Semper 6 *Plus*

BEGINNERS' USER GUIDE

Synoptics

Trademarks:

Semper is a registered trademark of *Synoptics Limited*
IBM PC, PC/XT, PC/AT are trademarks of International Business Machines Inc
Silicon Graphics Personal IRIS 4D is a trademark of Silicon Graphics Inc
Sun is a registered trademark of Sun Microsystems Inc
VAX is a registered trademark of Digital Equipment Corporation
WordStar is a registered trademark of *MicroPro International Corporation*

Manual number: SEM009
Update number: 0
Printed: June 1990

Table OF CONTENTS

Overview

Tutorial 1: Getting Started

Tutorial 2: Finding out about the Semper Environment

Tutorial 3: Viewing your Pictures

Tutorial 4: Image Processing

Tutorial 5: Particle Analysis

Tutorial 6: Programming with Semper

Conclusion

Appendix A: Troubleshooting

Appendix B: Picture Types

Appendix C: Glossary

Overview

About this manual

This manual provides an introduction to Semper. It consists of six short tutorials that give you step-by-step instructions on how to use Semper to solve some basic image processing problems. If you are new to Semper, it is a good idea to work through these tutorials methodically.

What is Semper ?

This section attempts to list some of the characteristics of Semper, and to state clearly some of the things it is not.

- ***Semper includes a high level image processing language***

By this we mean that you can use Semper to perform image processing operations by typing commands in a special language: the Semper language. The language permits you to manipulate whole images with a single command; you need not concern yourself with the intricacies of the operations themselves, and can concentrate on your problem instead. For example, you may wish to magnify an image by a factor of 3. Without Semper, you would have to write hundreds of lines of sophisticated computer programs to achieve this apparently simple task. With Semper, you need only type **magnify times 3**. This is why we say that the language is 'high level'; the commands are given at the level at which you think about your problem, not at the level of operations on individual pixels in the image. You need not be a computer programmer to use Semper.

- ***Semper's language is interpreted***

This term has a particular meaning when applied to computer languages; it means that the commands you give the computer can be decoded one by one and executed immediately. The 'BASIC' language is one which can be interpreted; in contrast, a program written in 'C' or 'Fortran' must be passed as a whole through another program, a compiler, before it can be run. The advantage of an interpreter is that you can experiment and test ideas very quickly, trying out commands and techniques as you think of them. When you have found a solution to your problem, you can build the same Semper commands into a program which you or others can subsequently use.

- ***Semper includes a visual user interface***

As an alternative to typing commands, you can control Semper by means of a visual user interface. Such interfaces consist of objects such as menus, icons, and textfields, which are drawn on a screen and can be activated by pointing to them (with a mouse, for example). You can use interfaces which are provided by *Synoptics* (such as *Tutor*), or you may construct your own, to suit your own application or preferences.

- ***Semper is a complete operating environment***

When you start Semper on your computer, you enter a new operating environment. All the commands that you type (or enter through the visual user interface) are interpreted by Semper. In a sense, Semper replaces the computer's operating system with its own, which is dedicated to image processing. In this respect, when you are using Semper, your computer is hidden from you; any computer running Semper will respond in the same way to the same commands. This is why we say Semper is *portable*.

- ***Semper is not a library of subroutines***

Some image processing 'systems' are merely a collection of program subroutines written in a conventional computer language such as 'C', 'Fortran', or 'Pascal'. To use these libraries you need to know how to translate your image processing problem into subroutine calls, how to tailor the subroutines to work on your computer, and how to compile, link, and run the resulting program on your computer. All these complexities disappear when using Semper.

- ***Semper is not an applications package***

Semper is not dedicated to any particular problem, nor any particular application of image processing, nor even any particular field of image processing science. It is incapable of solving any particular problem without your instructions. Think of it as a toolkit, a pocket calculator, a framework in which to work. What Semper gives you is time; time to concentrate on your application, free from having to consider the details of your imaging and computer hardware.

- ***Semper is not "just a menu"***

Whilst we can provide a menu for Semper (such as *Tutor*), the capabilities of Semper extend far beyond "just a menu". You can extend or modify the menus, or even construct your own. Alternatively, you can control Semper through the almost unlimited flexibility of its command language.

How this manual is organised

This manual is organised into the following six tutorials:

- *Tutorial 1* — This tutorial describes how to start Semper and some basic image processing operations. (The ABC of Semper). It also describes how to capture a live image using a camera.
- *Tutorial 2* — This tutorial lets you take stock of your surroundings. It describes examining and storing pictures and the Semper environment.
- *Tutorial 3* — Tutorial 3 gives you a fresh viewpoint on your pictures – using Semper partitions, look-up tables, ramps, frames and overlays.
- *Tutorial 4* — This tutorial settles down to the serious business of image processing, detailing filters, fourier transforms and performing arithmetic on an image.
- *Tutorial 5* — Tutorial 5 sorts the wheat from the chaff. It describes particle analysis using Semper.
- *Tutorial 6* — Tutorial 6 describes programming with Semper. It details how to combine commands into macros and program files.

This manual also contains a concluding section which describes how to customize your own Semper working environment and three appendixes. *Appendix A, Troubleshooting* suggests solutions to common error situations. *Appendix B, Picture Types* describes how Semper stores and classifies pictures. *Appendix C, Glossary* provides a glossary of the technical terms that are used in this manual.

The conventions used in this manual

Semper commands are shown in the following font:

```
rotate angle pi/3
```

unless they are embedded in the text, in which case they are shown in **bold**.

Computer output is shown in bold, for example:

```
type age
21
```

where the variable *age* holds the value 21.

What else to read?

To find out more about Semper, its underlying concepts and features, read the manual:

Advanced User Guide

This provides a formal and comprehensive description of Semper.

If you would like a specific introduction to Semper's menu interface, read the manual:

Tutor User Guide

which contains a number of tutorials to help you learn about Semper's visual interface. Note that using *Tutor* is an effective way of learning about the range of features that Semper provides. An advanced guide to the menu interface is given in the manual:

User Interface Guide

This manual describes how to create your own menu interface to Semper and to extend existing menus.

All of the above manuals are contained in the *Semper 6 Guide.*

If you would like details of the Semper language, you can choose from the following manuals:

Command Summary	A summary of the commands that make up the Semper language.
*Quick Reference List (*in the *Semper 6 Guide)*	A list of Semper commands according to function, devoting a few lines to each command and providing examples.
Command Reference	A complete and comprehensive description of each command with many examples. It details command syntax, defaults and ranges.

There are also two further manuals, for specialist applications. Read the manual:

Fortran Programmers' Guide

if you want to write your own Fortran routines to run under Semper as new commands. This manual also explains how the system is organised internally. The following manual:

Implementation Guide

details how to install Semper in a new environment and the software interface that its primitive routines provide with the particular peripherals available locally.

New to Image Processing?

If you are new to the field of image processing, we recommend the following introductory text:

Digital Image Processing
by R.C.Gonzalez and Paul Wintz
published by Addison-Wesley, 1987
I.S.B.N. 0–201–11026–1

Tutorial 1

GETTING STARTED

❑ Starting Semper ❑ Selecting and displaying an image ❑ Magnifying, transposing and rotating an image ❑ Semper help ❑ Semper Syntax ❑ Semper Short Cuts ❑ Capturing a live image ❑ Stopping Semper

If you are the first person at your location to use Semper, please consult the appropriate *Semper 6 Installation Guide* for your machine.

Overview

This tutorial describes how to start the Semper tutorial software and provides some basic examples of image processing operations. How you start Semper depends on the type of machine that you are using. Instructions are given below for the following machines:

- IBM PC or compatible
- DEC MicroVAX/GPX and VAXstation
- Sun 3 or 4 workstation running *Sunview*
- Silicon Graphics IRIS 4D workstation
- Hewlett-Packard workstation running *X-windows*

Starting Semper on a PC

Switch on your PC. If the SEMPER6P directory is not the current directory, make it current by typing the following command:

```
CD \SEMPER6P
```

Now refer to the section *Starting the tutorial software* on page 1-3.

Starting Semper on a VAX

Log onto the VAX using your username and password. You should be able to start Semper directly from this point (see the section *Starting the tutorial software* on page 1-3), assuming that your System Manager has set up the Semper environment correctly, as detailed in *Notes for VAX System Managers* below.

Starting Semper on a VAX workstation running UIS/VWS display

Move the cursor into a blank part of the workstation screen and click the left mouse button. A window appears that contains a number of options. Select the option that reads:

Create VT220 window (with autologin)

by moving the cursor over it and clicking with the left mouse button.

This option creates a VT220 terminal window, where you are asked to login using your username. At the command line prompt you can start Semper (see the section *Starting the tutorial software* on page 1-3). This assumes that your System Manager has set up the Semper environment correctly, as detailed in *Notes for VAX System Managers* below.

Starting Semper on a VAX workstation running DECwindows

Login to the VAX workstation using your username and password and run up *DECwindows.* You should be able to start Semper directly from this point (see the section *Starting the tutorial software* on page 1-3), assuming that you have defined a local display. Consult your System Manager if you see an error message to the effect that a display is not available. These instructions also assume that your System Manager has set up the Semper environment correctly, as detailed in *Notes for VAX System Managers* below.

