Introduction to GRASS 4

James Westervelt

June 11, 1991

Contents

1 Introduction **1**

2 What is GRASS? **1**

2.1 GRASS is Public Domain 1

2.2 GRASS is a Community of Users 2

2.2.1 Inter-Agency Steering Committee 2

2.2.2 Annual GRASS User Meetings 3

2.2.3 GRASS Government Users 3

2.2.4 Universities 4

2.2.5 Commercial Firms 4

2.3 GRASS is a Software Package 4

2.3.1 GRASS is a Programming Language 5

2.3.2 GRASS is Software for GIS Specialists 6

2.3.3 GRASS is a Software Development Environment 6

2.4 GRASS is Part of a Garden 7

2.5 GRASS is a Sample Data Base 7

3 GRASS Hardware Requirements **7**

4 Why Does GRASS Exist? **9**

4.1 Original Motivations 9

4.2 UNIX Based GIS 9

4.3 Easy to Distribute 11

4.4 Market Stimulus 11

4.5 Great for Sites With Talent 11

4.6 Huge, Friendly User Community 11

4.7 GRASS Versions and History 12

5 Programs **12**

6 GRASS Design and Implementation **15**

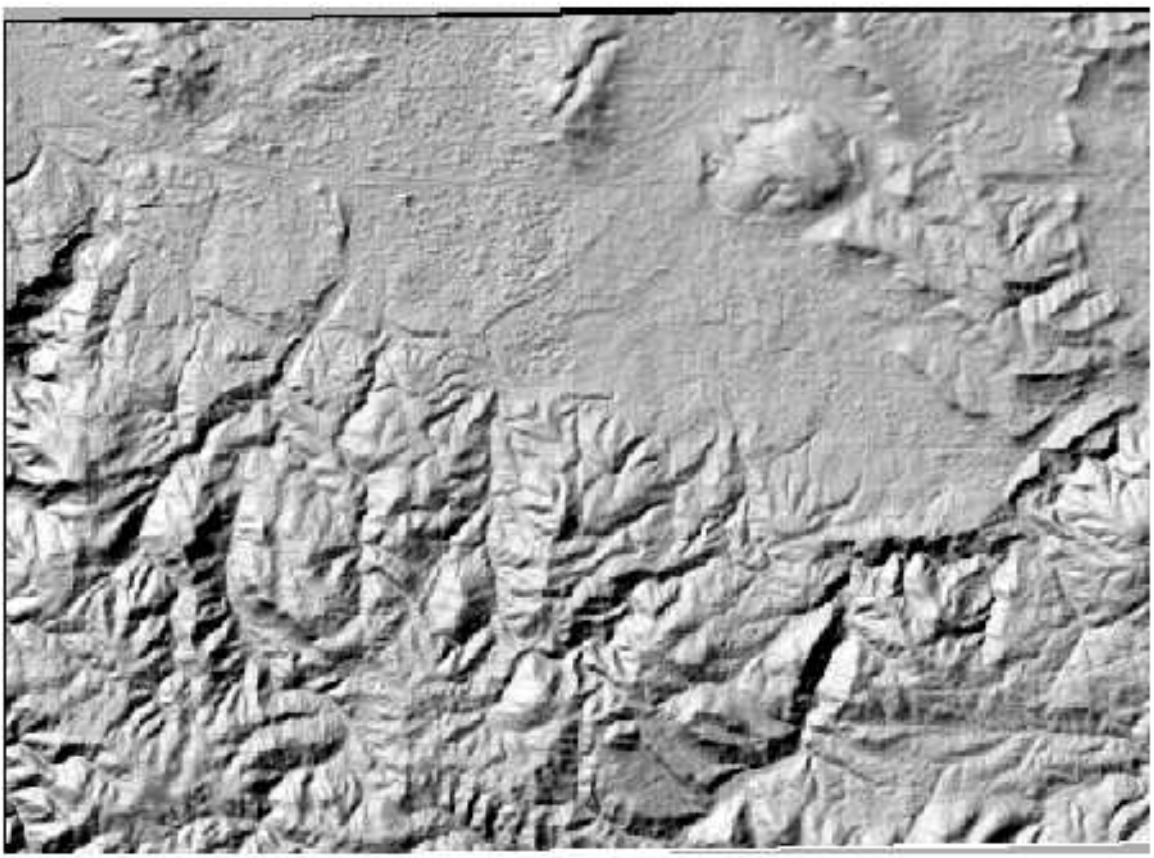
7 GRASS Research and Development **17**

A GRASS Commands **19**

B OVERHEADS **25**

Abstract

Welcome to GRASS, the Geographical Resources Analysis Support System. This document provides information on what GRASS is, where it comes from, and where to get more information. Basically, GRASS is a public domain raster geographical information system (GIS), vector GIS, image processing system, and graphics production system. It is used extensively at government offices, universities, and commercial organizations. It encompasses some 250 user programs, is written mostly in C for UNIX[1] machines, and contains more than 300,000 lines of software code. Overheads are provided to help the reader communicate GRASS to others.



Shade Relief Map of Spearfish - GRASS Sample Data Base

[1] UNIX is a trademark of ATT

1 Introduction

This document introduces the reader to the public domain geographical information system GRAASS. Information on the capabilities of GRASS are matched with discussions on the ongoing development efforts, inter-agency cooperative programs, and the GRASS user community. Appendix A provides a brief overview of the primary GRASS programs as released in the summer of 1991. Appendix B provides some overheads that may be used in local GRASS briefings.

2 What is GRASS?

GRASS, more formally known as the Geographical Resources Analysis Support System, is fundamentally a geographical information system (GIS). The question, What is GRASS? can be answered in at least the following ways:

- GRASS is Public Domain
- GRASS is a Community of Users
- GRASS is a Programming Language
- GRASS is Software for GIS Specialists
- GRASS is Software for Image Processing
- GRASS is a Software Development Environment
- GRASS is Part of a Garden
- GRASS is a Sample Data Base

Each of these answers is discussed in the following sections.

2.1 GRASS is Public Domain

Many people come to GRASS with the lure of *free* software. GRASS, developed with United States taxpayer funds, is available to the public without charge. Users should however expected to encounter nominal charges for distribution, tapes, phone support, installation assistance, turn-key systems, documentation, etc.

Several commercial and university organizations are involved with porting GRASS to and supporting GRASS on various hardware platforms. For the latest information concerning GRASS status, distribution and support centers, and user mailing lists, contact the *GRASS Information Center*:

GRASS Information Center
U.S. Army CERL
P.O. Box 9005
Champaign, Illinois 61826-9005

(217) 373-7220

The *GRASS Information Center* can provide the following type of documents:

General Information — Includes documents that introduce GRASS, describe the GRASS Inter-Agency Steering Committee's functions (see Section 2.2), and describes the various government, commercial, and government participants in GRASS.

GRASSCLIPPINGS — Place yourself on the mailing list for this page newsletter that is mailed several times a year.

Installing GRASS — Many documents describing hardware configurations, distribution and support sites, establishing return-on-investment estimates, and assistance in choosing the right system for you.

Educational Materials — Materials that you can use in briefings to describe GRASS to your co-workers.

