is excepted at the start of a new day. (Time corrections are not applied near midnight to avoid the necessity of changing the computer's calendar day and possibly year.)

Time corrections are applied only when the computer's minutes and seconds are within 10 minutes of the GPS's UT. A displayed message indicates the difference between these two times and whether the computer's clock has been reset. The user must reset the computer manually with the TIME command to the approximate time if the time difference is too great for automatic update. Apparent random delays on some GPS units between receiving a signal, processing it, and downloading it through the NMEA port give rise to reset times which can fluctuate from reset to reset by as much as several seconds making a precisely set system clock impossible to achieve. When Reset Clock is checked, the reset is applied once but only after a new fix appears within two seconds of the last one, an attempt to insure that only current readings are used. By clicking this option off and on, you can force another reset of the time. Subsequent resets will display a time difference demonstrating the limits imposed by the fluctuations of your system's aforementioned delay.

---

## **Settings:**

Controls COM port, pruning, and averaging options.

**COM Port** - Selects one of the computer's serial COM ports at which to expect data. A newly-chosen port doesn't become effective until the next time Receive GPS is set under Transfers. (*i.e.* Already in-progress communications are not terminated by choosing a new port.) A message indicates if the chosen port is already allocated or isn't supported by the underlying hardware. Data is fully buffered and should not suffer overrun under normal program execution.

**Prune** - Indicates whether duplicate fixes are output and exactly what a duplicate fix is. By default all fixes differing in time are output independent of their spatial displacements from each other. By checking the Enable Pruning box, you invoke GPS's simple pruning algorithm which enforces a minimum separation between adjacent fixes for them to be considered non-duplicates. The Minimum Point Spacing controls the value of this minimum separation. The default 0 value implies no effective pruning. By increasing the minimum separation to a positive value, you change the definition of what a duplicate point is and therefore how close together fixes can lie and yet be displayed. There are a variety of units you can choose ranging from feet to meters to nautical miles. You must choose OK before any changes become permanent and therefore effective.

To illustrate fix pruning, set your GPS receiver in a stationary location and Enable Pruning with a 0 Minimum Point Spacing. Because of the Selective Availability randomization applied by Department of Defense to the GPS signal, a number of slightly different fixes will be displayed. (Turn on Position output to monitor the output.) Selecting a Minimum Point Spacing of 35 feet will drastically reduce the number of fixes generated in any given time period. The DOD claims that GPS fixes will fall within 100 meters of the actual location 95% of the time.

**Average** - Averages the last GPS location with the current GPS location to produce the current fix when this option is checked. This produces a somewhat smoother track when not pruning but when examining the track near the limits of its accuracy.

## **Help:**

Displays About Gps.

**About Gps...** - Displays program version number, copyright, and author information.

## Sample Output:

This sample session includes

Log/Save as	opening a log file
Transfers/GPS input	enables display of raw GPS data
Transfers/Position output	enables display of calculated positions
Transfers/Receive GPS	turns on input data from selected COM port

At this point NMEA sentences from the GPS are displayed on the screen as they are received. The first in a series of five output by this unit (a Magellan Meridian) is \$GPGLL. Note that after the next fix is begun (marked by the second \$GPGLL), the first fix is output. After several fixes, the session was terminated.

Transfers/Receive GPS	toggles off input
Log/Close	closes the log file

A more typical example might include only logging locations to a file. In this case the signal was weak and transmission stopped for a minute or two before resuming. (This data was taken at a stationary location but with a loose antenna connection resulting in widely-fluctuating fixes and high indicated speeds.)

See \MAPIT20\EXAMPLES\GPS on your CD-ROM for other data examples and NMETOMP1 below for examples of data reduction.

## dB File Format:

The dB file created or opened by GPS must contain the following numeric fields. Lengths are those assigned during a GPS create.

Name	Type	Example	Meaning
NAME	C 20	"GPS"	Name record 1 "GPS".
LATITUDE	N 10.6	-90.000000	latitude, decimal degrees, neg. = south
LONGITUDE	N 11.6	-180.000000	longitude, decimal degrees, neg. = west

ALTITUDE	N 6.0	999999	altitude in meters
TIME	N 11	Any positive long integer.	Date/Time of fix in seconds since Jan 1, 1970. (UTC)
SPEED	N 5.1	999.0	kilometers/hour
HEADING	N 5.1	359.0	degrees true from north
COLOR	N 6.0	-1	-1 = none specified, pos = color number
ACTIVE	L 1	"T"	<b>MAPIT</b> ignores if not active.

## Error:

GPS generates errors of two types: those arising from incorrect or unexpected data in otherwise proper NMEA 0183 sentences and those caused by data overruns in your PC's com port. The latter, if they happen at all, will be flagged by the message

Read error  $n$  on Com Port  $m$ .

where

$n$  indicates an error number and  $m$  a port number (1 - 4)

1	Receiving queue overflow. No room in the input queue.
2	A character was not read from the hardware before the next character arrived and was, therefore, lost.

At this writing, only error  $n = 2$  has been noted under Windows 3.1, and then only on slower machines. Reducing the size of the GPS window or reducing it to an icon will reduce the amount of material to be redisplayed for each new position and will positively impact portables with slow screen I/O.

## Precision versus Accuracy:

The old conundrum of precision versus accuracy raises its confusing head again in the case of **MAPIT**-displayed GPS data. **MAPIT** is able to store points within approximately 100 feet of a location. In other words, a point whose location should be at  $x$  may be as much as 100 feet from  $x$  in any direction. GPS has the inherent capability of locating points within approximately 50 feet of their actual position. In deference to national security, however, the Department of Defense applies a randomizing signal called Selective Availability (SA) to the GPS system which decreases its accuracy to within plus or minus 300 feet 95% of the time. On the other hand, GPS signals, because of their precision,

appear to be very accurate. The Meridian's GPGLL latitude/longitude output with 0.001 minute precision incorrectly implies fixes accurate to six feet! Plotting this data on the **MAPIT** world of 100 foot squares is like trying to view super VGA data on an old CGA display. But one is quickly reminded that the data's apparent 6 foot accuracy is meaningless when the original reading is superimposed on a second taken some time later. The two readings from the same location may be as much as 600 feet apart! It is important to view all GPS data with a 300 foot halo of uncertainty around it.

There are, however, methods of obtaining more accurate GPS fixes. In what might appear to be an internal turf war, the US Coast Guard is establishing a series of transmitter sites along the coast broadcasting correction signals which, when applied to differential GPS receivers, more than compensate for the DOD induced SA, a great boon to boat owners! Similarly, surveying GPS units use locally generated correction signals transmitted from a known point to achieve truly remarkable accuracy within a limited area.