Notes for VAX System Managers

To ensure that VAX users can access Semper directly, you need to do two things:

1. Set up the logical name *semper62.*
2. Establish the command *semper* by a 'command set' command:
   semper62:semper.

These commands need to be in the *login.com* file of each Semper user.

Starting Semper on a Sun workstation

Login as user *semper* (or the name for the user account that was created at installation). Start up *SunView* by typing:

```
sunview
```

You will then see two windows *<<CONSOLE>>* and *cmdtool /bin/csh*. Move the cursor into the window called *cmdtool /bin/csh*.

If Semper is not the current directory, make it current by typing:

```
cd /usr/semper
```

You can now start Semper (see the section *Starting the tutorial software* below).

Starting Semper on a Silicon Graphics

Login to the 4SIGHT system as user *semper* (or the name for the user account that was created at installation). In a console or shell window, change the current directory to the Semper directory, if it is not current at start up. To do this type the command:

```
cd /usr/semper
```

You can now start Semper (see the section *Starting the tutorial software*).

Starting Semper on a Hewlett-Packard workstation

Login to the Hewlett-Packard workstation using your username and password and run up *X-windows*. You should be able to start Semper directly from this point (see the section *Starting the tutorial software* below), assuming that you have defined a local display. Consult your System Manager if you see an error message to the effect that a display is not available.

Starting the tutorial software

To start the Semper tutorial software, type the following command at your operating system prompt:

```
semper /run=tutorial
```

Ensure that the direction of the backslash in the above command is correct, otherwise you will run a Semper session rather than the tutorial software. Later, when you have gained in expertise and confidence, you may like to run a standard Semper session but for the purpose of these tutorials it is preferable to run the tutorial software. (See the *Conclusion* at the end of these tutorials for details of running a standard Semper session in your own customised environment).

If you have any problems when starting or running Semper, refer to *Appendix A: Troubleshooting.*

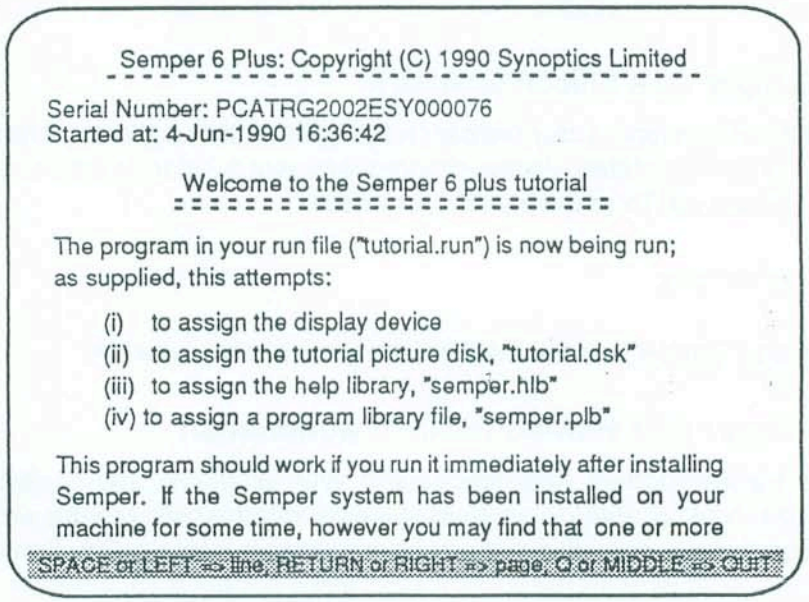
What do you see?

If you are using a system with a separate terminal and display monitor (**PC** and some **VAX** systems) Semper messages and commands are displayed on the terminal screen. The display monitor screen is blank for the display of images.

If you are using a workstation running a windowing system (VAX workstations, Sun, Silicon Graphics or Hewlett-Packard), you see two windows:

- a Semper 6 *Plus* display window (for image display)
- a terminal window (for entering Semper commands etc.)

The following is an example of the the text that you see when you run the tutorial software.



Press the space bar or the left mouse button to scroll through the printed information (do ***not*** press *<Q>* or the middle mouse button). At the end of the text, you see the following message on the screen:

```
Waiting - press any key or mouse button to continue
```

When you press a key or button, Semper assigns its devices (display, picture disc, help libraries etc.) Then you see the Semper prompt.

```
S$
```

This prompt appears whenever Semper is waiting for instructions. You can start typing Semper commands at this prompt. Note that you can type commands in either upper or lowercase letters and that you need to type a carriage return *<return>* (*<cr>* or *<enter>* on some keyboards) at the end of each command.

First steps

Before you start using Semper's image processing facilities, you may like to examine the contents of the tutorial picture disc. Type the following command at the Semper prompt:

```
examine all
```

Semper displays the following information about the pictures on your disc. (If your disc contents differ from those given below, it may be that one or more of the required files has been deleted. In this case, ask an existing Semper user to help).

```
Disc device 2: c:\semper6p\tutorial.dsk
   2001     Size 128, 128, 1  16.0kb   Image    Byte  wp
   Suzy
   2002     Size 128, 128, 1  16.0kb   Image    Byte  wp
   Crystal
   2003     Size 128, 138, 1  16.0kb   Image    Byte  wp
   Cell
```

The list above shows that there are three pictures on your tutorial picture disk which is disk device 2. (Devices are explained in the next tutorial, *Tutorial 2, Finding out about the Semper Environment*). The list shows picture 1 to be 128 points (pixels) square, occupying about 16 kilobytes of space and containing an image that is stored in byte form (that is, using a single 0–255 value for each picture point or pixel). The picture title is "Suzy". All of the pictures are write-protected (*wp*) so that you cannot accidentally alter or delete them. For details about how Semper stores and classifies pictures, refer to *Appendix B, Picture Types*.

A note about picture numbers

You can refer to a picture simply by its number, or by its device number and picture number. For example, you could refer to the above pictures simply as pictures **1**, **2** and **3**. By convention, Semper refers to the pictures as **2001**, **2002** and **2003** which is simply the device number multiplied by 100 plus the picture number. You can also refer to the above pictures as *device number : picture number*, for example, **2:1**, **2:2** and **2:3** or as **cd:1**, **cd:2** and **cd:3** (*cd* is a Semper variable that holds the current device number).
These different ways of referencing pictures give you complete freedom to perform operations on pictures between devices.

Now type the following command at the terminal:

```
erase=yes
```

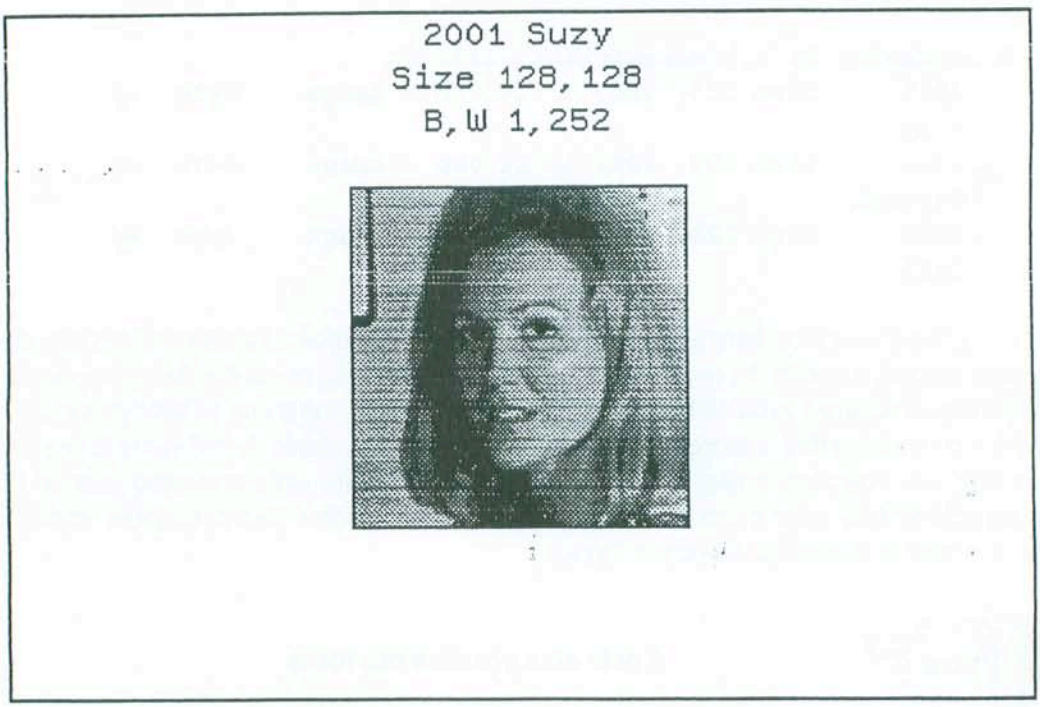
This command tells Semper to erase the current display picture area before displaying another picture. This means that when you are displaying and manipulating pictures in this tutorial you will start off with a "clean" display each time.

Displaying a picture

To see a test picture on the display screen or window, type the following command:

```
display 1
```

This command displays the image shown below.



The image is scaled to maximum contrast with a border around it and lettering at the top of the display. The lettering includes a picture number (2001), picture title ("Suzy"), dimensions (128 by 128 pixels) and the data range of the picture (B,W 1,252). The data range tells you how Semper represents the greyscale of the picture. In picture 1, the lowest valued pixel (1) is shown as black and the highest valued pixel (252) is shown as white.