GRASS Community Participation — This category includes documents that describe participation in the GRASS Steering Committee, the GRASSCLIPPINGS newsletter, the annual user meeting, and advertising possibilities.

System Development — For those sites interested in porting GRASS or creating new commands, there are many documents describing the GRASS programming environment.

Videos — Several videos which introduce GRASS and review some applications.

Advice — Advice on who to turn to for GRASS assistance.

2.2 GRASS is a Community of Users

Communities of GRASS users exist worldwide. Unlike user groups of commercial products, the GRASS user community members rely primarily on each other for support, system upgrades, and information. This process provides a wealth of "free" information and support for all members.

Figure 1 depicts the distribution of the GRASS user community based on a 1989 survey in which 500 sites responded.

2.2.1 Inter-Agency Steering Committee

GRASS is coordinated in large measure by the **GRASS Inter-Agency Steering Committee**. This not-for-profit organization meets 2–3 times a year for the purposes of coordinating GRASS community activities. Issues generally addressed include:

- Software design and development
- Software release schedules
- Inter-system integration issues
- User support issues
- Annual *GRASS User Meeting* arrangements
- Newsletter schedules and budget
- Inter-agency coordinations
- Commercialization issues

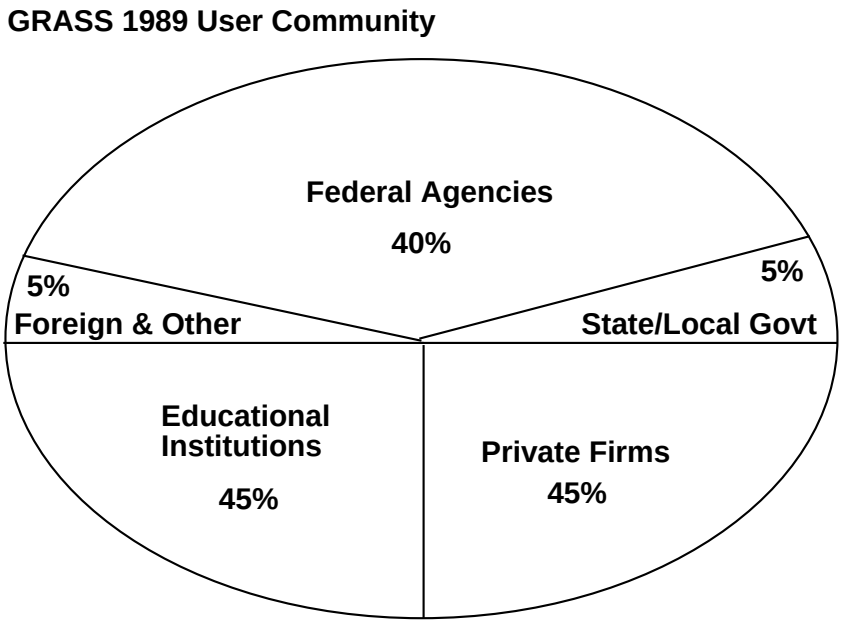


Figure 1: The GRASS User Community

The Steering Committee members represent GRASS users in many different Government agencies. As of Spring 1991, the agencies listed in table 1 had all participated in the GRASS Inter-Agency Steering Committee.

In addition to the representation of government groups, numerous universities and commercial groups have been involved as associate (non-voting) members.

2.2.2 Annual GRASS User Meetings

The GRASS user community is most visible each year when they gather together to present papers and participate in discussions ranging from GRASS fundamental design and development to experiences in applying GRASS to real-world problems. Meetings have been held annually since 1983; each meeting attracts 200-300 people who spend up to 5-days immersing themselves in the GRASS community. Meetings have been held in such exotic locations as Berkeley, California; Washington, D.C.; Champaign, Illinois (the birthplace of GRASS); Stennis Space Center (near New Orleans); Fort Hood, Texas; and Fort Lewis, Washington.

Annual GRASS user meetings have been held in Europe since 1990. Turnouts there suggest a long, healthy European user community. As of this writing plans are being considered for an Australian user meeting.

2.2.3 GRASS Government Users

The primary sources of funding for the design and development of GRASS has come from the largest government users of GRASS. The National Park Service and the Soil Conservation Service, in addition to the Army Corps of Engineers, has established GRASS support centers of expertise for GRASS within their agencies.

U.S. Army Construction Engineering Research Lab	Champaign IL.
USDA, Soil Conservation Service	Washington DC.
National Park Service	Denver CO.
Bureau of Census	Washington DC.
Bureau of Land Management	Washington DC.
Department of Defense, Environmental Office	Alexandria VA.
Environmental Protection Agency, ERL-Corvallis	Corvallis OR.
NASA	Stennis Space Center MS.
NOAA	Boulder CO.
Navy, Naval Oceanographic Lab	Stennis Space Center MS.
U.S. Army Corps of Engineers	Washington DC.
U.S. Army Engineering and Housing Support Center	Ft. Belvoir VA.
U.S. Army Engineering Topographic Lab	Ft. Belvoir VA.
U.S. Geological Survey	Reston VA.

Table 1: Organizations involved with Guiding GRASS

2.2.4 Universities

GRASS was modeled in many ways after UNIX; a system whose original followers were graduate programs in computer science at numerous universities. GRASS similarly has found a firm and growing user community within numerous university departments including geography, geology, anthropology, regional planning, landscape architecture, archeology, and entymology. GRASS is taught at nearly 50 different Universities across the world today. The number grows yearly as GRASS offers a user environment for computer scientists as well as artistic landscape architects.

2.2.5 Commercial Firms

GRASS is being strongly embraced by many commercial entities. Many commercial firms that cannot afford commercial packages choose GRASS for the purpose of "getting into the GIS market." Many of the GRASS software distribution sites provide GRASS enhancements. The inexpensive nature of GRASS allows many potential GIS users to get experience with GRASS. This increases the number of people, firms, agencies, and universities that are familiar with the technology; this in turn broadens the commercial GIS marketplace.

2.3 GRASS is a Software Package

Yes, GRASS is a software package. This software package could only grow out of the type of user community described in the previous section. It is the user community which will drive GRASS design and development into the future.

Even at the software level, GRASS is different things to different people. Consider some of the statistics:

- Nearly 200 user programs
- Nearly 4000 software files
- More than 300,000 lines of C program code

- Approximately 300 subroutines
- Programs written at about 15 different locations
- About 40 man-years of development time
- Distributed with several other public domain software packages
- Interfaced with numerous RDBMSs, expert systems, image processing systems

GRASS programs themselves are divided into the following groups:

Number	Programs that
40	Render images on monitors and paper
57	Manipulate raster maps
31	Manipulate vector maps
6	Manipulate sites maps
26	Manipulate multi-spectral imagery
16	Manage the data base
~20	Programs in alpha test
~20	Contributed programs
Numerous	"Related" Programs

2.3.1 GRASS is a Programming Language

Though GRASS, like most GISs, has a user community that operates the programs directly, GRASS has been conceived to be a programming language. UNIX itself is a programming language; it provides shells (like the Bourne shell (sh), the C-shell (csh) and the Korn-shell (ksh)). The shells provide programming flow control, and user input and output, and use other UNIX programs as subroutines. GRASS programs join the community of UNIX subroutines and, hence, enhance the power and flexibility of the UNIX operating system.

