# About This Manual

## Objectives of This Manual

This manual describes CM/AVS. Working in conjunction with the Application Visualization System (AVS), CM/AVS provides a graphic programming environment for building distributed visualization applications. This manual tells how to build applications that include computation on a CM-5 system, and how to create your own CM/AVS modules.

## Intended Audience

This manual is intended for

- Users who are familiar with the AVS product and who want to visualize data on a CM-5 system. You must also be familiar with using a CM-5.
- Application developers who want to write modules that are compatible with CM/AVS. You should be an experienced C or Fortran programmer, knowledgeable about AVS, and familiar with C* or CM Fortran and using a CM-5.

## Revision Information

This is a new manual.

## Organization of This Manual

This manual contains the following chapters:

**Chapter 1 Introduction**
An overview of CM/AVS concepts and a list of installed components.

## Related Documents

The following document contains information concerning the hardware and software requirements and installation of CM/AVS:

- *CM/AVS Release Notes* for Version 1.0.

You should have the complete AVS document set. The following manuals are required:

- *AVS User's Guide*
  An introduction to AVS. To use CM/AVS effectively, you must be familiar with the concepts introduced in this manual.

- *AVS Developer's Guide*
  How to write AVS modules.

- *AVS Module Reference*
  Detailed descriptions of all the AVS modules.

- *AVS Tutorial Guide*
  A tutorial introduction to using AVS.
- *AVS Applications Guide*
  Information on using the Module Generator.

## Notation Conventions

The table below displays the notation conventions observed in this manual.

| Convention | Meaning |
|---|---|
| **`bold typewriter`** | UNIX and CM System Software commands, command options, and filenames, when they appear embedded in text. Also, programming language elements, such as keywords, operators, and function names, when they appear embedded in text. |
| `%` **`bold typewriter`**<br>`regular typewriter` | In interactive examples, user input is shown in **`bold typewriter`** and system output is shown in `regular typewriter` font. |
| `typewriter` | Code examples and code fragments. |
| *italics* | Argument names and placeholders in function and command formats. |