You may also like to try the following commands, to see their effect on the test picture:

```
display 1 negated
display 1 size 75 top right
```

Magnifying an image

Semper provides a simple way to magnify an image. To do this, type the following command at the terminal:

```
magnify from 1 to display
```

Your image is re-displayed on the screen magnified by a factor of 2.

Note that it is only the image shown on the display that has changed. The image on disc remains the same. This is because Semper allows you to specify a *source* and an *output*. The source is the picture or device from which Semper reads its data, the output is the destination that holds the altered data. In this case, you have specified the *display* as the destination, but if you require a permanent record of your work you could have sent the output to another disc picture. For example, type the following sequence of commands:

```
magnify from 1 to 4
display 4
```

You will see that disc picture 4 holds a magnified version of disc picture 1. Type **examine all** to see the pictures that now exist on your disc. Note that if you do not explicitly specify a destination – a display or another picture – Semper overwrites the source picture with the magnified image.

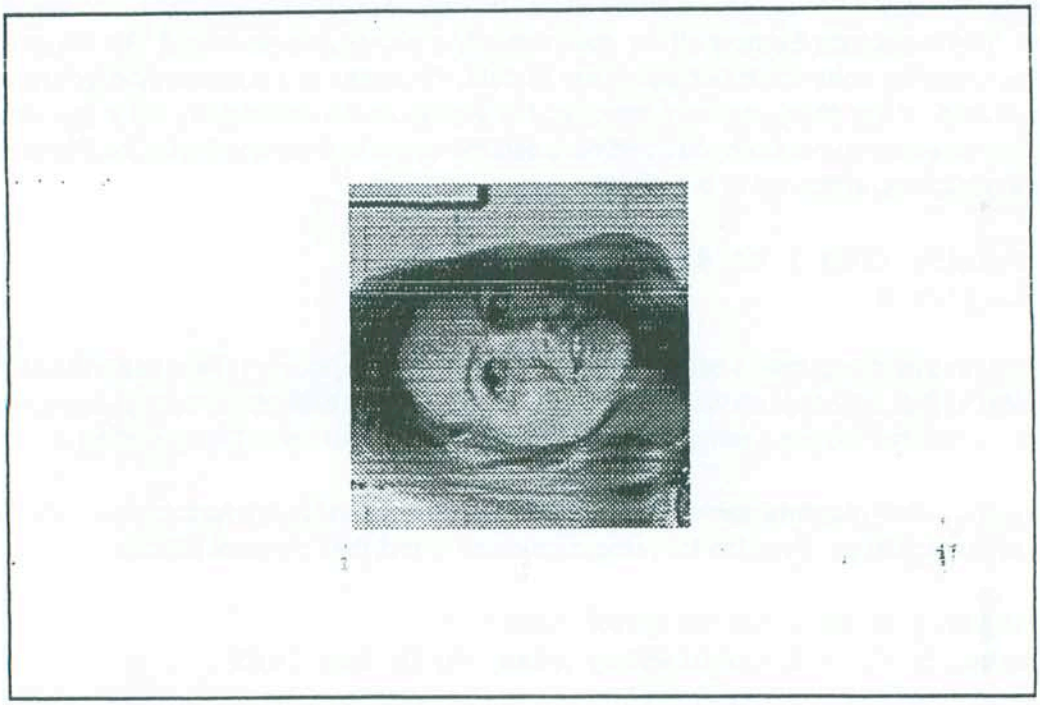
By building on the **magnify** command, you can specify the magnification factor, a subregion of the picture to magnify etc. Type the following commands to test their different effects:

```
magnify from 1 to display times 3
magnify from 1 to display size 75,75 top left
magnify from 1 to display times 4 repeating
```

Transposing an image

Semper allows you to transpose a picture, that is, to interchange the *x* and *y* axes so that they are reflected in the top left/bottom right diagonal. To see the effect of this on your test picture, type the following command:

```
transpose from 1 to display
```



Note that the **transpose** command requires a square source picture of factorisable size.

Rotating an image

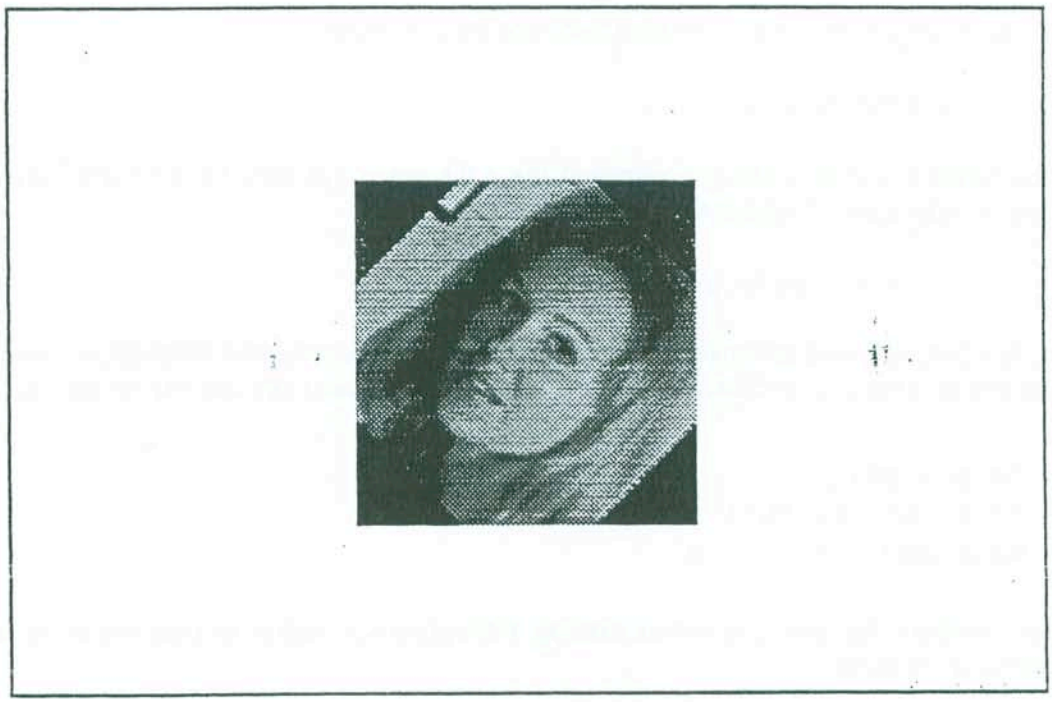
As a final exercise, you may like to try to rotate the test image. Type the following command:

```
rotate from 1 to display angle 0.785
```

Note that Semper only accepts angles that are specified in ***radians***, but there is a Semper function called *rad* which converts degrees to radians. For example, the following command is identical to the one given above.

```
rotate from 1 to display angle rad(45)
```

The test image appears on the screen rotated by 45 degrees.



Semper on-line help

Semper is a language, made up of *commands*. There is on-line help available for each Semper command. For a summary of the capabilities of the help facility, type the command:

```
help
```

To see a list of topics for which Semper help is available, type:

```
help /topics
```

To see the help for an individual command you simply type **help** followed by the *command name*:

```
help command name
```

For more comprehensive and detailed help, type the command:

```
help command name /full
```

For information on the syntax of a command, that is the way you construct a command line so that Semper understands it, type the command:

```
help command name.syntax
```

Try typing the following commands at the terminal. They will demonstrate the range of the on-line help and increase your understanding of the commands that you have used in this tutorial.

```
help magnify
help magnify/full
help magnify.syntax
```

Now substitute the command names **display**, **transpose** and **rotate** for **magnify** in the above command sequence.

Note that you can also obtain help about Semper error messages by typing **help ?*error number***. For example **help ?41**.

A note about syntax

A Semper command is made up of the following items:

- a command name
- keys
- options

The *command name* is the basic command, for example, **magnify**.

A *key* is added to a command to make its meaning more specific. For example, the keys **from** and **to** allow you to specify source and output pictures respectively and the key **times** allows you to specify a magnification factor for the **magnify** command. A key *value* is required for each key.

An *option* does not require a value and is usually an instruction to alter the way in which a command works. For example, the option **repeating** can be used with the **magnify** command to specify that the pixels of a picture are repeated rather than interpolated during the magnification process. The on-line help contains details of the syntax of each command, as does the manual:

Semper 6 Command Reference

For example, if you type the command:

```
help magnify.syntax
```

you are provided with the information shown below.

---------- Magnify.syntax ----------

Keys:

[FROM]	source picture [SELECT]
[TO]	output picture [FROM]
TIMES	integral magnification factor [2]
SIZE (2)	dimensions of subregion to be magnified [whole picture]
POSITION (2)	position/offset of subregion [0,0]
MARK	display to be marked with border indicating source region [none]

Options:

REPEATING	repeat pixels instead of interpolating
LEFT|RIGHT, TOP|BOTTOM	magnify subregion abutting indicated border

See SUBREGIONS for more details of subregion keys/options.
See SYNTAX.NOTATION for an explanation of notation used in –.syntax entries

The value in square brackets at the end of each key description is the default for that key. For example, the default for **times** is 2. Keys given in square brackets [KEY] can be omitted and are still understood by Semper. (See the following section, *Semper short cuts*).