A growing community of end-users use shell scripts (UNIX macros) that contain GRASS programs as part of their main analysis functionality. In many cases the users of these scripts need know nothing about GRASS. Additionally, they need know nothing about other UNIX commands that may be embedded deep in the scripts.

To improve the possibilities of writing UNIX shell scripts, the GRASS design and development community is providing an application generator. This is a shell-script-like language that allows the script writer to communicate with the user through a graphical user interface on machines supporting X-windows. This program is called **xgen**; an **X**-windows based application **gen**erator.

What are the advantages of using a GIS as a programming language?

A GIS is hard to learn — Geographical information systems are powerful, general-purpose, and highly flexible software packages. Even a basic user must have significant knowledge of not only the GIS, but of the computer, the peripherals, the data bases being analyzed, the areas under analysis, and the requirements of the analysis. Most every GIS has been **developed as a scientific tool box**. They are generally not suitable for the casual or occasional user. If a user does progress through the learning curve to become proficient with a GIS, that user can be more effective if the GIS is a programming language. Using this language, software can be written for the local users who have limited analyses that are used repetitively. These local users need never be bothered with the scope and strength of the full-blown general purpose system.

A GIS is Not a DSS — Decision Support Systems (DSS) in landscape planning, simulation, and analysis are best done if they approach a narrow problem. A GIS can play an integral role in the design and operation of a DSS in addition to graphical user interfaces, statistical packages, expert systems, report generators, etc. A GIS as a programming language allows the GIS expert user to generate decision support systems that are:

- Application specific
- Location specific
- User specific

Backend to User Interfaces — Numerous times users have suggested that GRASS be provided with various different types of front-ends: graphical user interfaces, expert systems, data base management systems, and natural language interfaces. With GRASS developed as a programming language, all of these types of interfaces are much easier to implement. As of this writing several such interfaces are under development at different sites.

Programming languages have long lives — User interfaces have relatively short lives. The recent fascination with 2-dimensional graphical user interfaces may give way to 3-dimensional interfaces, natural language interfaces, intelligent interfaces that "get to know the user", and holographic interfaces. Behind these interfaces can be the old tried-and-true systems. A GIS, stripped of its user interface, will command a much longer life than a GIS with a intimately integrated user interface. GRASS, fundamentally, is a command-line driven language devoid of any consequential interface. It is suitable as is to support numerous interfaces for years to come.

2.3.2 GRASS is Software for GIS Specialists

GRASS is most often viewed as software designed to do serious geographical data management, analysis, and display. Initially it was designed as a **raster-based GIS**. A co-developed **image processing package** was merged into GRASS early in its development because of the obvious utility in providing traditional GIS operations to image processing and vice-versa. GRASS always provided a vector approach to digitizing and data input. In recent years, this tradition has blossomed into a developing **vector-based GIS**. Chapter 5 provides a review of the GRASS 4.0 commands and displays the variety of capabilities offered.

2.3.3 GRASS is a Software Development Environment

At another level, GRASS is a software development environment for C programmers. To facilitate the design and development of new modules in GRASS, users at this level are provided with:

- Approximately 300 GRASS software subroutines that provide interface to the data and the user
- GRASS source code
- The document *GRASS 4.0 Programmer's Manual*
- Access to a community of GRASS Programmers

GRASS capabilities that have become part of the standard GRASS distribution have come from more than 15 different sites. Innumerable other sites have programmed local capabilities in GRASS.

2.4 GRASS is Part of a Garden

GRASS grows in a "garden." In the process of using GRASS for, say, an Environmental Impact Statement, a user site may employ the capabilities of many other software systems. Some of these are public domain (or at least free) systems that are actually distributed with GRASS. Some of these include:

RIM	A relational data base management system out of the University of Washington, Seattle
MAPGEN	A cartographic output package for plotters developed by the USGS
GCTP	A coordinate translation package developed by the USGS
pbmplus	An image manipulation package developed by Jeff Poskanzer
xgen	A shell-script like language for generating X-windows interfaces

UNIX commands like awk, sed, grep, cat, csh, and sh round out a powerful programming environment in which these programs become the programmers subroutines. Other packages available on one's local system can provide yet more capabilities.

2.5 GRASS is a Sample Data Base

GRASS software is delivered with three sample data bases:

Spearfish — This famous data base (see Figure 2) was first developed by (or for) EROS Data Center. It became a member of the GRASS package and was greatly enhanced to provide more sample data layers. It is a complete data base that can be used to learn about GRASS.

Imagery — This data base provides several raw satellite and NHAP images for the Spearfish area which can be manipulated by users to generate new layers for Spearfish.

World — Portions of a larger data base used to study the earth. This data base demonstrates the format for a latitude-longitude data base as well as providing an environment for demonstrating GRASS's latitude-longitude capabilities.

3 GRASS Hardware Requirements

GRASS was designed and developed to operate under the UNIX operating system. It has been ported to numerous hardware environments supporting this operating system including:

- Sun computers (all)
- Concurrent workstations (formerly Masscomp)
- Intergraph Interpro workstation
- Silicon Graphics workstations
- Apple Macintosh (AUX)