### Hardware Connections:

Serial input to your computer is commonly through a 9 pin male connector with the following pin connections:

pin	I/O		
1	I	DCD	Data Carrier Detect
2	I	SIN	Serial Input
3	O	SOUT	Serial Output
4	O	DTR	Data Terminal Ready
5	N/A	GND	Signal Ground
6	I	DSR	Data Set Ready
7	O	RTS	Request to Send
8	I	CTS	Clear to Send
9	I	RI	Ring Indicator

If your GPS's NMEA 0183 cable does not come supplied with a connector, you can wire your own 9 pin female connector.

NMEA +	2	
NMEA -	5	
7	8	RTS - CTS (optional)
6	4	DSR - DTR (optional)

CTS and DTR needed to be

asserted for your computer to accept data in an earlier non-Windows program. The jumpered pins indicated below aren't needed for the Windows GPS.)

## Supported NMEA 0183 Version 2.00 Sentences:

GPS currently supports the following NMEA sentences:

GLL	*	Geographic Position - Latitude/Longitude <i>latitude, longitude, time</i>
GGA	*	Global Positioning System Fix Data <i>latitude, longitude, time, number of satellites in use, horizontal dilution of precision, antenna altitude</i>
VTG		Track Made Good and Ground Speed <i>track made good, speed over ground</i>
RMA		Recommended Minimum Specific Loran-C Data <i>latitude, longitude, track made good, speed over ground</i>
RMC	*	Recommended Minimum Specific GPS/TRANSIT Data <i>latitude, longitude, time, date, track made good, speed over ground</i>

\*Sufficient by itself to provide a minimum fix of *latitude, longitude, and time*.

## Problems and Solutions:

*I'm not getting any output to GPS.DBF.*

Is the right Settings/COM Port turned on? Is Transfers/Receive GPS enabled? Have you created (by DBF/Save As) and DBF/Opened GPS.DBF? Is your GPS unit turned on (The NMEA submenu is deeply buried in the Aux Setup of my Meridian.) and plugged in (☺) to the right COM port? Is the GPS currently receiving data? On the **MAPIT** side, is Tools/GPS Markers/Active enabled? Is your Interface GPS GPS.DBF in the same directory as the GPS's?



## **MP1TOMP3**

This **MAPIT** utility converts text files containing strings of latitude/longitude pairs (as might be generated from a CAD system or digitizer) to .MP3 file format. .MP3 files are about 30 percent of the size (in bytes) of their .MP1 counterparts.

Usage:

```
MP1TOMP3 data_file [/options]
```

where

*data\_file* — name of an .MP1 file to be converted to .MP3 format. An .MP3 file with the same root name is created.

options:

*/id=* — Region id stored in the region header; defaults to 0 (range: 0–65,533)

*/layer=* — Layer assignment stored in entity header; defaults to layer 0 (range: 0–65,533)

*/color=* Force the line to color *n* rather than to the layer color. Defaults to -1 (layer color).

*/minzoom=* — Min zoom factor stored in entity header; defaults to 0 (range: 0–65,533)

*/maxzoom=* — Max zoom factor stored in entity header; defaults to 0 (range: 0–65,533)

The .MP1 file format is a simple ASCII listing of latitude/longitude pairs representing strings of connected points. String termination is marked by separator (blank) records. See APPENDIX E — Standard Lat/Lon Notation. “;” is a comment character which instructs MP1TOMP3 to ignore the rest of the line. A comment in the first column of a line means to ignore the line and **not** treat it as a blank to start a new line segment.

An example of the .MP1 data format is found in the ASCII file TRACK.MP1.

48.944667	2.271500	Flying west of Paris by J-F MARTIN
48.945334	2.272000	using a GARMIN GPS
48.946000	2.272500	Course 023.5, speed 92.8 mph
48.946667	2.272833	
48.947334	2.273333	
48.948000	2.273833	
48.948834	2.274167	
48.949500	2.274667	
48.950167	2.275167	
48.950834	2.275500	
48.951500	2.276000	
*48.952167	2.276500	
48.952834	2.276833	
48.953500	2.277333	
48.980667	2.295167	
48.981334	2.295667	
48.982000	2.296000	
48.982667	2.296500	
48.983500	2.297000	

## **NMETOMP1**

NMETOMP1 is the part of the **MAPIT** GPS Utility Package designed to convert and process raw NMEA 0183 Version 2 sentence data logged on disk by the GPS program and save it to an ASCII lat/long .MP1 format file.

NMETOMP1 is included because it can do the more computationally-intensive straight line corridor pruning which GPS is not able to do on the fly. Beyond straight format conversion, NMETOMP1 also provides a great deal of error checking, the coalescing of data partially and multiply defined by a set of NMEA sentences generated from a single fix, the dropping of duplicate points, simple data smoothing, and the pruning of unneeded points.

### **The NMETOMP1 Command Line:**

Usage:

```
nmetomp1 input_file [output_file] [/options]
```

where

input_file	The required input data file name. The input file must be an ASCII NMEA sentence file whose extension, unless otherwise specified, defaults to ".NME".
output_file	An optional name for the ASCII .MP1 formatted output file. If no extension is specified, the default extension ".MP1" is used. If no output_file is specified, the input_file name with the extension ".MP1" is used for output.

options

/average	Apply a simple smoothing algorithm to the data.
/avg	Averaging is accomplished after duplicate points are removed and minimum spacing is determined. /average has no noticeable effect on the display of data by <b>MAPIT</b> unless the /minxx=spacing option is applied with a spacing value greater than the 100 foot resolution of <b>MAPIT</b> 's ability to reproduce data. See Precision versus Accuracy below for a related discussion.

---

/decimals=n	<p>The required number of places of precision after the decimal point for latitude, longitude, and time input data. Defaults to 2. A value of 0 requires none but allows data after the decimal point. The primary purpose of this option is to allow for the further checking for dropped data to the right of the decimal which are not spelled out in the NMEA specification. Because the exact precision of your data is GPS vendor-dependent, you may want to examine the raw data and adjust this option accordingly. As an example, specifying /decimals=1 will cause the otherwise correctly formatted time field "120156" in vendor X's GLL statement to be logged as an error but is not an overall debilitating error if the time is available from another sentence. Some compromise toward less stringent error checking may be necessary. (This is a fine-tuning option. If it works, don't fool unless the output shows too many bad points.) See /error below.</p>
/duplicates	<p>Don't drop duplicate points but convert all data. By default, NMETOMP1 doesn't convert duplicate points: points taken within half a time second of the previous point or located within a distance of the last point implied by /decimals above (applied to the location expressed in minutes: e.g. /decimals=2 implies a minimum separation of 0.01 minutes). Duplicate points can be created as the GPS loses signal acquisition, in which case the last fix may be retransmitted a number of times, or when physical movement of the GPS ceases.</p>
/error	<p>Log parsing error messages in output_file.ERR. Errors are normally due to incomplete sentences at GPS capture program startup.</p>
/minfeet=n	<p>Prune (drop) points to support a precision of n feet, meters, kilometers, miles, or nautical miles.</p>
/minft=n	
/minmeters=n	
/	
minkmeters=n	<p>GPS fixes can be downloaded as often as every second or two resulting in a huge number of points which may be too close together to be</p>