Semper short cuts

Semper only takes note of the first three letters of any command, key or option, so you can considerably reduce the amount of typing you do by abbreviating the command. For example:

```
mag fro 2 to dis tim 4
```

means the same to Semper as:

```
magnify from 2 to display times 4
```

Semper also has a number of keys that you do not need to state explicitly, but are still understood. If a key does not need to be stated explicitly it is surrounded by square brackets in the command syntax. For example the keys **[from]** and **[to]** meaning source picture and output picture respectively, are used so often with Semper commands that they do not need to be stated. For example:

```
magnify 2 display
```

is the same as

```
magnify from 2 to display
```

Semper also provides a short cut in that you do not need to state the **display** command explicitly. When you type a picture number on its own, Semper assumes a **display** command. For example, if you type:

```
1
```

Semper understands this to mean:

```
display 1
```

Another short cut is provided in the form of *global* keys and options. These are keys and options that you can set at the start of a session and do not need to state in each command line. For example, at the beginning of this tutorial we set the global option **erase** to *yes* (on). This meant that before each command you did not need to type **erase**. Other examples of global keys and options include **mark** (which turns on or off display marking) and **verify** (which verifies the processing of a command at the terminal). For further detail, refer to *Chapter 3, The Command Interpreter* in the manual:

Advanced Users' Guide

contained in the *Semper 6 Guide*.

Capturing a live image

Not all Semper installations are able to perform live image capture (for example DEC VAXstations, some Silicon Graphics and Sun workstations cannot). Also the Semper command(s) that perform image capture vary according to the type of installation. Type the following command at your terminal for a description of the characteristics of your installation:

```
help framestore
```

You may like to capture an image of your own to experiment with, as you discover more about using Semper. A generalised procedure for capturing a live image is given below. These instructions assume that you have connected a camera to your framestore in accordance with the manufacturer's instructions. Refer also to the manual *Installing Semper 6 Plus on a PC* for details of framestore connections.

1. Switch on your camera and position it to point at the required object.

2. To activate the framestore and camera connection, type the command:

   ```
   live
   ```

   An image of the object appears on the display screen. The following message also appears on the screen:

   ```
   Press <RETURN> to grab...
   ```

3. Adjust the camera focus and position, if necessary, to create the required image on the screen. When you are satisfied with the image as it appears on the display, press the *<return>* key on the computer keyboard. This tells Semper to store the image in framestore memory.

4. To save the image in a file on disc, type the following command:

   ```
   copy display:1 to 5
   ```

This command copies the image from the display (assigned as device number 1) to a picture file, given the number 5, on the tutorial picture disc.

Stopping Semper

To leave the Semper environment at the end of your session, type the command:

```
stop
```

This returns you to the operating system of your own machine.

Summary

The table below gives a summary of the Semper commands that you used in this tutorial.

Action	Command Used
Starting Semper	**semper /run=tutorial**
Displaying a Semper picture	**display**
Magnifying a picture	**magnify**
Transposing a picture	**transpose**
Rotating a picture	**rotate**
Accessing the Semper on-line help	**help**
Capturing a live image	**live**
Stopping Semper	**stop**

Tutorial 2

FINDING OUT ABOUT THE SEMPER ENVIRONMENT

- ❑ Examining pictures
- ❑ Storing pictures
- ❑ The Semper Environment
- ❑ Semper Devices

Overview

This tutorial is designed to let you find out more about Semper pictures and about the Semper working environment.

Follow the start-up procedure described in *Tutorial 1, Getting Started.*

1. Start the tutorial software:

```
semper /run=tutorial
```

2. Set the global option **erase** by typing:

```
erase=yes
```

3. Display the test picture by typing :

```
display 1
```

Follow this simple start-up procedure at the beginning of each subsequent *Tutorial*, unless you are explicitly asked to do otherwise.

Finding out about your picture

In this section of the tutorial, we concentrate on finding out more about a picture, using the inspection facilities provided by Semper.

Type the following command at the terminal:

```
examine full 1
```

The **examine full** command produces the detailed report given overleaf.

```
2001  Size  128,  128,  1  16.0kb  Image  Byte  wp
      Suzy
      Range: 1, 252
      Origin column, row, layer: 65 65 1
      Created: 9-May-1988 14:49:19
      Blocks occupied 260 starting at 253
```

This supplements the standard report with information about:

- the actual minimum and maximum pixels values
- the position of the coordinate origin
- the time at which the picture was created (or last changed)
- the exact number of disk blocks it occupies
- the picture's starting block number within the disk file.

Now try the following command:

```
survey full 1
```

This command scans the test picture to find its data range, mean and standard deviation:

```
2001  Suzy
Size 128, 128
Range 1, 252
Mean 135.215   Sd 64.8124
```

Type the following command:

```
print 1
```

This prints a table of pixel values on the screen. A typical report might be:

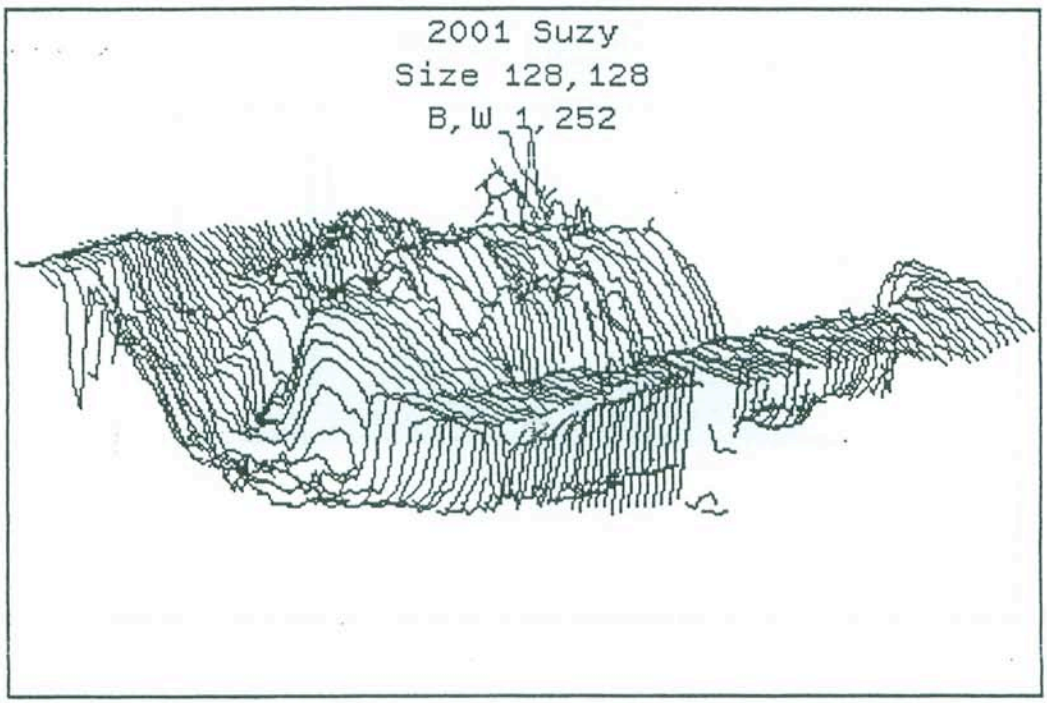
	-4	-3	-2	-1	0	1	2	3	4
4	203	197	194	191	189	187	181	176	167
3	201	198	200	191	192	185	186	184	181
2	207	205	200	192	186	183	179	182	186
1	213	209	206	202	193	180	178	178	178
0	210	214	213	203	196	188	185	179	180
-1	215	219	219	217	201	196	185	179	178
-2	219	221	217	204	197	194	186	179	175
-3	213	217	202	183	185	185	180	177	180
-4	198	202	199	187	171	171	170	170	177

Note that it can be useful to print a table of pixel values when debugging image processing algorithms.

The next three commands, **ymodulus, histogram** and **contour,** present information about the test picture in a *graphical form* on the display. Type the following command at the terminal:

```
ymodulus from 1 to display
```

Semper creates a 'perspective' display of the test picture, where pixel intensity is plotted as surface height in a line drawing, as is shown below:

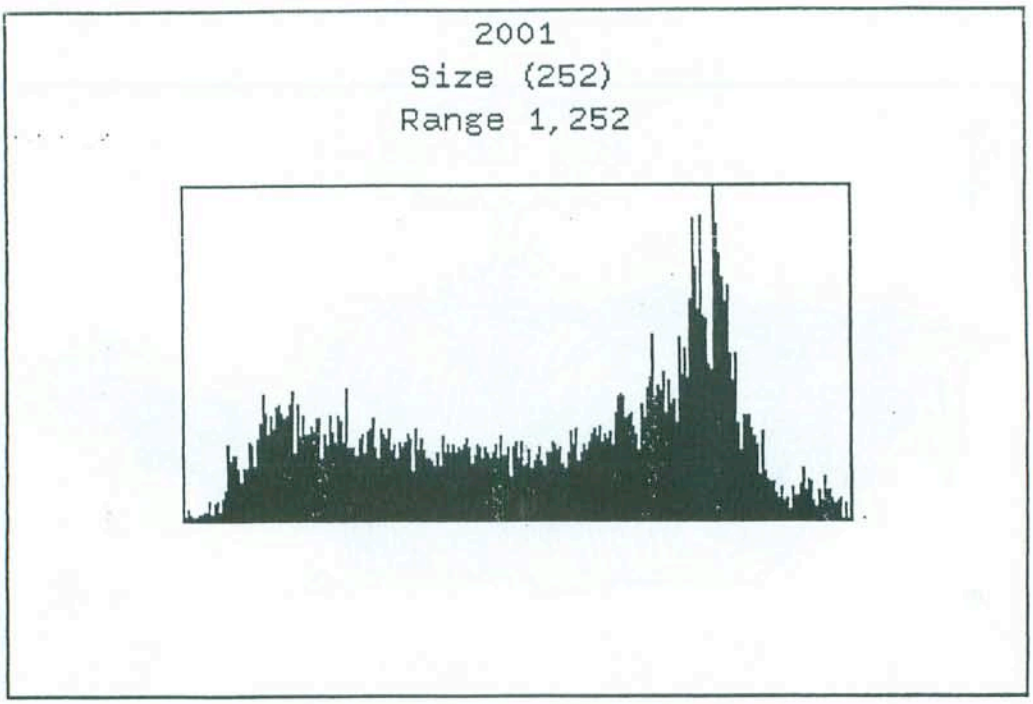


Try the effect of the following commands on the test picture:

```
ymodulus from 1 to display size 10 times 2
ymodulus from 1 to display size 50,50 top left
```