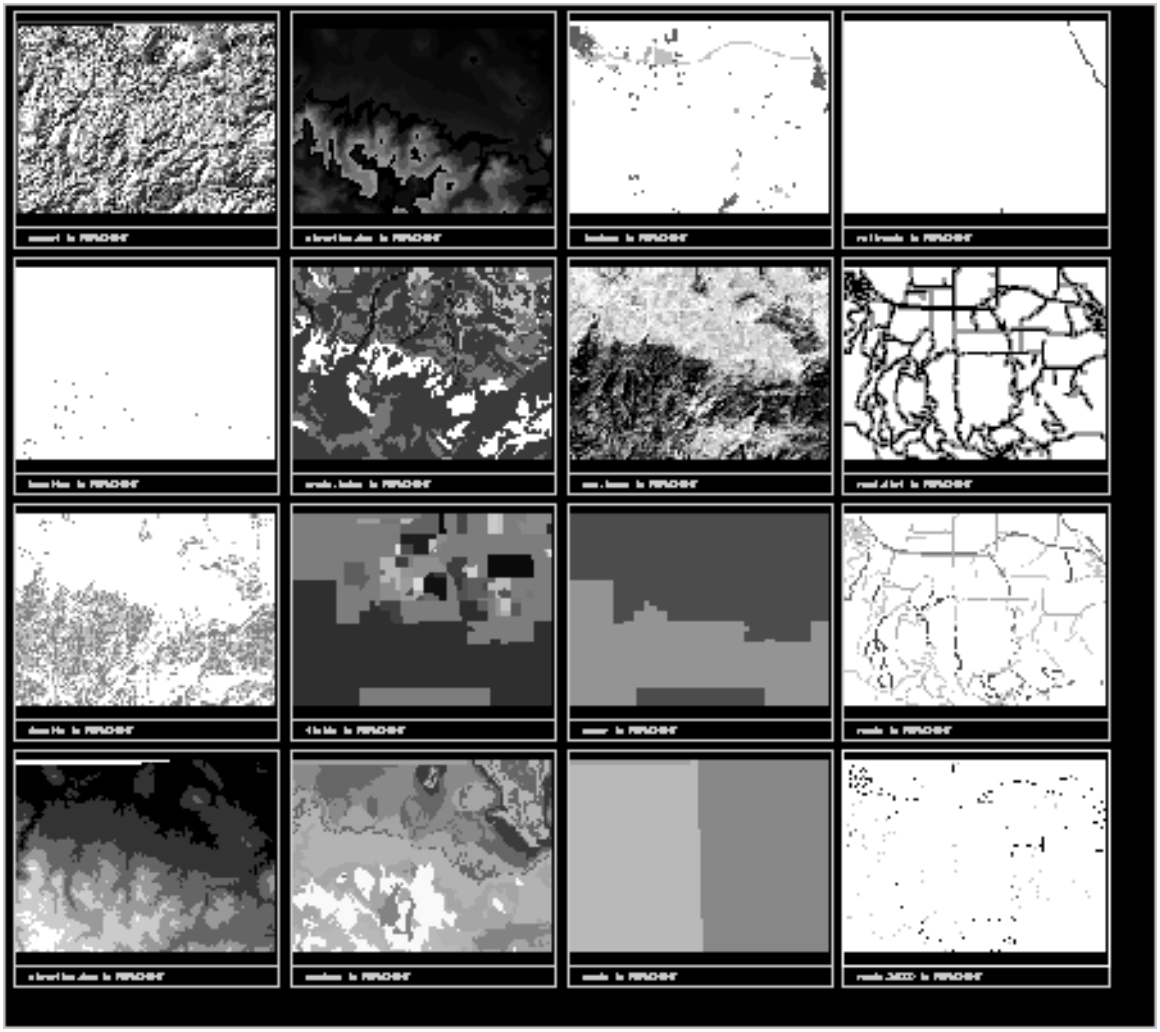


Figure 2: Sample maps from the Spearfish Data Base

- PC 386's and 486's (with Microport or SCO UNIX)
- Digital Equipment Corporation (DEC) workstations
- Data General's Aviion workstation
- Apollo workstations
- Hewlett Packard 9000
- IBM PS-2 (AIX)
- IBM 6000 workstations
- Tektronix workstations
- AT&T 3B2 workstation

Each port is supported by a different software house and set of distributors. Contact the *GRASS Information Center* (see section 2.1) for the latest list of distributors and hardware configuration options.

4 Why Does GRASS Exist?

4.1 Original Motivations

GRASS was born in the early 80's to address needs at an Army environmental office at Fort Hood, Texas (See Table 2). Budgets were tight and talent was available. A VAX 11/780 computer was available (funded via other projects), but there were no UNIX based geographical information systems commercially available at the time. Graduate student programmers took over and created the beginnings of GRASS in the VAX's UNIX environment. When Sun Microcomputers offerred their first workstations with color monitors (Sun/1), GRASS development was moved to the workstation environment where it still thrives today.

4.2 UNIX Based GIS

GRASS was initially developed as an unnamed series of demonstrations operating on a Cromemco powered by a Z-80 cpu. Researchers at the Army Corps of Engineer's Construction Engineering Research Laboratory (CERL) were trying to demonstrate the concept of geographical information systems to the Corps. Enough funds were originally generated to do smalonly l pilot systems on available computers. The GRASS fore-runner was a small system running on a VAX 11/780 that powered black and white graphics terminals over 300-baud modems. This system was funded by the Environmental Office at Fort Hood, Texas and was called the Fort Hood Information System (FHIS). Other environmental offices became interested and the more general purpose GRASS was born. Funding was still insufficient to purchase standard geographical information systems available at that time. Members of the National Park Service that now employ GRASS warned CERL researchers against putting in the required 10 man-years of programming. But, CERL did not gather enough funds to purchase hardware to run available software. CERL had hardware and expertise to operate UNIX machines and did the initial GRASS work on a VAX running UNIX. Within a year the fledgling GRASS was ported to an early SUN-1. So far as was known, this was the first GIS to operate in UNIX.

Year	Name	Additions to the System	Users
1982	FHIS	Single program, single location, raster database. Development and operation on UNIX based VAX 11/780. Digitizing on separate computer. Graphics on B/W monitor and B/W dot-matrix printer. User accessed remote via 1200 baud modem. Digitizing via polygon method.	Fort Hood, Texas
1983	IGIS	Multi-program accessed via menu, adapt to different data. SUN-1 microcomputer workstation. User operated on-site. Color display.	Fort McClellan, Alabama
1984	GRASS	Development on MASSCOMP MCS-560 and SUN-1. Digitizing via arc-node method. First color hardcopy output. 20 different programs.	Ft. Lewis, Central Washington University, National Guard Headquarters, data for 13 sites total.
1985	GRASS 1.0	First annual GRASS user group meeting - Fort Hood, Texas. First open distribution. Started inter-site electronic mail communication. Central Washington University offered first classes.	About 20 sites.
1986	GRASS 1.1	Run-length encoded raster files. Multi-byte grid-cell files supported by some programs. Image analysis as GRASS sub-system (runs only on MASSCOMP). Receives *Exemplary Systems in Government* from URISA. Interagency steering committee established.	Distribution to several dozen sites.
1987	GRASS 2.0	First programmer documentation. Some sites begin doing GRASS enhancements. GRASS distributed by Institute for Technology Development and DBA Systems. Rework of database. Adopt USGS digital line graph (DLG-3) format for vector data. Digitizing package for users. Ported to about seven different manufacturers machines. Army steering committee established.	Distribution to several hundred sites. Pilot tested by Soil Conservation Service. Adopted by National Parks Service.
1988	GRASS 3.0	Image analysis reworked into GRASS; runs on all GRASS machines. Raster files can contain from 1-4 bytes per cell. DLG dropped in favor of an internal vector format. Completely new digitizing package for user. Full assortment of command-line driven programs. Approximately 170 different programs.	Distribution to about 1000 sites.
1989	GRASS 3.1	Many new commands added. Significantly improved digitizing.	
1991	GRASS 4.0	This version realized the design goals set in 1986. Major additions include a standard command line parser for virtually all commands, substantial support for latitude-longitude based data, complete reorganization of command names, significantly better documentation, and performance enhancements.	Distribution to every household.

Table 2: GRASS History

4.3 Easy to Distribute

A major strength of GRASS is that it is very easy to distribute. Anyone who wants to cut a tape or copy GRASS across the Internet may do so without violating copyright requirement. Sites developing software using GRASS libraries or programs can freely copy GRASS as much as desired. For this reason, government agencies and universities that support large user communities find GRASS especially attractive.

4.4 Market Stimulus

GRASS stimulates the GIS market in several ways.

- Provides basic research. GRASS is a scientific research and development environment where programs and algorithms with unproven market potential can be tested. Numerous journal articles have been published that discuss research done both with and on GRASS. Techniques proven in the GRASS environment may become adapted by the commercial GIS marketplace.
- Brings new users to the GIS marketplace. Many potential users are in situations where the purchase of a commercial GIS package is not possible. This can be due to a lack of funding, too much red-tape, unfamiliarity with the technology, or insufficient staffing to make a large purchase cost effective. An excellent public-domain system provides an alternative to such groups. As these sites get involved with the technology they most certainly will turn to the commercial marketplace to purchase data, computers, peripherals, expertise, and advice. GRASS facilitates a growing number of commercial vendors of hardware, software, and expertise.