<code>/minmiles=n</code>	warranted by a particular application. One of the main strengths of NMETOMP1 is the program's ability to reduce the number of points to meet the user's criteria of data spacing and accuracy. The pruning algorithm consists of two parts: fixes which are within the stated distance <code>n</code> of the last fix are dropped as being obviously unnecessary. Also intermediate points falling within a straight line corridor <code>2n</code> wide between the last accepted fix (anchor point) and the next necessary one are themselves unnecessary. A dramatic reduction in data is possible by specifying a minimum precision of even 500 feet. (Remember that <code>S/A</code> - see below - reduces the accuracy of any given fix to $\pm 300$ feet with a 95% probability.)
<code>/minmiles=n</code>	
<code>/unsupported</code>	Log unsupported NMEA sentences in <code>output_file.ERR</code> . Unsupported NMEA sentences are those of which NMETOMP1 is not aware. They neither hurt nor are interpreted. (See Supported NMEA 0183 Version 2.00 Sentences: for a list of supported sentences.)
<code>/?</code>	This option displays a brief explanation of all command line options on the screen.

## Example Input/Output:

Two samples of data input and one of data output are listed below. The data input, although conforming to the NMEA 0183 version 2.00 specification, can vary from GPS to GPS because of the variety of sentences available, their ordering, data precision, and dropped elements.

The input listed below is from a Magellan GPS Meridian, a less expensive if not popular hand-held GPS. The Meridian supports three different types of NMEA output chosen from deep within its menuing system (Aux Menu, second page, NMEA): 0183A, 0813B, and 0813C. The 0813A uses NMEA sentences from a prior standard no longer recommended for use and not supported by NMETOMP1. 0813B's output is shown in Sample Input A, and 0813C's output is shown in Sample Input B. Note that in both cases a single fix gives rise to multiple sentences which may contain redundant data. (See Supported NMEA 0183 Version 2.00

Sentences: below for a list of required and supported sentences.)  
 In the Meridian examples, 0813C (Sample Input B) gives more data unless you require that the date be included. On the other hand, Sample A's data carries mandatory checksums for data accuracy.

#### Sample Input A:

```
$GPRMB,A,0.02,L,0008,0009,4027.05,N,07935.27,W,003.9,162.,00.0,V*21
$GPRMC,200150,A,4030.81,N,07936.83,W,00.0,000.0,150394,08.,W*4C
$GPRMB,A,0.01,L,0008,0009,4027.05,N,07935.27,W,003.9,162.,00.0,V*22
$GPRMC,200156,A,4030.80,N,07936.83,W,00.0,000.0,150394,08.,W*4B
$GPRMB,A,0.01,L,0008,0009,4027.05,N,07935.27,W,003.9,162.,00.0,V*22
$GPRMC,200157,A,4030.80,N,07936.83,W,00.0,000.0,150394,08.,W*4A
```

#### Sample Input B:

```
$GPGLL,4030.899,N,07936.455,W,012611.861,A
$GPAPB,A,A,0.3,R,N,V,V,162.7,T,009,166.8,T,85.3,T
$GPGGA,012611.86,4030.90,N,07936.46,W,1,07,4.1,,,,,
$GPVTG,81.5,T,89.9,M,2.7,N,5.0,K
$GPBWC,012611.86,4027.05,N,07935.27,W,166.8,T,175.2,M,4.0,N,009
$GPGLL,4030.910,N,07936.449,W,012615.921,A
$GPAPB,A,A,0.3,R,N,V,V,162.7,T,009,166.9,T,160.9,T
$GPGGA,012615.92,4030.91,N,07936.45,W,1,07,5.1,,,,,
$GPVTG,6.0,T,14.4,M,8.2,N,15.2,K
$GPBWC,012615.92,4027.05,N,07935.27,W,166.9,T,175.3,M,4.0,N,009
$GPGLL,4030.936,N,07936.449,W,012621.369,A
$GPAPB,A,A,0.3,R,N,V,V,162.7,T,009,167.0,T,173.6,T
$GPGGA,012621.37,4030.94,N,07936.45,W,1,07,3.9,,,,,
$GPVTG,353.4,T,1.8,M,22.7,N,42.1,K
$GPBWC,012621.37,4027.05,N,07935.27,W,167.0,T,175.3,M,4.0,N,009
```

Note: GPAPB Autopilot Sentence "B" - unsupported .

GPBWC Bearing & Distance to Waypoint - unsupported.

#### Sample Output:

1	2	3	4	5	6	7	8	9
40.514983	-79.607583	0	012611.861UT	000000	81.5	5.0km/hr	4.1	7
40.515167	-79.607483	0	012615.921UT	000000	6.0	15.2km/hr	5.1	7
40.515600	-79.607483	0	012621.369UT	000000	353.4	42.1km/hr	3.9	7

1	Latitude in decimal degrees. Negative implies south.
2	Longitude in decimal degrees. Negative implies west.
3	Altitude in meters above sea level.
4	Universal Time (GMT) HHMMSS.SSS
5	Date at the Prime Meridian. DDMMYY
6	True course made good.
7	Speed over ground in kilometers per hour.
8	Horizontal dilution of precision.
9	Number of satellites in use (versus in view). 0 - 12.

**Terms defined:**

Speed over ground, track made good: GPS terminology, probably rightly, is oriented toward non-land-based conveyances (airplanes and ships). In terms of air and water navigation, ones course (track) and speed can be referenced either to that steered and covered through the air or water or referenced to the "absolute" coordinate system of the ground. Hence the unambiguous terminology speed over ground and track made good.

Dilution of precision: This is a number showing the added uncertainty of a fix due to the relative location of the satellites when the fix is made. A simple way to think of the geometry of a fix is to picture two fuzzy lines crossing each other at right angles. If the lines' fuzziness represents the limits of their accuracy, the area of fuzziness at their intersection is the uncertainty of the fix. As the lines move from the optimal right angle intersection to an oblique intersection, the area of imprecision increases. Dilution of precision due to satellite geometry works the same way. Even in the best of circumstances, a minimum value of uncertainty is established by minute variations in the satellites' atomic clocks and variations of the speed of the propagation of the signal through the atmosphere due to changes in barometric pressure, temperature, moisture, and so forth. This inherent imprecision can be amplified by the satellites' relative positions to each other and to the observer. Hence the dilution of precision. Smaller values are better.