Alternatively, you can use the **histogram** command to create a picture intensity histogram showing the intensity ranges of pixels in a picture. The **histogram** command produces a grey level histogram of an image – you can see from the graph where the majority of grey levels (higher channels) are concentrated in the picture. Type the following command to produce a histogram on the display:

```
histogram from 1 to display
```



Try the effect of the following commands on the test picture:

```
histogram from 1 to display channels 150
histogram from 1 to display size 20,20 position 20,20
```

You can also specify a disc picture as the destination for a histogram, instead of the display. For example, try typing the commands:

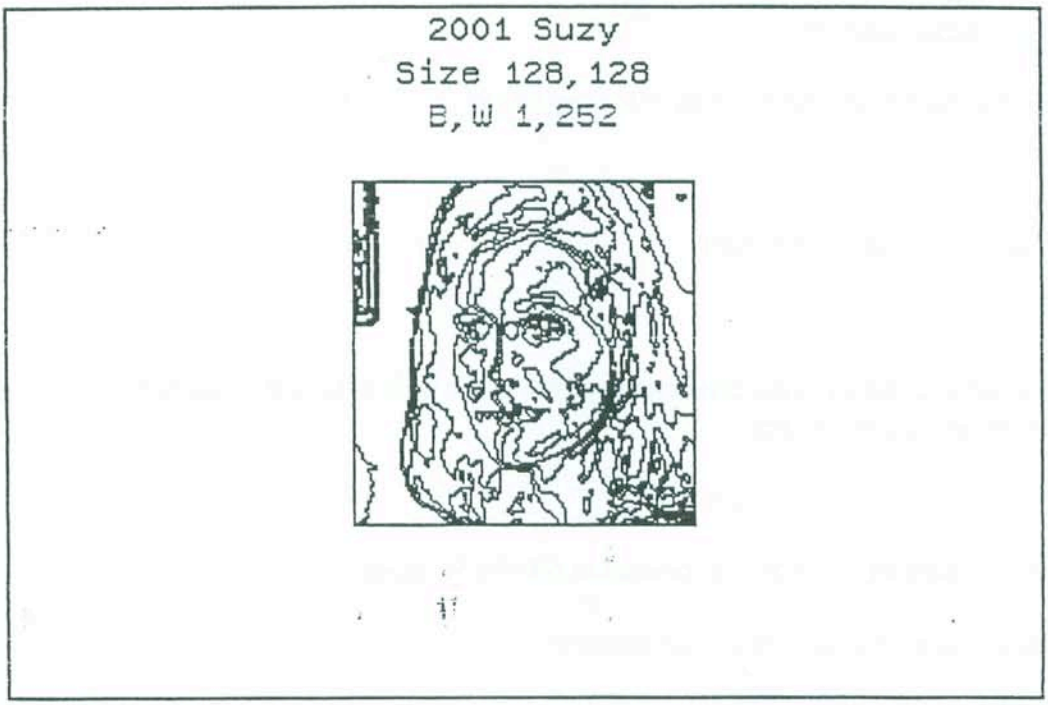
```
histogram from 1 to 6
examine all
```

Semper stores the histogram in a class *Histogram* picture, which you can display when required or use with the **map** command to perform histogram equalisation. For further details of picture classes, refer to *Appendix B, Picture Types*.

The **contour** command draws picture intensity contours on the display. Type the command:

```
contour from 1 to display
```

Semper produces the following display:



Try the effect of the following commands on the test picture:

```
contour from 1 to display times 3
contour from 1 to display levels 8
```

Storing pictures

Now that you have learnt more about the test picture, you may like find out more about the Semper working environment. This part of the tutorial shows you how to:

- copy a picture
- rename a picture
- turn write protection on and off
- delete a picture

First of all we'll make another copy of the test picture.

```
copy from 1 to 7
```

To verify this, type the command:

```
examine all
```

To rename a picture, use the **title** command. For example, to rename the picture that you have just created, type the command:

```
title 7 text 'Suzycopy'
```

Type **examine all** to confirm this change or **display** the picture.

To write-protect a picture, type the command:

```
wp 7
```

Type **examine all** to see the change and then turn off write-protection using the command:

```
wp 7 off
```

Finally, to delete picture 7, type the following command:

```
delete 7
```

This command releases the space that a picture occupies on disk. Type the command:

```
examine all
```

again to confirm the deletion.

Note that you can also delete a range of pictures, for example, the following command would delete any pictures numbered in the range 50 to 90:

```
delete 50,90
```

The Semper environment

So far in this tutorial we have concentrated on pictures, learning how to examine and store them. In this section of the tutorial we shall apply a wider view and look at the Semper system as a whole.

The **show** command is a source of valuable information about your working environment. Try typing the following commands at the terminal.

```
show system
show commands
show variables
```

The first command gives you details about your installation (size limits, i/o units etc.) The second command shows you all the available Semper commands. The third command lists the variables that are set and their values. Try typing the **show** command on its own. This gives you an overview of the **show** options.

Semper devices

To find out about the devices which Semper is currently using, type the command:

```
show devices
```

You should see something like the following information on the terminal:

```
Devices assigned:
   1 Display
      Number of frames: 1 Frame size: 512, 512 Monitor size: 512, 512
      Character size: 8,13 Grey pixel size: 1
   2 Disk tutorial.dsk
      Size: 1008.8kb = 16128 blocks Directory size 1024 slots
   3 Help library semper.hlb wp
      Size 352.0kb = 5632 blocks
   4 Program library semper.plb
      Size 88.0kb = 1408 blocks Directory size 32 slots
```

The term *device* in Semper means a storage area which may contain several images or other data; devices are identified by numbers 1,2,3.., and the list printed when you type **show devices** shows you what is initially available to your session.

The list tells you that you have access to four devices:

1. A display (device 1, probably of the framestore type) with one frame of 512 by 512 pixels.

2. A disk file *tutorial.dsk* (device 2), one-quarter of a Megabyte in size which is your main picture storage area.

3. A further disk file (device 3) containing information to be used by Semper's **help** command, and which you cannot alter (**wp** indicates write-protected).

4. A library of Semper programs (*device 4*). See *Tutorial 6, Programming with Semper* for details of Semper's program libraries.

You can use the **assign** command to create new devices, for example, a temporary work disk, or assign further resources (including magnetic tapes) to your session. To assign a scratch (temporary) work disk of 1000 kilobytes in size, type the following command:

```
assign scratch size 1000
```

Semper displays the following messages:

```
Device 5 assigned
Device 5 initialised with 2008 size directory
```

If you now type **show devices** you will see that Semper has created a temporary workspace on disk, which is called device 5. This device, and any pictures it may contain, is deleted automatically at the end of a Semper session. You may now like to experiment with copying between devices. For example, to copy the test picture *Suzy* on device 2 to device 5, type the command:

```
copy 2001 to 5001
```

Next type the command:

```
examine device 5
```

Semper has created a copy of the test picture on the scratch work disk:

```
Scratch workspace device 5: c\semper6p\tutorial.dsk
5001  Size 128, 128, 1 16.0kb Image Byte
      Suzy
```

To obtain full details of the powerful **assign** command you can type the following command:

```
help /full assign
```

or consult the manual:

Semper 6 Command Reference

To end the Semper session, type the command:

```
stop
```

Summary

The table below presents a summary of the Semper commands that you used in this tutorial:

Action	Command Used
Displaying information about a picture	**examine/survey**
Printing picture pixel values	**print**
Creating a perspective display of a picture	**ymodulus**
Displaying a grey scale histogram of a picture	**histogram**
Displaying picture intensity contours	**contour**
Copying a picture	**copy**
Renaming a picture	**title**
Write-protecting a picture	**wp**
Deleting a picture	**delete**
Displaying installation details	**show system**
Displaying a list of assigned devices	**show devices**
Assigning a new device	**assign**

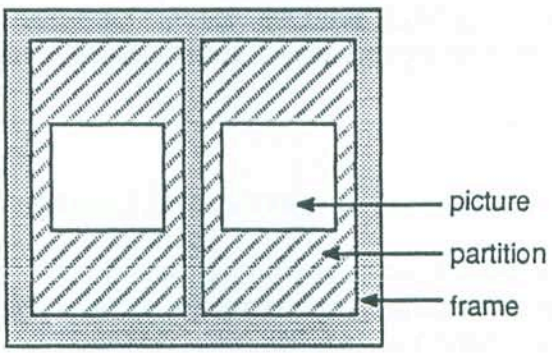
Tutorial 3

VIEWING YOUR PICTURES

❑ Frames ❑ Partitions ❑ Look-up tables ❑ Overlays ❑ Erasure

Overview

This tutorial shows you how to change the viewing conditions of your display. Semper is able to manage quite sophisticated displays in a very flexible manner. It can divide a display into *frames* – a frame is a surface that can carry images. For example, a colour display has three frames; one each for red, green and blue. Semper can also subdivide frames into *partitions*, which are independent picture storage areas within your display. Usually, the number of frames is fixed by the system's physical capacity, but you can define the number, size and position of partitions at any time to suit yourself. The diagram below illustrates these concepts:



This tutorial also describes viewing pictures using monochrome, false colour or true colour look-up tables, creating a test pattern on the screen and using overlays. To use this tutorial, start Semper using the procedure that is outlined in tutorials 1 and 2, but do not display the test picture at this stage.