4.5 Great for Sites With Talent

Sites that have more money than talent and man-power often find that the purchase of a commercial GIS provides them with an out-of-house support staff. Sites that have significant talent, but less purchasing power consider GRASS to be a wonderful option. There are several reasons for this. First, the purchase of a commercial GIS is often an initiates a long-term relationship with the GIS vendor. Talented sites generally underutilize such relationships which often take the form of phone support, on-site debugging, on-site classes and training, custom product design and development and the continual presence of a friendly assistant. Sites requiring only a powerful GIS environment will find GRASS an extremely attractive option.

4.6 Huge, Friendly User Community

In lieu of a single software vendor that provides all the services a GIS site might desire, GRASS comes with a huge, friendly user community. The community meets en-mass once a year to exchange ideas, software, and presentations. Recognizing their co-dependence, the group tends to be very helpful to those with questions. The commercial folks involved with GRASS are highly enthusiastic about the product and offer services at nominal charges. Numerous Universities and commercial training sites across the world offer GRASS classes at very low cost. Information on the user community and commercial offerings is maintained by the *GRASS Information Center*.

4.7 GRASS Versions and History

GRASS can be appreciated through a cursory review of its history. It grew along with a user community new to GIS. A majority of its strength resides in its close development ties with such a user community. This development resulted in the history outlined in Table 2. GRASS 1.0 was in a class by itself. It was distinguished by a a user interface that allowed users to do something useful with the GIS in a very short time. This attracted a number of new users. It was also cited as one of the main reasons for being given the URISA ESIG Award[2].

The U.S. Army Corps of Engineer's research community awarded GRASS for it's excellence with the Army R&D Award.

Through the GRASS 4.0 release, GRASS software coordination, integration, testing, and distribution has been carried out by research staff at the Army Corps of Engineer's Construction Engineering Research Laboratory.

Beginning with GRASS 4.1, the GRASS Inter-Agency Software Coordination Center will be responsible for GRASS releases. This staff will likely be sited at USACERL. The office will be jointly funded by the major agencies doing GRASS system design and development.

5 Programs

This section introduces the GRASS 4.0 commands. For a very short listing and description of each command, refer to Appendix A on page 19. GRASS commands come in several flavors. Command names have the basic format: lower case letter, followed by a period (.), followed by a description string:

Command Prefix	Command Description
g.	General data (file) management commands
d.	Display (monitor) graphics
p.	Paint paper graphics
r.	Raster data manipulation and analysis
v.	Vector data manipulation and analysis
s.	Site data manipulation and analysis
i.	Imagery (multi-spectral) data manipulation and analysis
m.	Manipulation of external data

The naming convention was chosen to get a good balance in meeting the following objectives:

Non-conflict — Because GRASS commands are used in conjunction with other UNIX commands, it is important that GRASS commands not get confused with other programs. This requirement is met with the convention of prefixing each command with a letter and a period.

Separate raster, vector, etc. series — GRASS operations work with raster, vector, multi-spectral, and site data types. It is important that the user know what data type is being manipulated. It is also important that operation names be available for each data type. For example, by renaming the vector digitizing program from "digit" to "v.digit", there is room for a "r.digit" or "s.digit" program to be introduced in a later release.

[2]The Urban and Regional Information System Association's Exemplary Systems in Government (URISA ESIG) Award. This is presented annually to especially significant government developed software, and ws awarded to USACERL developers (Goran, Shapiro, and Westervelt) in 1987.

Self-explanatory — Program names were chosen to be as self descriptive as possible without being verbose.

Easy to type — Names needed to be easy to type. The upper case letters of GRASS 3.0 were apparently harder for users (in general) to enter than the corresponding lower case and period combination.

The display commands (d.) manage the GRASS monitor. They manage the monitor operations and provide commands that render all data types on the display graphics monitor:

- 2-D display
- 3-D display (Figure 3)
- Profiles
- Histograms
- Pie Charts
- Graphs
- Screen dumps to hard copy

Raster analysis commands all begin with r. and are the oldest of GRASS data analysis commands. These operations provide the traditional raster GIS operation primitives:

- Overlay (boolean, weighted, cellophane, rule base)
- Full mathematical functions
- Filters
- Proximity analysis
- Measurements (length, area, volume, counts)
- Clustering
- Import/Export
- Line-of-sight
- Cost analysis
- Transformation to vector
- Rotation
- Patching
- Reclassification
- Network flow analysis
- Thinning
- Elevation transformations

Perhaps the fastest developing set of GRASS capabilities is the vector GIS command set. Operations include the following:

- Digitizing
- Editing
- Labeling
- Measurements
- Import/Export
- Full topological linking
- Display
- Patching
- Transformation to raster
- Grid generation
- Contour labeling

Imagery commands form a powerful set of tools which manipulate multi-spectral images. GRASS users have made good use of Landsat, SPOT, and high-altitude NHAP images using this software. This suite of programs is continuing to grow and now contains:

- Tape handling for multiple formats
- Scene and sup-scene extraction
- Color enhancements
- Image registration
- Terrain correction
- Automated classification
- User directed classification
- Filtering
- Histograms
- Full integration with raster operations

Some geographical information can be stored as site data. The s. commands provide analysis capabilities for such data. These provide the following types of operations:

- Surface generation (interpolation)
- Statistical comparisons
- Reclassification
- Transformation to raster and vector

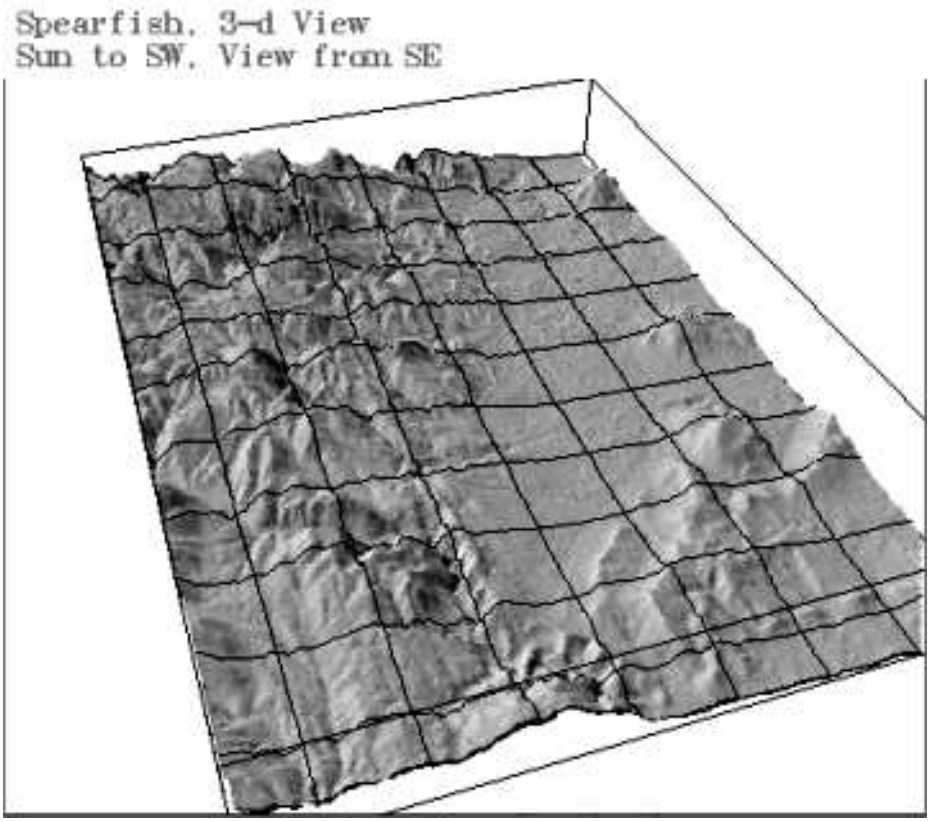


Figure 3: Sample 3-d Display

- Coordinate registration
- Data base searches and retrievals
- Proximity analysis
- Import/Export to the S statistical package