The format of this output, known as (extended) .MP1 by Allison Software, is not precisely defined. The program MP1TOMP3 converts the first two columns, the latitude and longitude, to .MP3 format and treats all following text as comments. Whether there are adjustments in the following columns as GPS data conversion evolves remains to be seen. The data is defined in terms of the order of white space-separated characters groups and not by absolute column position.

**Example Data Reduction Session:**

From the output data left in GPS2.NME by the GPS Windows utility (\MAPIT20\EXAMPLES\GPS on your **MAPIT** CD), create two .MP3 track files, one of the raw data and the second with an average of points at a minimum spacing of 200 feet. Color the averaged data red, and combine them both in the same file for display and plotting from within **MAPIT**.

```
C:> nmetomp1 gps2 x
```

---

680 fixes output out of 680. 0 errors in 3871 lines.

```
C:> nmetomp1 gps2 x2 /minft=200 /avg
```

175 fixes output out of 680. 0 errors in 3871 lines.

```
C:> mp1tomp3 x
```

```
C:> mp1tomp3 x2 /color=4
```

```
C:> copy /b x.mp3+x2.mp3
```

```
C:> mymap x
```

The pen associated with color 4 was increased to 0.5mm wide in WIDE.PEN (made from STD.PEN) and MYMAP.BAT modified with /pen=wide to create the following plot.

### **GPS's Raw Data vs. 200 Foot Average**

The block-like character of the original track represents the resolution limits of **MAPIT** permanent line data. **MAPIT**'s dynamically displayed moving markers are not subject to this quantum limitation and will display between minimum line spacings.

### **COMBINED.MP3 Plotted at 1:300,000**

The preceding plot shows the results of three tracks taken from an automobile on both major roads (shown) and minor roads (not shown). The plot, some 15 miles wide and at approximately 3.3 times the resolution of **MAPIT**'s underlying data, demonstrates the difficulty of creating maps truly accurate in all detail. Parts of this plot make more sense when viewed in color with rivers, county, and populated place boundaries turned on. The preceding more-detailed GPS plot was taken from a portion of the left side of the smallest triangle near the upper right of this plot.



## **SIMULATE**

This **MAPIT** utility uses simulation scripting instructions to update **MAPIT**'s moving marker dBASE IV file forcing marker icons to travel across **MAPIT**'s map display in accelerated real time. The simulation scripting instructions are free-format statements describing the motion of one or more objects in terms of location, date, time, speed, and heading which the user enters into a text file. **SIMULATE** interprets these instructions, generates a time-based simulation stream, and feeds this stream to **MAPIT**'s moving marker dBASE IV GPS interface. To view the simulation, the user must simultaneously run **MAPIT** within another session of Windows while enabling the TOOLS/GPS Markers/Active menu item to make them visible. See the **MAPIT TOOLS/GPS Markers** instructions elsewhere in this manual for detailed general instructions and the **Advanced Considerations** section below for specific instructions.

### **The SIMULATE Command Line:**

Use:

```
SIMULATE script [output_dbf] [/options...]
```

where

- `script` — a text file holding simulation instructions. Assumes a .SIM extension. Command line arguments take precedence over statements in the script file.
- `Output_dbf` — an optional dBASE IV GPS Interface database to which the simulation instructions will be written. Defaults to "GPS" and assumes a .DBF extension.

options:

- `/rate=` — run the simulation at *n* times simulated time.
- `/delta_time=` — update the simulation every *n* seconds of simulated time. If not specified, the delta time is automatically calculated and adjusted to maintain `/rate` above.
- `/delay=` — delay the start of the simulation *n* seconds.

- 
- Use this option to facilitate task switching between the simulation task and **MAPIT**. Defaults to 5.
- `/startat=` — begin the simulation at this date/time offset.
  - `/stopat=` — end the simulation at this date/time offset.
  - `/full_time` — override any “start at/stop at” simulation statements in .SIM script and run the simulation its full natural length.
  - `/single=` — output only the named object
  - `/mp1=` — output the simulation tracks to an ASCII .MP1 file rather than to the .DBF. When converted to .MP3 format, the tracks are stripped of their time-dependent orientation and may be viewed in detail from within **MAPIT**.
  - `/short` — output only lat/lon pairs to an ASCII .MP1 file without date/time information to produce a smaller file.
  - `/overwrite` — delete any data in an existing .mp1 before adding new simulation tracks. By default, an error message is issued if the .mp1 file already exists and neither the `/append` nor the `/overwrite` option is specified.
  - `/append` — add newly simulated track data to the end of the .mp1 file.
  - `/dump` — create a debug dump of the contents of each leg of each object
  - `/help` — display a list of legal simulation file statements.
  - `/version` — display program’s version.
  - `/?` — display this output.

## Script File:

The required script file is a text file with the nominal file name extension .SIM. Script files hold instructions defining objects and the legs of a journey each object takes. This file is free format on a line-by-line basis and uses the ‘;’ character as a comment. Everything to the right of the comment character is ignored. Within limitations, the order of statements is not important. Object legs may be defined out of sequence within the constraints

of the syntax. (See USAIR.SIM.) The script file can also contain some statements specifying information which otherwise may be entered as command line arguments. For instance, DELTA TIME corresponds to /delta\_time=. Selections made at the command line override those made in a script file allowing you to have a general definition which you can override easily at program invocation. Reference your script file from the command line as the first command line argument as in

```
C:\> simulate goose
```

where goose.sim is a script file.

### **General statements:**

OUTPUT DBF: file_name	; Output to MAPIT's Moving Marker GPS .DBF. Defaults to "GPS".
SIMULATION RATE number	; Speed up simulation by factor <i>n</i> .
DELTA TIME seconds	; Set the time between simulation calculations in seconds.
DELAY seconds	; The delay before starting the simulation. Defaults to 5 seconds.
START AT Date/Time	; Begin the simulation at Date/Time.
STOP AT Date/Time	; End the simulation at Date/Time.
GREAT CIRCLE or GC	; Plot each leg of the course as a great circle. (Default.)
STRAIGHT LINE or RHUMB	; Plot each leg of the course at a constant heading.

If it doesn't already exist, SIMULATE creates the OUTPUT DBF and dynamically adds records to it as required by the simulation script. Each object requires one .DBF record except when two exist in non-overlapping time intervals and can therefore share the same physical record.

The SIMULATION RATE indicates how fast a simulation will run. A rate of 3600 would run the simulation at one second wall clock time for each hour of simulated time. Rates of several hundred to several hundred thousand are typical.

DELTA TIME sets the time between simulation calculations in seconds and can be thought of as the granularity of the simulation. If DELTA TIME is specified, the SIMULATION RATE may

not be attained especially under compute-bound conditions. Not explicitly specifying DELTA TIME allows **SIMULATE** to set that value dynamically in an effort to achieve the target SIMULATION RATE. In this case, **SIMULATE** initially sets DELTA TIME to 60 seconds and adjusts it up or down every few seconds as the complexity of the simulation and the multitasking load of the computer dictate.