Partitions

In previous tutorials you have treated the display device as if it could contain only a single picture (*display:1*). In fact, you can subdivide the display to provide many different display partitions, each capable of holding one picture. These are defined by numbers 1, 2... (in full, *display:1* or *dis:1*, *dis:2*) up to an installation dependent maximum. To see the default partitions for your installation, type the following command:

```
show partitions
```

The following results are given for a display frame with the dimensions 768 by 512:

Display partitions:

1	Size: 768, 512	Centre: 0, 0	Frames: 1, 1	Lut: 1
2	Size: 512, 512	Centre: –128, 0	Frames: 1, 1	Lut: 1
3	Size: 512, 512	Centre: 128, 0	Frames: 1, 1	Lut: 1
4	Size: 256, 256	Centre: –256, 128	Frames: 1, 1	Lut: 1
5	Size: 256, 256	Centre: 256, 128	Frames: 1, 1	Lut: 1
6	Size: 512, 256	Centre: 0, –128	Frames: 1, 1	Lut: 1
7	Size: 512, 512	Centre: 0, 0	Frames: 1, 1	Lut: 1
8	Size: 512, 512	Centre: 0, 0	Frames: 1, 1	Lut: 1

This means that all the partitions are designed to fit the entire current display screen.

You can define more useful partitions than the default partitions, using the cursor to mark a region on the screen. The following sequence of commands defines 3 partitions on the screen.

Type the command:

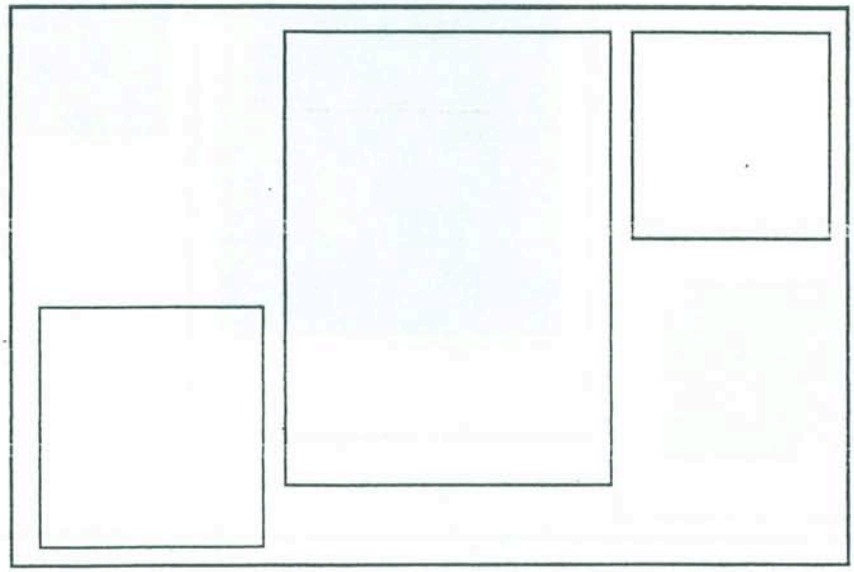
```
xwires frame region; partition @region 1
```

You will see a cross-hair cursor appear on the image display. Use the mouse to move the cursor to define a partition in the bottom left of the screen or display window. Click the left mouse button once at the required position of the bottom left corner of the partition. You will see a "rubberband" effect on the display. Click the left mouse button again at the position of the top right corner of the partition box.

Then define two more partitions – partition 2 in the centre of the display window/screen and partition 3 in the top right corner of the display window/screen, using the following commands:

```
xwires frame region; partition @region 2

xwires frame region; partition @region 3
```

Your display screen/window should now look something like the one shown below. (Note that it is also possible to define partitions that overlap).



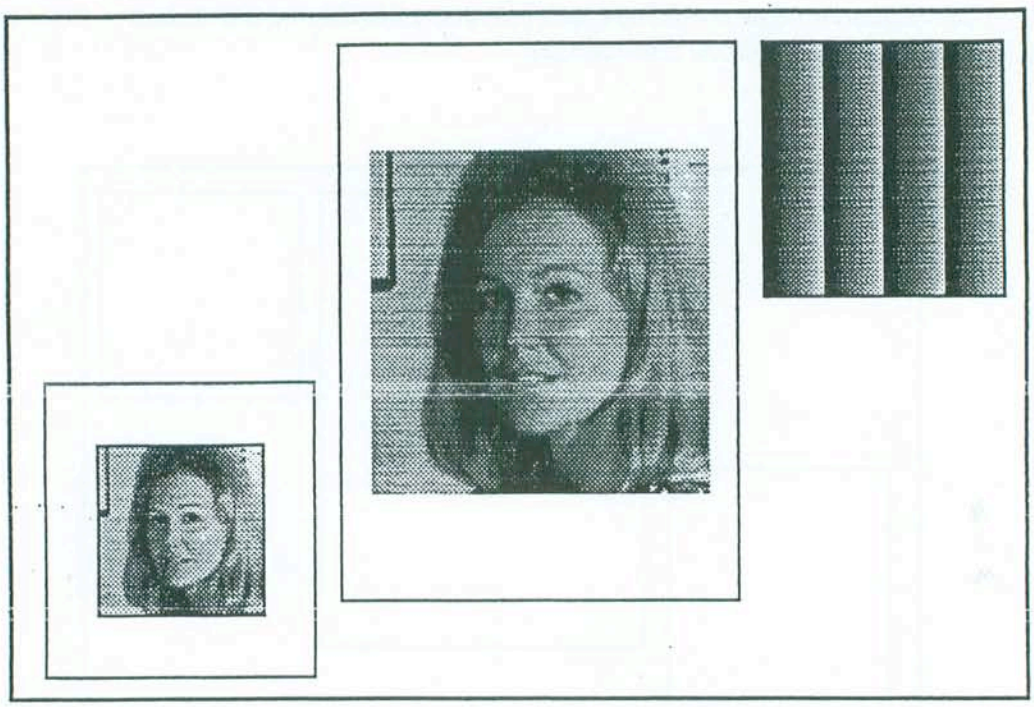
Having defined your partitions, the next step is to test the effect of the following commands:

```
display 1 to display:1

magnify 1 to display:2

create 1003 size 128

ramps display:3
```

The **display** command displays picture 1 in partition 1.

The **magnify** command attempts to double the size of your picture and display it in partition 2, but the magnification may not be apparent if you have defined a relatively small central partition. (Semper scales a picture that is too big to fit its partition by undersampling. You will also find that Semper refuses to read back or process further an undersampled picture, as some of its data is missing).

The **create** command creates a Semper display picture in partition 3. The **ramps** command writes a test pattern to the display picture. You can use this pattern whenever you adjust the brightness or contrast of your monitor. Your screen should now look something like the one shown overleaf.



Look-up tables

The human eye is much more sensitive to variations in colour than gradations in grey-scale levels. Accordingly Semper has look-up tables that allow you to map digital data in a variety of ways. To see the default look-up tables at your installation type:

```
show luts
```

You should see the following appear on the terminal screen/display window:

```
Display look-up tables

1  Mode: monochrome
2  Mode: false
3  Mode: monochrome
```

Now try viewing using a false colour look-up table, where the normal grey scale is replaced by a series of colour gradations:

```
view lut 2
```

To return to monochrome, type:

```
view lut 1
```

Note that the **view** command allows you to control the way in which you view your image on the screen, without altering or destroying the information in the display framestore.

You can also use the commands **lut**, **lset** and **ladjust** to create further look-up tables of your own, or to adjust existing ones. For example, the **ladjust** command allows you to alter the following parameters of a look-up table interactively, using the mouse:

- brightness (the intensity of a colour)
- contrast (the degree of difference between tones)
- hue (the dominant colour)
- saturation (the amount of white light blended with the hue)

Try typing the following command:

```
ladjust brightness contrast
```

and move the cursor using the mouse or cursor keys. This command adjusts the display brightness on vertical movements of the mouse/cursor and adjusts the contrast on horizontal movements.