During the process of importing data into GRASS it is often necessary to resample data in various ways. The m. commands in provide mechanisms for doing some types of data resampling. These are most frequently used to assist in importing data into GRASS.

6 GRASS Design and Implementation

The design of GRASS has been quite successful because of sound guidelines established early in its development. This section looks at the key design and implementation policies currently embodied in GRASS and then looks at guidelines that may direct future work. First, a look at the current design:

UNIX and C — GRASS was to be developed for powerful UNIX workstations. The standard language on such machines was (and is) C.

Modular design — GRASS demonstrates a highly modular design allowing easy module upgrades. There are hundreds of subroutines in about 10 basic libraries, which support almost 300 different programs.

Portable — GRASS development aimed at generating capabilities that would compile on a wide variety of UNIX workstations.

Workstations — Programs designed to run on individual color UNIX workstations.

Integrated GIS and image processing — Traditional image processing programs enhanced with traditional raster GIS functions (and vice-versa). GRASS is designed to provide both capabilities in a manner that uses the same subroutines and data bases interchangeably.

Integrated raster and vector — Some spatial operations are better performed in vector, others in raster. Most user communities require both.

Multiple user interfaces — Geographical information systems address the needs of a diverse community of users. It is very important that different interfaces be available to different users. GRASS is a powerful GIS for geographical information specialists (Section 2.3.2, page 6.) It is also a software development environment (Section 2.3.3, page 6). Lots of very easy user interfaces have been developed that are location, project, and user specific.

Mixed raster resolutions — GRASS may be unique in its ability to resample raster data at run-time into a user selected region and resolution. GRASS raster data bases are stored with their original resolution and map placement. Most all similar systems require that data be resampled into a fixed grid during the data import process. GRASS stores the raw data and resamples as needed.

Assorted temporary data storage formats — GRASS is a potpourri of analysis functions. Some access the data in a regular manner - grid-cell after grid-cell in row after row. Others require the simultaneous access of data spanning several rows of data (e.g. Satellite data resampling). Some access the data randomly within small regions (e.g. line-of-sight and proximity analyses). Some can be highly random. Methods exist to temporarily load the data into formats that make access of each approach efficient.

Future design and development guidelines will likely implement the following requirements. Studies are now underway to determine what should be implemented and how.

C++ — C++ has become a standard object-oriented programming language. This language offers easy access to capabilities that are inherently difficult in C. Future GRASS development must seriously consider C++ for the sake of improving program development.

Multiple-attributes — GRASS raster and vector files have the capability only to access single strings of information for each map entity. This is very limiting and has forced the user community to store multi-attribute information in RDBMSs operated in parallel with GRASS. This approach requires users to pay special attention to the management of two separate data areas. In some cases user communities have addressed this difficulty by gluing GRASS and RDBMSs together within shell scripts that maintain the integrity between information in the two systems.

Multiple map management — Some GRASS sites have developed data bases with hundreds and even thousands of digital maps. The management of such data needs to become better automated.

Data access via network servers — GRASS was not initially designed to operate on networks. Though modifications have made such operation possible, GRASS must be thoroughly reviewed to more adequately function on networks. GRASS will likely be reworked so that software servers on machines with prodigious data storage parcel out data to GRASS systems running on local and remote workstations through UNIX sockets.

Efficient sub-map region queries — Individual GRASS data maps can be arbitrarily large. This is a problem, especially for programs that want to read in entire maps. Raster data needs to be accessible from these large maps without sacrificing efficiency.

Manage permissions — GRASS users are interested in sharing map data locations, maps, and map records. GRASS now allows only the owner of a data location write access to the location. A user can not run GRASS siumltaneously on different computers that share the GRASS data base. This must be relaxed.

Mix spatial and DBMS queries — GRASS provides a spatial analysis capability. RDBMSs provide query capabilities on map attribute information. Some queries, such as locating all sites withing 1 km of a sinuous road contain gold and silver, are best addressed if the spatial and RDBMS operations can occur simultaneously.

Floating point data — Currently only integer data in GRASS raster maps is possible (1, 2, 3, or 4 bytes). This must be relaxed to accommodate floats, doubles, structures, subroutine pointers, etc.

3-D data base — Future GRASS implementations will be called on to perform analyses on 3-D data bases.

7 GRASS Research and Development

GRASS is based on years of design and development, most of which was accomplished under research projects. Such research continues to evolve, resulting in the promise of more GRASS modules. Some of the projects planned for the next five years include the following:

X Interface — The UNIX community is standardizing their graphics efforts in a product called X-windows. This environment makes it possible to write portable code across a wide variety of UNIX hardware platforms providing sophisticated data visualization and user interfaces. Several efforts are underway to bring graphical user interfaces to GRASS.

Image Segmentation — New algorithms are being developed to identify homogeneous regions in images.

Raster editor — Two independent efforts are working toward a raster editing capability operating under X-windows.

Garden links — Numerous efforts are underway to link GRASS more closely with other systems.

Latitude-Longitude — Several significant research efforts will be undertaken to enable more of the GRASS capabilities to work directly with latitude-longitude data bases. Of particular interest will be modifications to allow digitizing of various projections.

Visualization — As real-time 3-D image rendering becomes more commonplace, the rapid rendering of GRASS data will become necessary.

Floating-Point Data — The GRASS integer data bases (for raster) limit the modeling capabilities of GRASS. The data structures must be updated to provide for not only the storage of floating point values, but imaginary numbers, structures and arrays, rules, function pointers, etc.

Parallel processing — Within a very few years, parallel processing hardware will become standard equipment within workstations. GRASS can benefit dramatically from such equipment which will required re–thinking the data bases and the software.

Modeling — GRASS data bases can support modeling of information as it moves and changes across the landscape. Such operations require large amounts of processing time and will work best on future parallel processing environments.

Fundamental DBMS-GRASS links — Data base management software has been linked with GRASS in numerous configurations. It is appropriate however to re–think the internal management of GRASS data using object-oriented and relational data base technology.

Full vector GIS — GRASS must provide its user community with a more complete set of standard vector operations.

Neural Networks — Neural networks are being evaluated for incorporation into GRASS analyses. These might enable a user to train the computer to identify sites with similar features to known sites.