Each simulation has a natural start and stop time dictated by its script. You can override these times to start later or end earlier by with the START AT/STOP AT Date/Time statements.

### **Date/Time format:**

MM-DD-[YY]YY, hh:mm:ss (6-7-97,16:08)

or

DD-MMM-[YY]YY, hh:mm:ss (7-June-1997,16:08)

A single number by itself defaults to the hour. If there is no date, the date defaults to 1-1-1970. Unfortunately the range of the format is rather limited. No date before 1-1-1970 is acceptable as is none after February 5, 2036 due to limitations in Microsoft's mktime() function. An attempt is made to adjust between standard and savings time based on the date.

### **Object/Leg statements:**

DEFINE object_name	; Establish an object. The name may be up to 20 characters but must not contain blanks.
Date/Time, +Time, or NEW LEG	Establish a course leg in an object. "+" indicates relative.
Latitude Longitude	; Define starting/ending point of a leg.
SPEED number	; The speed of the projection at the surface of the earth.
KNOTS, MPH, KMPH, MPS (meters/sec)	; Units of speed. (Default: KNOTS)
HEADING ddd	; Course heading in degrees from true north.
TOWARD Latitude Longitude	; Continue on this leg toward a point. May over or undershoot.
FOR Time	; Continue on this leg for this

COLOR number	time. ; The color number (0 to 15) of the object's marker as displayed within <b>MAPIT</b> . -1 to default to the <b>MAPIT</b> default.
PERIOD Time	; Period of an orbit around rotating earth. Note that a period of 90 minutes should be stated as PERIOD 1:30.
GEOSYNCHRONOUS	; Establish a geosynchronous orbit around the rotating earth.
FIXED	; Establish an orbit fixed in space (relative to the stars) around the rotating earth.

Each object must be defined once and only once by the DEFINE statement. Multiple legs may follow a definition uninterrupted or they may be interspersed with other objects. In the latter case, focus is re-established on an already-defined object by stating its name without the DEFINE keyword.

Each leg of an object is defined in context of a Date/Time. Stating a Date/Time, a relative time (+Time), or NEW LEG establishes a new leg. "+3" would start a new leg 3 hours after the last.

The keywords PERIOD, GEOSYNCHRONOUS, and FIXED imply an orbital simulation of a great circle around a rotating earth. Because the orbit is fixed in space above the rotating earth, the great circle orbit will appear shortened or lengthened depending upon its direction and will not, in general, come back to its starting point as would a great circle fixed on the earth's surface. Specifying the PERIOD of an orbit in hours, minutes, and seconds, implies a projected speed on the surface of the earth of the object which will be less than its speed at whatever altitude would be necessary to give it that period. All orbits are assumed to be circular. The circumference of the spherical earth is assumed to be exactly 21,600 nautical miles (60 miles/degree times 360 degrees). The length of a sidereal day (when the earth turns back to the same orientation in space) is assumed to be 23:56:04.098 hours. A GEOSYNCHRONOUS orbit sets the speed such that an object traveling in an easterly direction (heading = 090) remains forever above the same point on the earth. A FIXED orbit implies 0 speed with respect to the stars. Heading, in that case, doesn't matter.

## **Types of legs:**

pt to pt, time, no speed	Types 1 & 5
pt to pt, speed, no time	Types 2 & 6
pt, time, speed, toward destination	Types 3 & 7
pt, time, speed, heading, no destination	Types 4 & 8
GC Orbit around the rotating earth: pt, period, speed, heading, no destination	Type 9

## **Example Output:**

**SIMULATE** produces the following output when run against the GOOSE.SIM simulation file found on the **MAPIT** CD-ROM under the \MAPIT20\EXAMPLES\SIMULATE directory.

```
Natural starting time: 8-1-96,00:32:00, Forced
starting time: 9-30-96,00:00:00
Stopping time: 10-21-96,19:35:00
Running simulation to C:GPS.DBF
21 days, 19:35:00 sec Simulated Time at a rate
of 50000 X real time.
Estimated time to simulate: 38 seconds.
```

```
Average delta time increments: 98 seconds.
35 seconds elapsed time, 53854 effective rate
Used 3 records out of 5.
```

Note that the starting time was delayed (the simulation shortened) by a START AT statement in GOOSE.SIM. Similarly, the requested SIMULATION RATE of 50,000 X over 38 seconds yielded a 53,854 X effective rate over 35 seconds by increasing the average delta time increment from the default 60 seconds to 98 seconds. If you needed a more precise simulation, you would have to use the DELTA TIME statement to specify it exactly (at the probable expense of the simulation rate.) Three records out of a total of five were used in C:GPS.DBF.

## Example Simulation Script File:

```
; demotype.sim - Demonstrate 4 basic types.
```

```
Output DBF: c:gps
```

```
Simulation rate 1000
```

```
08:48 define xxx                                ; TYPE 1
        color 4 40.3N 80W                       ; Pittsburgh

09:30    36.9N 76.3W                             ; to Norfolk
        color 2                                  ; TYPE 2
        speed 250 mph
        42.9N 78.9W                             ; to Buffalo

New Leg  toward 36.2N 115.1W                    ; to Las Vegas
        color 6                                  ; TYPE 3
        speed 1000
        for 03:16

New Leg  heading 070 Rhumb                       ; TYPE 8
        color 5
        for 03:16
```

## Demonstration Scripts:

To view these demonstration simulation scripts, you must have two DOS sessions running concurrently under Windows, the first running **MAPIT** and the second, **SIMULATE**. Your Windows environment must be set up to allow the background execution of the second process, **SIMULATE**. From within **MAPIT**, mouse to the TOOLS/GPS Markers sub menu and click on Active to enable looking for marker activity. (If you get the warning message, "Error opening C:GPS.DBF. Create an empty DBF?", respond "Y". You may wish to click off the other options and experiment with them later. You will probably want to set TOOLS/GPS Markers/Scan delay to 0.1 seconds for minimal delay between scans. This will enhance marker response at the expense of making grabbing the attention of the top menu command levels harder, you may have to sit with your finger on the mouse button longer. Begin (Ctrl<Esc>) or context switch back (Alt<Esc>) to your other session, switch to your CD-ROM drive, and execute a script as follows.

Run these scripts by entering

```
D:\> cd mapit20\examples\simulate
D:\...> dir (See what's there.)
D:\...> ..\..\exes\simulate goose (DOS 6.x/W3.1)
D:\...> ..\..\exes\simulate goose (DOS 7/W95)
```

If you've set up a mapit directory on your hard drive and are pointing to the proper directories, you could enter:

```
C:\mapit> simulate D:goose
```

or

```
D:\...> C:simulate goose
```

(The variations and joys of DOS are boundless for the initiated.) Context switch back to your **MAPIT** session and look for the moving markers to appear after a five second delay. Enter "2" or "A" to position yourself to the whole US. While you're entering commands within **MAPIT**, the moving markers will cease to update but will restart after your command or repaint is finished.