A note about true-colour systems

If you have a true-colour system, the colour display has three frames which are used to display red, green and blue. To view in true colour, you need to display a three-layer picture and create a full colour look-up table to view it, using the command:

```
lut 3 create colour
```

The three layers of the picture are treated as red, green and blue components respectively. Refer to *Appendix B, Picture Types* for details of multi-layer pictures.

Overlays

Semper uses a separate overlay plane to display text, annotation, graphics and the cursor, from the plane that it uses to display images. You can specify a colour for the overlay plane using the **overlay** command, for example, try the following commands:

```
overlay green
overlay red
```

If the **overlay** command does not produce the expected results, it may be because the overlays operate in a special way for your installation. In this case, read the installation specific information for the **overlay** command by typing:

```
help overlay
```

Erasing

You can erase the contents of an individual partition using the **erase** command. Type the following command:

```
erase partition 2
```

or you can erase a partition overlay, by typing the command:

```
erase overlay partition 1
```

Try erasing the contents of a whole frame, using the command:

```
erase frame
```

To end this Semper session, type the command:

```
stop
```

Summary

The table below presents a summary of the Semper commands that you used in this tutorial:

Action	Command Used
Displaying a list of the current partitions	**show partitions**
Marking an area on the display	**xwires...region**
Creating a Semper picture	**create**
Displaying a test pattern on the screen	**ramps**
Displaying a list of the current look-up tables	**show luts**
Altering the viewing conditions	**view**
Adjusting a look-up table interactively	**ladjust**
Specifying a colour for an overlay	**overlay**
Erasing all or part of the display	**erase**

Tutorial 4

IMAGE PROCESSING

❑ Filtering an Image ❑ Transformations ❑ Performing arithmetic on an Image

Overview

In this tutorial, we are going to perform some typical image processing operations on an image, including filtering and transformation of an image and using arithmetic calculations on an image.

Before beginning this tutorial, follow the instructions given below:

1. Start the tutorial software by typing:

```
semper /run=tutorial
```

2. Set the **erase** option by typing:

```
erase=yes
```

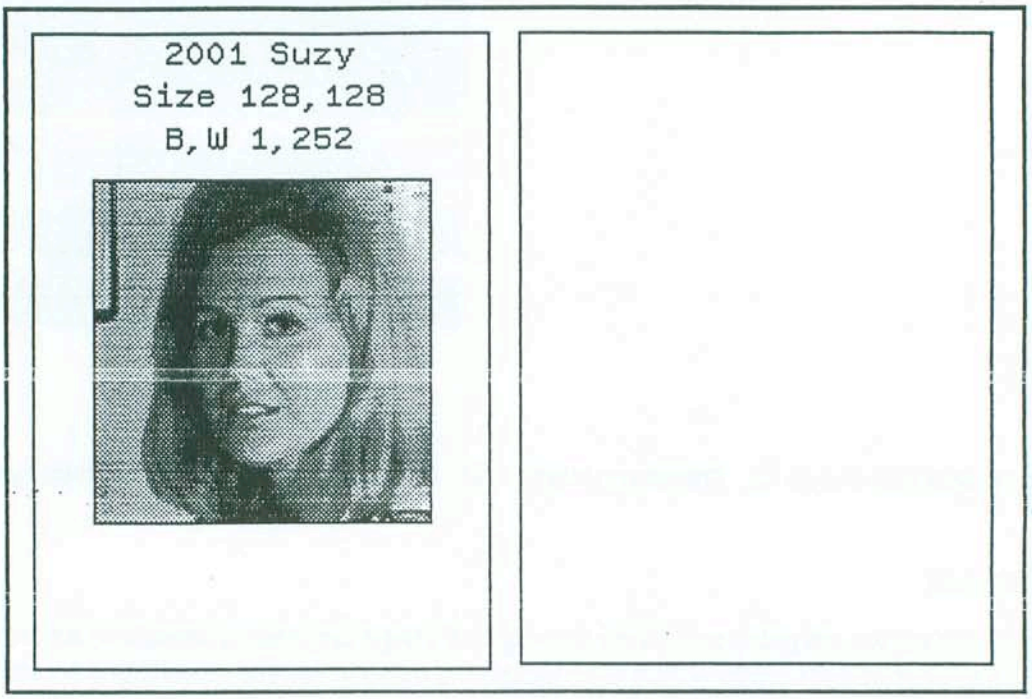
3. Define two partitions on your screen, to split up the screen or display window horizontally, using the mouse. Use the following two commands (you will see a cross-hair cursor appear on the image display):

```
xwires frame region; partition @region 1
xwires frame region; partition @region 2
```

4. Display the test picture in the first partition using the command:

```
display from 1 to display:1
```

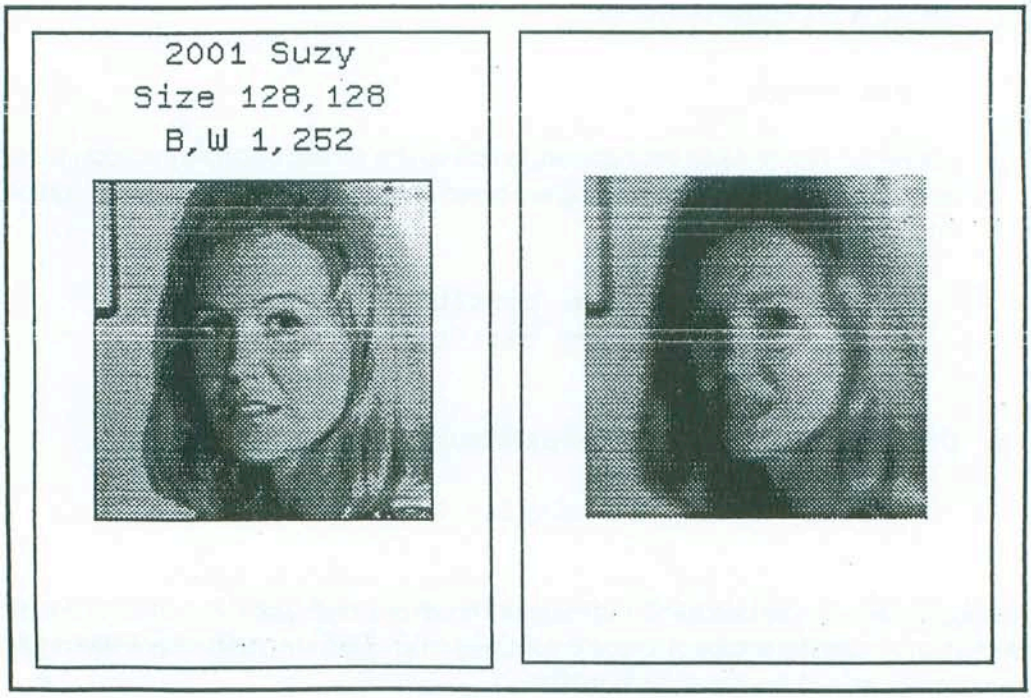
Your display should now look similar to the one shown overleaf. The two partitions are defined so that you can display the results of image processing on an image in partition 2, whilst displaying the original image as a reference copy in partition 1.



Filtering an image

To perform some simple filtering on your image, type the command given below. An averaged image appears in *display 2*, as is illustrated:

```
lmean over 5 from 1 to display:2
```



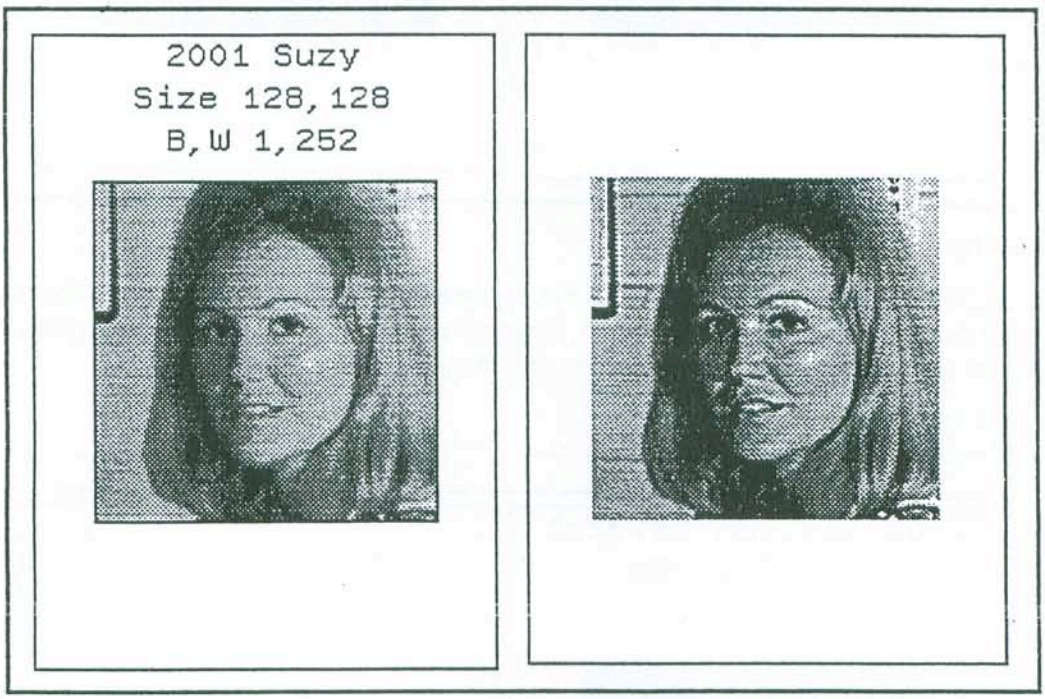
The **lmean** command calculates the local mean over a square block of neighbouring pixels. In effect, it tells you the general brightness around each pixel and reduces the noise level of a picture at the expense of fine detail, as shown below:

To sharpen the original image, type the following command:

```
sharpen over 5 from 1 to display:2
```

The **sharpen** command applies a simple picture sharpening filter, effectively doubling the high spatial frequencies (fine detail) by adding the difference between the original picture and a locally averaged version.