Automated Digitizing — The development of data continues to be the most expensive component of any working geographical information system. Users would like to minimize the cost of turning paper maps into digital data layers. Advances are being made in rendering scanned map images into vector representations. Associated R&D is looking at automatically labelling and aggregating features of scanned images.

Contouring — Mathematical solutions for efficiently generating elevation maps from sparse information are being investigated for inclusion in GRASS.

A GRASS Commands

GENERAL GRASS DATA MANAGEMENT COMMANDS

g.access	Controls user access to the current GRASS mapset.
g.ask	Prompts the user for the names of GRASS data base files.
g.copy	Copies available data files in the user's current mapset search path and location to the appropriate element directories under the user's current mapset.
g.filename	Prints GRASS data base file names.
g.findfile	Searches for GRASS data base files and sets variables for the shell.
g.gisenv	Outputs the user's current GRASS variable settings.
g.help	GRASS help facility.
g.list	Lists available GRASS data base files of the specified data type to standard output.
g.manual	Accesses GRASS reference manual entries.
g.mapsets	Modifies the user's current mapset search path, affecting the user's access to data existing under the other GRASS mapsets in the current location.
g.region	Program to manage the boundary definitions for the geographic region.
g.remove	Remove data base element files from the user's current mapset.
g.rename	Rename data base element files in the user's current mapset.
g.version	Outputs the GRASS version number and date.

HARDCOPY GRAPHICS COMMANDS

p.chart	- Prints a color chart for the printer currently selected.
p.colors	- Allows the user to modify a color table for a raster map layer, assigning colors to the categories in the raster map layer based on printer color numbers (instead of red, green, blue percentages).
p.icons	- Allows the user to create and maintain icons which are used by the p.map command to depict sites.
p.labels	- Creates and modifies labels for hardcopy maps.
p.map	- Command language interface to color hardcopy and graphics monitor output.
p.screen	- Prints a graphics monitor display image file that has been saved by d.savescreen.
p.select	- Selects a device (printer) for GRASS hardcopy output. Outputs can also be previewed on the graphics display monitor.

DISPLAY GRAPHICS COMMANDS

d.3d	Three-dimensional raster map display program.
d.ask	Prompts the user to select a GRASS data base file from among files displayed in a menu on the graphics monitor.
d.colormode	Allows the user to establish whether a map will be displayed using its own color table or the fixed color table of the graphics monitor.
d.colors	Allows the user to interactively change the color table of a raster map layer displayed on the graphics monitor.
d.colortable	Display the color table associated with a raster map layer.

d.display	Interactive program to display textual, symbolic, raster, vector, and site data in the active frame on the user's graphics monitor.
d.erase	Erases the active display frame on the graphics monitor.
d.font	Selects text fonts to be used for text display on the graphics monitor.
d.frame	Manages display frames on the graphics monitor.
d.geodesic	Displays a geodesic line, tracing the shortest distance between two geographic points along a great circle, in a latitude–longitude data set.
d.graph	Program for generating and displaying simple graphics to the graphics display monitor.
d.grid	Overlays a user-specified grid in the active display frame on the graphics monitor.
d.his	Produces and displays a raster map layer combining hue, intensity, and saturation (his) values from user-specified input raster map layers.
d.histogram	Plots a histogram in the form of a pie or bar chart for a user-specified raster file.
d.icons	Displays points, as icons, at user-defined locations in the active display frame on the graphics monitor.
d.label	Creates and places text labels in the active display frame on the graphics monitor.
d.labels	Create/edit GRASS label files for display on the graphics monitor.
d.legend	Displays a legend for a raster map layer in the active frame on the graphics monitor.
d.mapgraph	Generates and displays simple graphics on raster map layers drawn in the active graphics monitor display frame.
d.measure	Measures the lengths and areas of features drawn by the user in the active display frame on the graphics monitor.
d.menu	Creates and displays a menu within the active frame on the graphics monitor.
d.mon	Establishes and control use of a graphics display monitor.
d.paint.labels	Displays text labels formatted for use with GRASS paint (paint, p.map) output to the active frame on the graphics monitor.
d.points	Displays point graphics in the active frame on the graphics display monitor.
d.profile	Displays profiles of a user-specified raster map layer in the active frame on the user's graphics monitor.
d.rast	Displays and overlays raster map layers in the active display frame on the graphics monitor.
d.rgb	Allows multiple maps to be overlaid in the active display frame on the graphics monitor.
d.rhumbline	Displays the rhumbline joining two user-specified points in the active frame on the user's graphics monitor.
d.scale	Displays a map scale and north arrow relevant to a user's current geographic region setting in the active display frame on the graphics monitor.
d.sites	Displays site markers in the active display frame on the graphics monitor.
d.text	Draws text in the active display frame on the graphics monitor.
d.title	Outputs a title for a raster map layer in a form suitable for display by d.text.
d.vect	Displays GRASS vector data in the active frame on the

	graphics monitor.
d.vect.dlg	Displays a vector USGS digital line graph (DLG-3) file in the active frame on the graphics monitor.
d.what.rast	Allows the user to interactively query the category contents of multiple raster map layers at user-specified locations within the current geographic region.
d.what.vect	Allows the user to interactively query the category contents of multiple (binary) vector map layers at user-selected locations within the current geographic region.
d.where	Identifies the geographic coordinates associated with point locations in the active frame on the graphics monitor.
d.zoom	Allows the user to change the current geographic region settings interactively, using a mouse.

RASTER DATA MANIPULATION COMMANDS

r.average	Finds the average of values in a values map within polygons of a user-specified base map.
r.basins.fill	Generates a raster map layer showing watershed sub–basins.
r.binfer	Bayesian expert system development function.
r.buffer	Creates a raster map layer showing buffer zones surrounding cells that contain non-zero category values.
r.cats	Prints category values and labels associated with user-specified raster map layers.
r.clump	Recategorizes data in a raster map layer by grouping cells that form physically discrete areas into unique categories.
r.coin	Tabulates the mutual occurrence (coincidence) of categories for two raster map layers.
r.colors	Creates/Modifies the color table associated with a raster map layer.
r.combine	Allows category values from several raster map layers to be combined.
r.compress	Compresses and decompresses raster files.
r.cost	Outputs a raster map layer showing the cumulative cost of moving between different geographic locations on an input raster map layer whose cell category values represent cost.
r.covar	Outputs a covariance/correlation matrix for user-specified raster map layer(s).
r.cross	Creates a cross product of the category values from multiple raster map layers.
r.describe	Prints terse list of category values found in a raster map layer.
r.drain	Traces a flow through an elevation model on a raster map layer.
r.grow	Generates an output raster map layer with contiguous areas grown by one cell (pixel).
r.in.ascii	Converts an ASCII text file into a raster map layer.
r.in.ll	Creates a raster file from data in latitude–longitude coordinates.
r.infer	Outputs a raster map layer whose category values represent the application of user-specified criteria (rules statements) to other raster map layers' category values.
r.info	Outputs basic information about a user-specified raster map layer.
r.line	Creates a new GRASS vector (digit) file by extracting linear features from a thinned raster file.