## ***Goose.sim***

This script shows the migration of three GPS-transponder-banded Canada Geese in their flights from Canada to the US over a period of several months. The data was provided by the New York Coop Research Unit (USFWS-NBS) at Cornell University and Ducks Unlimited with the help of Susan Sheaffer at Cornell. Every few hours or days, a passing satellite would turn on a lightweight GPS attached to each bird thereby establishing its position. This location was relayed back to the original satellite and then on to a data gathering site in Maryland. This data demonstrates the advantages as well as the limitations of **SIMULATE**. If you plot the tracks of the geese by entering the data directly into a .MP1 file and converting it to .MP3 format for display by **MAPIT**, you can see where the geese have flown but have no idea when or of their relationships to each other over time. Displaying the data through **SIMULATE** immediately shows the flocks as a function of time clearly revealing their relationships to each other in space at any particular time. You will also note that all three flocks begin their migration within a few days of each other. The granularity of the data and the limitations of **SIMULATE** become apparent. **SIMULATE** calculates the average speed between sightings or fixes, if you will. Sightings separated by several days and a few miles may show an average speed of only one or two miles an hour, well below goose stall speed, when, in point of fact, the birds



were actually stationary most of the time with a brief but fast flight between locations. Similarly when viewing data over a long period, one may choose a large SIMULATION RATE (50,000 in this script) to speed the process thereby masking short but significant side excursions which warrant more detailed examination under a finer DELTA TIME simulation. You can override script simulation settings from the command line as demonstrated below.

## **USAir.sim**

Taken from the *USAir City Timetable for Pittsburgh*, USAIR.SIM simulates many of the Monday, May 8, 1995 morning flights into and out of the Greater Pittsburgh International Airport. The assumptions are that the aircraft are in flight from origin to destination between the listed gate times, and that they fly great circle routes from point to point. Obviously these assumptions are oversimplified, but the resulting simulation reflects, to a greater or lesser extent, the actual traffic patterns around Pittsburgh and to many places in the country and the world flown by USAir from Pittsburgh. USAIR.SIM is a large file simulating up to 271 simultaneous flights. It is easy to get buried in the data. You may want to view it several different times in several different ways. As delivered, **MAPIT**'s history file MAPIT.SAV contains four different views (positions) associated with digits 1 through 4.

pos	zoom	location	
1	2.6	37:40N 55:24W	US & Europe
2	6.4	37:29N 97:05W	US
3	19.4	40:35N 80:00W	NE US
4	173.6	40:27N 79:49W	Western PA

Positions 1 and 2 (POSITION/Restore *n*, or just type *n* from within **MAPIT**) give global and US views, position 3 shows more detail, and position 4 shows aircraft activity over the Pittsburgh area with the type of detail an air-traffic controller might see. You can most rapidly switch between views if you run **MAPIT** without any background data.

```
C:\> mapit nul /noe
```

From the broader viewpoints, turn on TOOLS/ GPS Markers/Marker trails to see the entire air routes. Marker trails off displays the location of each aircraft as it flies en-route. The simulation is set to run from 6 AM until 2 PM. If you start **MAPIT** with the command line argument /FULL\_TIME, you'll see all the data for a whole day. Notice that the early morning red eye flights starting on the west coast heading to arrive in Pittsburgh about morning rush hour. There is a flurry of air traffic between 9:30 and 10:05 AM which you can examine in detail by reducing the DELTA TIME to 15 seconds for a smoother display.

```
C:\> simulate USAir /delta=15 /start=5-8-95,9:30 /stop=5-8-95,10:05
```

View this from positions 4 and 3.

After running the USAir simulation, delete the output file C:GPS.DBF because it is large enough to slow down subsequent smaller simulations.

### USAir.sim Output, Pos 2, /Full\_time

#### ***orbits.sim***

This simulation depicts satellites in various circular orbits. For it to be effective, you should have the markers trails option turned on in **MAPIT** and should delete the file C:GPS.DBF to speed repaint if you've just run the USAir simulation. View this simulation from the standard world view.

The following map was produced from output from ORBITS1.SIM combined with a great circle through Cape Canaveral, Florida and demonstrates a rocket's being launched on a heading of 090 for minimum north/south trajectory excursion. Notice how, because of the earth's rotation, each pass falls short of the previous one while the great circle passes back through the same point. The figure eight over Africa is a geosynchronous orbit with a vertical component viewed for a day.

You can replicate the rocket orbit from your **MAPIT** hard drive directory with:

```
C:\> simulate orbits1 /single=153 /mp1=temp
C:\> mp1tomp3 temp /color=4
```

C:> **mymap temp**

***day.sim***

***year.sim***

These simulations show the positions of the sun and a stationery object (with respect to the stars) over the course of a day and year. See the scripts for instructions about altering the scripts and zooming in to view the end conditions. You may well ask weighty questions such as, "What is a day? What is a year?" The technical details of these simulations is beyond the scope of this manual and certainly beyond the understanding of its author. View these simulations first from the standard world view.

## **Advanced Considerations:**

**SIMULATE** dynamically adds to the length of its output .DBF as needed. **MAPIT** will scan to the end of the .DBF looking for active records. If the numbers are large, this scan, even if there are few active records, can take an appreciable time (the better part of a second or more depending upon processor speed and loading). The TOOLS/GPS Markers/Scan delay, commonly set to 1 second for GPS and other slow sources but to 0.1 seconds for smoother display of accelerated simulations, is added to the natural delay of the .DBF scan and display. The total of these is the time between the redisplay of marker icons and affects the smoothness of movement and the closeness of adjacent marker displays. If you've just run a simulation such as USAIR.SIM which can allocate almost 300 records, subsequent less-demanding simulations can be adversely affected. Your best bet, then, is to delete the C:GPS.DBF database before running the smaller simulation. You cannot delete that file while **MAPIT** is running and has it active. Either deactivate it or exit **MAPIT**, then delete. If you restart **MAPIT** immediately, **MAPIT** will prompt you when it tries to open the non-existent file asking whether it may create it. Answer "Y"es. If you start **SIMULATE** first in the absence of its target file, it will notify you that it's creating a new file and then create it. Either way insures that you will run with the minimum database length required for that simulation under those settings. **SIMULATE** reuses inactive records during a simulation, but the exact amount of overlap and therefore the requirements is dependent upon the DELTA TIME, SIMULATION RATE, and start/stop times of each simulation.

## APPENDIX A — FEATURE/LAYER ASSIGNMENTS

### By Feature:

#### Political/Ocean Boundaries

INTL_PO	80
TREATY_PO	82
OCEAN_PO	83
DATE_LINE_PO	88
ADMIN1_PO	81
ADMIN2_PO	89
ADMIN3_PO	94
SPECIAL_PO	95
DISPUTED_PO	96
MET_PO	65
ICE_PO	84
ICE_WATER_PO	86
GLACIER_PO	44
SNOW_PO	45
ICE_LAND_PO	46
COAST_PO	85
SEA_WALL_PO	87
POLITICAL_TXT	117
COUNTRIES_TXT	118
STATES_TXT	64
COUNTIES_TXT	66

#### Rivers

RIV_DN	40
LAKES_DN	41
SAND_DN	42
CANALS_DN	43
CON_DN	47
DRAINAGE_PT	74
DRAINAGE_TXT	75

#### Roads

DIV_HWY_RD	70
PRI_HWY_RD	71
TRAILS_RD	72
CON_RD	73

#### Railroads

SINGLE_RR	90
MULTI_RR	91
LIGHT_RR	92
CON_RR	93

#### Transportation

SHEDS_TS	54
----------	----

BRIDGES_TS	55
CAUSEWAYS_TS	56
TUNNELS_TS	57
FERRIES_TS	58
FORDS_TS	59
Also specifies airports	
TRAN_STRUCT_TXT	78

#### Utilities

POWER_UT	100
PHONE_UT	101
EX_PIPES_UT	102
UN_PIPES_UT	103
UTILITIES_TXT	62
TOWERS	63

#### Cities & Landmarks

CULT_LM	50
POP_PLACES_TXT	119
0 population	
CITY_0	111
1 - 1000	
CITY_1	112
1001 - 10,000	
CITY_10	113
10,001 - 100,000	
CITY_100	114
>100,000	
CITY_BIG	115

#### Ocean Features

MISC_OF	51
REEFS_OF	52
LIMITS_OF	53
DEPTH_200M_OF	97
DEPTH_1000M_OF	98
DEPTH_2000M_OF	99
DEPTH_3000M_OF	104
DEPTH_4000M_OF	105
DEPTH_5000M_OF	106
DEPTH_6000M_OF	107
DEPTH_7000M_OF	108
CONT_SHELF_OF	109
TEC_PLATES_OF	110
FRAC_ZONES_OF	67
LINEATIONS_OF	68
MAJOR_SEAS_TXT	76

## 112 APPENDIX A

---

OCEAN\_FEAT\_TXT 77

LAND\_COVER 60

LAND\_COVER\_TXT 116

**Land Cover**

**By Layer:**

RIV_DN	40
LAKES_DN	41
SAND_DN	42
CANALS_DN	43
GLACIER_PO	44
SNOW_PO	45
ICE_LAND_PO	46
CON_DN	47
CULT_LM	50
MISC_OF	51
REEFS_OF	52
LIMITS_OF	53
SHEDS_TS	54
BRIDGES_TS	55
CAUSEWAYS_TS	56
TUNNELS_TS	57
FERRIES_TS	58
FORDS_TS	59
LAND_COVER	60
UTILITIES_TXT	62
TOWERS	63
STATES_TXT	64
MET_PO	65
COUNTIES_TXT	66
FRAC_ZONES_OF	67
LINEATIONS_OF	68
DIV_HWY_RD	70
PRI_HWY_RD	71
TRAILS_RD	72
CON_RD	73
DRAINAGE_PT	74
DRAINAGE_TXT	75
MAJOR_SEAS_TXT	76
OCEAN_FEAT_TXT	77
TRAN_STRUCT_TXT	78
INTL_PO	80
ADMIN1_PO	81
TREATY_PO	82
OCEAN_PO	83
ICE_PO	84
COAST_PO	85
ICE_WATER_PO	86
SEA_WALL_PO	87
DATE_LINE_PO	88
ADMIN2_PO	89
SINGLE_RR	90
MULTI_RR	91
LIGHT_RR	92
CON_RR	93
ADMIN3_PO	94
SPECIAL_PO	95
DISPUTED_PO	96
DEPTH_200M_OF	97

---

APPENDIX A — Feature/Layer Assignments 114

DEPTH_1000M_OF	98
DEPTH_2000M_OF	99
POWER_UT	100
PHONE_UT	101
EX_PIPES_UT	102
UN_PIPES_UT	103
DEPTH_3000M_OF	104
DEPTH_4000M_OF	105
DEPTH_5000M_OF	106
DEPTH_6000M_OF	107
DEPTH_7000M_OF	108
CONT_SHELF_OF	109
TEC_PLATES_OF	110
CITY_0	111
CITY_1	112
CITY_10	113
CITY_100	114
CITY_BIG	115
LAND_COVER_TXT	116
POLITICAL_TXT	117
COUNTRIES_TXT	118
POP_PLACES_TXT	119

## APPENDIX C — PEN DEFINITIONS

For additional flexibility during plotting, **MAPIT** provides override access to internal definitions of pen assignments, line type and width assignments, and plot border and lat/long grids assignments via the pen definition file. The standard definitions are outlined in the default file `std.pen` shown below. Change a copy and invoke **MAPIT** with the command line calling parameter `/pendefinition=my_file` to customize your plotting operations.

The standard pen color assignments given by *The HP-GL/2 Reference Guide* are as follows. They may not be reflected in your plotting hardware/software implementation. Certainly if you have a real pen plotter with physical pens, you are free to put whatever colors you desire in whatever holders.

Pen	Color		Pen	Color
0	white		4	yellow
1	black		5	blue
2	red		6	magenta
3	green		7	cyan

```
; Std.pen - This file reflects the default internal HP-GL/2 pen plotting
;           settings of MAPIT. Copy and change to your liking invoking
; with:
;           MAPIT /pendefinition=file_name.pen

; Video Color is the index into this table meaning that the MAPIT layer
; (color) determines the pen, width, and line type according to this table
; when making HP-GL/2 plots.

; MAPIT doesn't allow reassigning the default color associated with pens.
; If pen 2 defaults to red on your output device, that is the color you
; are stuck with. Sorry, no robin's egg blue.

; 0.1 mm is supposed to be narrowest width. 0 may be narrower on some
; devices.

; Some devices support 8 pens. MAPIT allows 16.

; Line type 0 is a solid line. Types 1 - 8 are variously dashed and are
; device dependent.

; Video Color Pen # Width in mm. Line type Video to pen color
0 0 0 0 ; black to black (white)
1 5 0 0 ; blue to blue
2 3 0 0 ; green to green
```



---

3	7	0	0	; cyan to cyan
4	2	0	0	; red to red
5	6	0	0	; magenta to magenta
6	4	0	0	; brown to yellow
7	1	0	0	; white to black
8	1	0	0	; gray to black
9	5	0	0	; light blue to blue
10	3	0	0	; light green to green
11	7	0	0	; light cyan to cyan
12	2	0	0	; light red to red
13	6	0	0	; light magenta to magenta
14	4	0	0	; yellow to yellow
15	1	0	0	; bright white to black

; The plotting characteristics of MAPIT's lat/long grid and print borders  
; can be specified independently of the standard assignments in the above  
; table for raster devices.

gridpen=1 ; Specifies pen color only.  
gridwidth=0  
GridType=5

borderpen=1 ; Specifies pen color only.  
borderwidth=0.5  
bordertype=0

## **APPENDIX E — STANDARD LAT/LON NOTATION**

**MAPIT**, **MP1TOMP3**, **GARTODBF**, and other **MAPIT** utilities use the standard lat/lon notation defined below, basically a latitude followed by a longitude in free or fixed format.