r.los	Line-of-sight raster analysis program.
r.mapcalc	Raster map layer data calculator.
r.mask	Establishes/removes the current working mask.
r.mfilter	Raster file matrix filter.
r.neighbors	Makes each cell category value a function of the category values assigned to the cells around it, and stores new cell values in an output raster map layer.
r.out.ascii	Converts a raster map layer into an ASCII text file.
r.patch	Creates a composite raster map layer by using known category values from one (or more) map layer(s) to fill in areas of "no data" in another map layer.
r.poly	Extracts area edges from a raster map layer and converts data to vector format.
r.profile	Plots profiles of a GRASS raster map layer.
r.random	Creates a raster map layer and sites list containing randomly located sites.
r.reclass	Creates a new map layer whose category values are based upon the user's reclassification of categories in an existing raster map layer.
r.report	Reports statistics for raster map layers.
r.resample	GRASS raster map layer data resampling capability.
r.rescale	Rescales the range of category values in a raster map layer.
r.slope.aspect	Generates raster map layers of slope and aspect from a raster map layer of true elevation values.
r.stats	Generates area statistics for raster map layers.
r.support	Allows the user to create and/or modify raster map layer support files.
r.surf.contour	Surface generation program.
r.surf.idw	Surface generation program, from raster data.
r.thin	Thins non-zero cells that denote linear features in a raster file.
r.traj	Ballistic trajectory model.
r.transect	Provides transect information across a raster map.
r.volume	Sums cell values within clumps and calculates volumes and centroids of patches or clumps.
r.watershed	Watershed basin analysis program.
r.weight	Raster map overlay function.
r.what	Queries raster map layers on their category values and category labels.

VECTOR DATA MANIPULATION COMMANDS

v.cadlabel	Attaches labels to vector contour lines that have been imported to GRASS from DXF format.
v.digit	A menu-driven, highly interactive map development program used for vector digitizing, editing, labeling, and converting vector data to raster format.
v.import	Converts ASCII Digital Line Graph (DLG) files, binary DLG files, and ASCII vector files into binary vector files and creates the needed vector support files.
v.in.arc	Converts data in ARC/INFO format to GRASS's vector format, and stores output in the user's current GRASS mapset.
v.in.ascii	Converts ASCII vector map layers into binary vector map layers.

v.in.dlg	Converts binary DLG (BDLG) file to binary vector (dig) file.
v.in.dxf	DXF format to GRASS vector format conversion program.
v.mkgrid	Creates a (binary) GRASS vector map of a user-defined grid.
v.mkquads	Creates a GRASS vector map layer and/or sites list and/or geographic region definition file for a USGS 7.5-minute quadrangle.
v.out.arc	Converts GRASS vector files to ARC/INFO's "Generate" file format.
v.out.ascii	Converts binary vector map layers into ASCII vector map layers.
v.out.dlg	Converts binary GRASS vector data to binary DLG-3 Optional format.
v.out.dxf	GRASS vector format to DXF format conversion program.
v.out.moss	Converts GRASS site, line, or area data into MOSS import format. (Data Export Program)
v.patch	Creates a new binary vector map layer by combining other binary vector map layers.
v.prune	Prunes points from binary GRASS vector data files.
v.spag	Processes a Spaghetti-digitized vector file.
v.stats	Displays information about a user-specified GRASS vector file.
v.support	Creates GRASS support files for (binary) GRASS vector data.
v.to.rast	Converts a map layer in (binary) GRASS vector format to GRASS raster format.
v.to.sites	Converts a GRASS vector file to a GRASS site-lists file.
v.transform	Transforms an ASCII vector map layer from one coordinate system into another coordinate system.
v.trim	Trims small spurs and removes excessive nodes from a GRASS vector (digit) file.

MULTISPECTRAL DATA MANIPULATION COMMANDS

i.build.blk	Interactive ortho-photo rectification program.
i.camera	An imagery function that establishes a camera reference file for an imagery group.
i.cca	Canonical components analysis (cca) program for image processing.
i.class	An imagery function that generates spectral signatures for an image by allowing the user to outline regions of interest. The resulting signature file can be used as input for i.maxlik or as a seed signature file for i.cluster.
i.cluster	An imagery function that generates spectral signatures for land cover types in an image using a clustering algorithm. The resulting signature file is used as input for i.maxlik.
i.colors	An imagery function that creates colors for imagery groups.
i.composite	An imagery function that creates a color composite image from three band files specified by the user.
i.fft	Fast Fourier Transform (FFT) for image processing.
i.grey.scale	An imagery function that assigns a histogram-contrast-stretched grey-scale color table to a raster map layer.
i.group	An imagery function that creates and edits groups and subgroups of (raster) imagery files.
i.his.rgb	Hue-intensity-saturation (his) to red-green-blue (rgb) raster map color transformation function.
i.ifft	Inverse Fast Fourier Transform (ifft) for image processing.
i.maxlik	An imagery function that classifies the cell spectral reflectances

	in imagery data based on the spectral signature information generated in i.cluster.
i.mod.camera	An imagery function that creates or modifies entries in a camera reference file.
i.pca	Principal components analysis (pca) program for image processing.
i.points	An imagery function that enables the user to mark coordinate system points on an image to be rectified and then to input the coordinates of each point for creation of a coordinate transformation matrix. The transformation matrix is needed as input for the GRASS program i.rectify.
i.rectify	An imagery function that rectifies an image by computing a coordinate transformation for each cell (pixel) in the image using the transformation coefficient matrix created by the GRASS program i.points.
i.rectify.blk	An imagery function that ortho-rectifies an imagery group file.
i.rgb.his	Red-green-blue (rgb) to hue-intensity-saturation (his) function for image processing.
i.tape.mss	An imagery function that extracts Multispectral Scanner (MSS) imagery data from half-inch tape.
i.tape.mss.h	An imagery function that extracts header information from Landsat Multispectral Scanner (MSS) imagery data stored on half-inch tape.
i.tape.other	An imagery function that extracts scanned aerial imagery (NHAP, etc.) and SPOT imagery from half-inch magnetic tape.
i.tape.tm	An imagery function that extracts LANDSAT Thematic Mapper (TM) imagery from half-inch magnetic tape.
i.target	An imagery function that establishes a GRASS target LOCATION for an imagery group.
i.zc	Zero-crossing "edge detection" raster function for image processing.

SITE DATA MANIPULATION COMMANDS

s.db	RIM data base management/query interface for GRASS.
s.in.ascii	Converts a GRASS site-lists file in ASCII format to binary format.
s.menu	Accesses and manipulates GRASS sites data.
s.out.ascii	Converts a GRASS site-lists file in binary format to ASCII format.
s.surf.idw	Surface generation program, from sites data.

EXTERNAL DATA MANIPULATION COMMANDS

m.dem.extract	Extracts USGS Digital Elevation Model (DEM) data from 1/2-inch magnetic tape.
m.examine.tape	Provides a description of the files on a 1/2-inch magnetic tape.
m.ll2u	Converts geographic (latitude–longitude) coordinates to Universal Transverse Mercator (UTM) coordinates.
m.region.ll	Converts Universal Transverse Mercator (UTM) coordinates falling within the current geographic region from UTM coordinates to geographic (latitude/longitude) coordinates.
m.rot90	Rotates elevation data extracted by either m.dted.extract or m.dmaUSGSread.

m.u2ll	Converts Universal Transverse Mercator (UTM) coordinates to geographic (latitude–longitude) coordinates.

B OVERHEADS