Fixed Latitude Formats:

DDMM.mmm	2316N
DDMMSS.sss	133947.6642 S

Fixed Longitude Formats:

DDDMM.mmm	12318.002
DDDMMSS.sss	-0042200

Free Formats:

[D]DD.ddd	42.667
[D]DD:MM.mmm	101:40 W
[D]DD:MM:SS.sss	S 16:0:3.5

Note: any number of places to the right of the decimal may be used. The numeric string may be preceded by an option Indicator I or - or followed by an I. See definitions below.

Fixed formats refer not to character positions on a line, but to the relationship of a part of a token with its starting character. (A fixed format of 4 characters means that a particular part of a latitude must be four characters wide, not that it must be the fourth character in the line.)

In those cases in which input is taken from a text file (**MP1TOMP3**) a latitude/longitude pair is expected on each line. In fact, the pairs may be given in reverse order (longitude first) if that fact is unambiguously apparent and in a number of different formats. The latitude/longitude pair can be placed anywhere on any given line as long as they appear as the first two non-blank tokens.

- I — Indicator. 'N' or 'S' for latitude. 'E' or 'W' for longitude. Case insensitive. Indicators are optional and may be preceded and/or followed by optional white space (blank and tab characters). Indicators are themselves optional and may precede or follow a number but may not both precede and follow the same number. (Coordinates of the form N12E34 are legal, however, because the second indicator,

- 'E', acts as a separator and is interpreted as preceding the second number.
- — The minus sign may be used to indicate either 'S' or 'W'. If used, it must always precede and never follow the number it modifies. Whereas indicators imply latitude or longitude, the minus sign implies neither and may not be used in combination with an indicator.
  - D — Degrees.
  - M — Minutes.
  - S — Seconds.
  - ; — The comment character causes all following characters on the line to be ignored. If placed first, the line is as if it didn't exist thus preserving the continuity of a multisegmented line.

If no indicator or no fixed format describes or implies which of the pair is the latitude and which the longitude, latitude is assumed to be first. Inconsistent indicators or fixed formats generate error messages.

N pairs of coordinates, each on a line of a file, yield n - 1 line segments. Lines are terminated by one or more blank input lines. (See ';' above. Actual creation of a line takes place in MP1TOMP3.) Thus 100 points with a blank line between the 10th and 11th points will yield two lines in a .MP1 file, the first with 9 segments and the second with 89.

#### Terrible Data Examples:

```
; From file special.pt
39.807655 -91.049645
N3948.4831 W09102.9837
3948.5255NW09102.9713
3948.5390N W09103.0006
3948.5275N 09103.0002W
N 3948.5422W 09103.0565
N 3948.5472 09103.1256 W
N 3948.5275 W 09103.1252
3948.5005 -09103.1128
3948.4767-09103.0336
N39:48.4774 W091:03.0336
39:48.0034N 091:02.9576W
39:46.9837 N 091:03.9729 W
N39:46:51.774 -91:03:53.652 ;N3946.8629 W09103.8942
394658.734 -0910359.832 ; N3946.9789 W09103.9972
```

---

## **INDEX**

<Enter>, 8  
<Esc>, 7  
<Tab>, 7, 27  
Arrow Keys, 6  
City, txt\_offset, 16  
**Clock Reset, GPS**, 83  
*Color numbers*, 12, 40  
COM Pin Connections, 87  
Database, 30, 32  
dBASE Editor, 22, 38  
dBASE File Format, GPS, 85  
Definition File  
    GEOCODE, 59  
    GEOTOENT, 67  
    GEOTOMP1, 73  
    pen, 112  
    SIMULATE, 98  
Distance, 27, 29  
E\_Type, 22  
Edit DBF Entities, 38  
Extended Files, 41  
Features, 49, 108  
Features Display, 49  
FIGEDIT, 54  
File Names and Conventions, 50  
Fonts, 45  
Fracture Zones, 50  
**Gazetteer**, 32  
GEOCODE, 58  
*Geocoding*, 12  
Geosynchronous, 101  
GEOTOENT, 66

- GEOTOMP1, 72
- Goose.sim, 104
- GPS, 80
- GPS Markers, 35
- Great Circles, 8, 27, 29, 61, 72, 99
- Help, DOS command line, 9
- HP-GL/2, 5, 33, 112
- Lat/Lon, Standard Notation, 114
- Layers, 39, 109
- Line of Sight, 29
- Magnetic Lineations, 50
- MAPIT
  - auxiliary files*, 13
  - command line, 19
  - computer requirements, 2
  - copy, 48
  - gotcha's, 16
  - problems and solutions, 17
  - tips, 18
  - tools, 27
  - utilities, 8
- Maps, making, 5
- Marker Trails, 36
- Mercator, 27
- mktime() function, 100
- Mouse
  - installing in W3.1, 4
  - left button, 7
  - use, 5
- MP1TOMP3, 89
- NMEA 0183, 35, 87, 93
- NMETOMP1, 91
- Orbits.sim, 106
- PCX, 5, 34, 37

Period, 101  
*Pins, sticking into a map*, 10  
Plate Boundaries, 50  
Plotting, 5, 33  
Position Menu, 8, 26  
Private Files, 41  
**Prune**, 84, 91  
Regions, 52  
Register PCX, 37  
Rhumb, 27, 72, 99  
Scale, 25, 50, 56  
*Scan Table*, 11, 20  
Scatter Diagrams, 15  
Script File, SIMULATE, 98  
**SET TZ**, 3  
Sidereal Day, 101  
SIMULATE, 97  
Standard World, 26  
Status  
    Bottom Status Line, 6  
    Header Status Field, 6  
*Style*  
    new style entities, 38  
    old style entities, 41  
    *old vs. new*, 13  
Tiles, 23  
**Time Zone**, 3  
*Tracking*, 11  
Type Face, 45  
USAir.sim, 105  
Windows 3.1, 3  
Windows 95, 3  
Xline, 44  
Zoom, 24