The effect on the image is shown below.



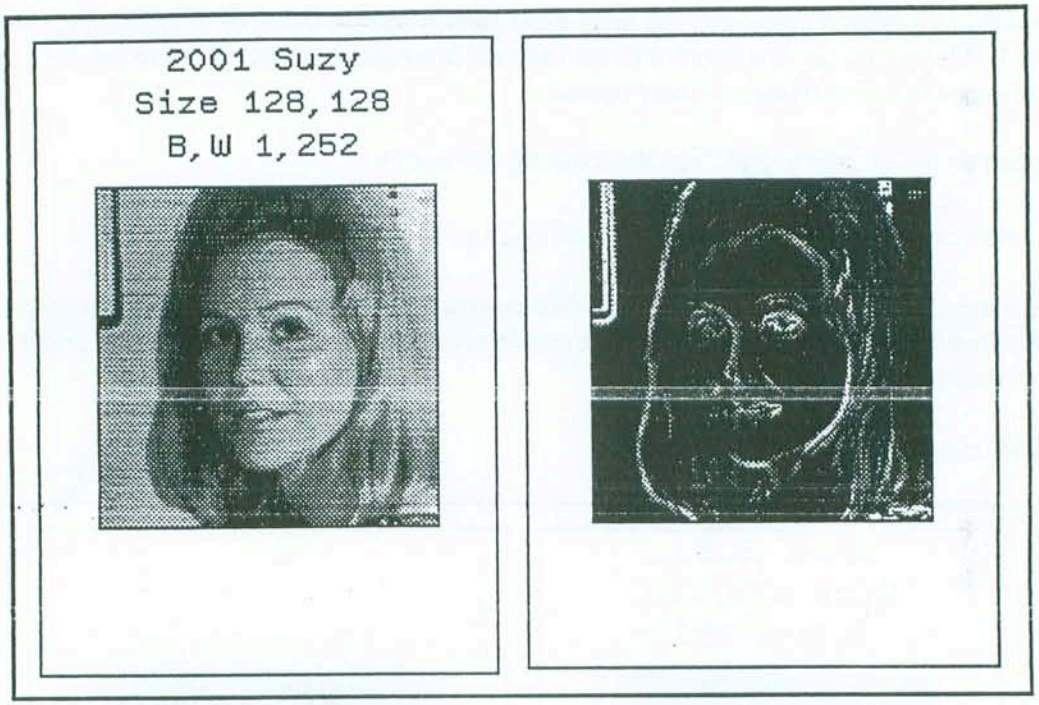
Now change to a false colour look-up table. This allows you to see the effect of the following commands more clearly.

```
view lut 2
```

Try two edge detection operations on the test image using the following commands:

```
edge from 1 to display:2
edge roberts from 1 to display:2
```

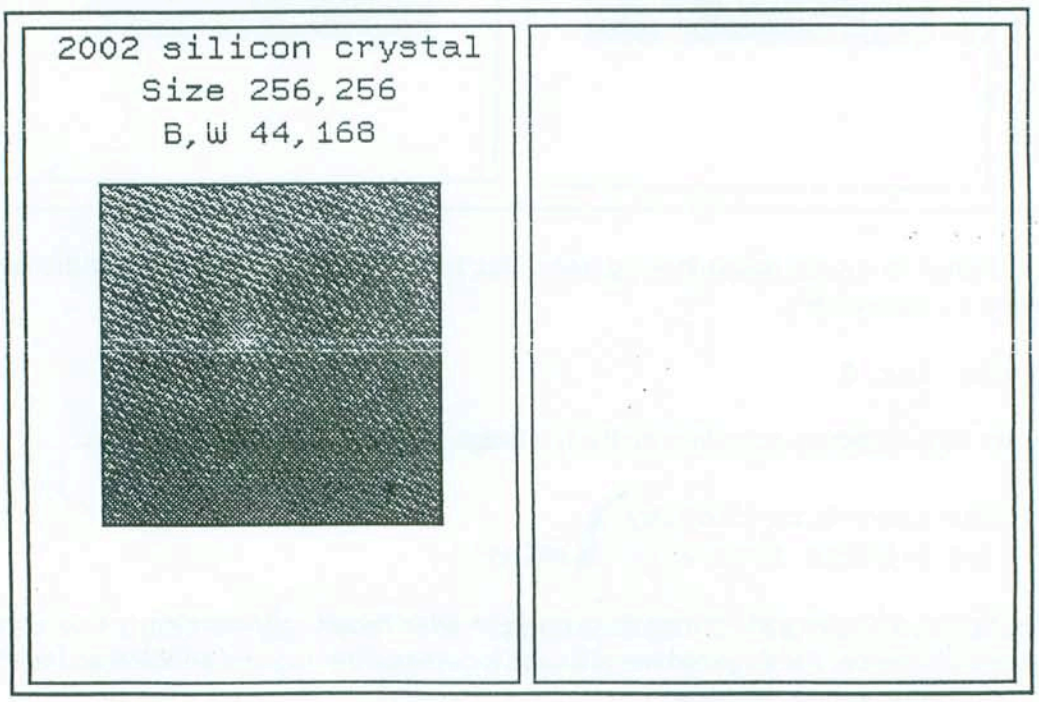
edge applies a 3-point gradient magnitude operator, **edge roberts** applies a larger 4-point absolute diagonal difference. The **edge** command is used to delineate the edges of an object and its effect on the test picture is shown overleaf.



Transforming an image

This section describes how to calculate a *Fourier* transform of the test picture of a silicon crystal (picture 2) and display its power spectrum. Clear the screen and display the silicon crystal image using the commands given below. The following image appears on the screen:

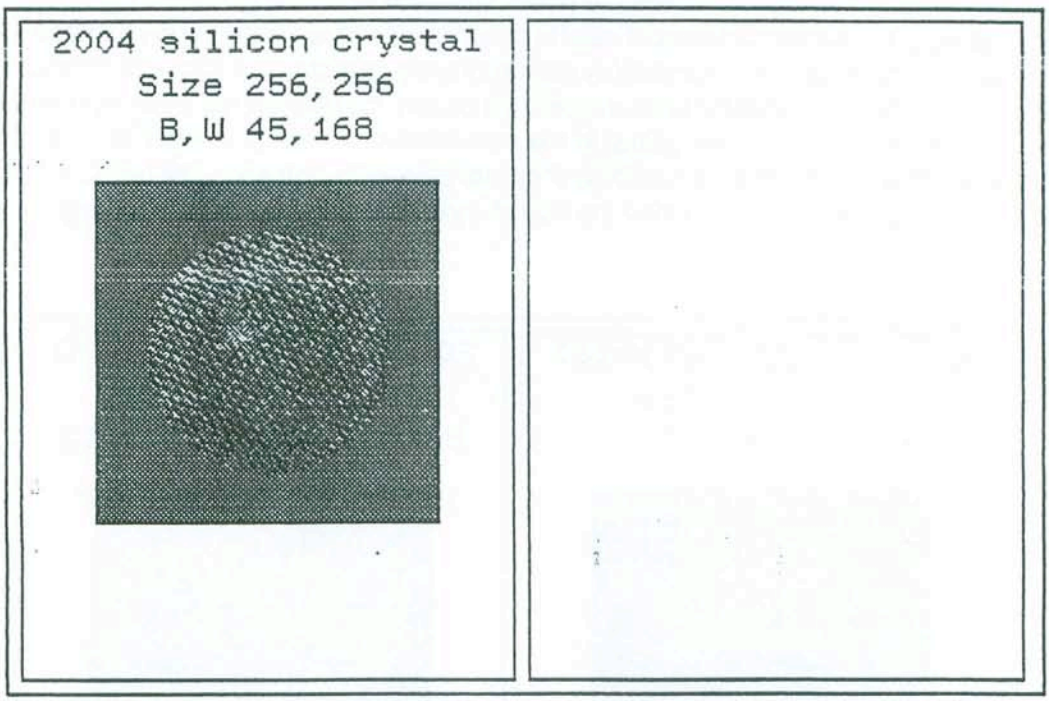
```
erase frame
display from 2 to display:1
```



To calculate a discrete *Fourier* transform we first need to mask off part of the image, to avoid a discontinuity at the edge of the image (as Semper assumes a repeating pattern). Create a new masked source picture 4 and display it using the following commands:

```
mask from 2 to 4
display from 4 to display:1
```

The following image appears on the display:



Calculate the *Fourier* transform of the masked image using the command:

```
fourier from 4 to 5
```

The output is sent to a new disk picture instead of to the display, because it is not useful to display the output of a *Fourier* transform as it is in complex floating point form. (See *Appendix B, Picture Types* for details of picture class and form). Create the power spectrum of the new disk picture, by typing the following command:

```
ps from 5 to 6 ln
```

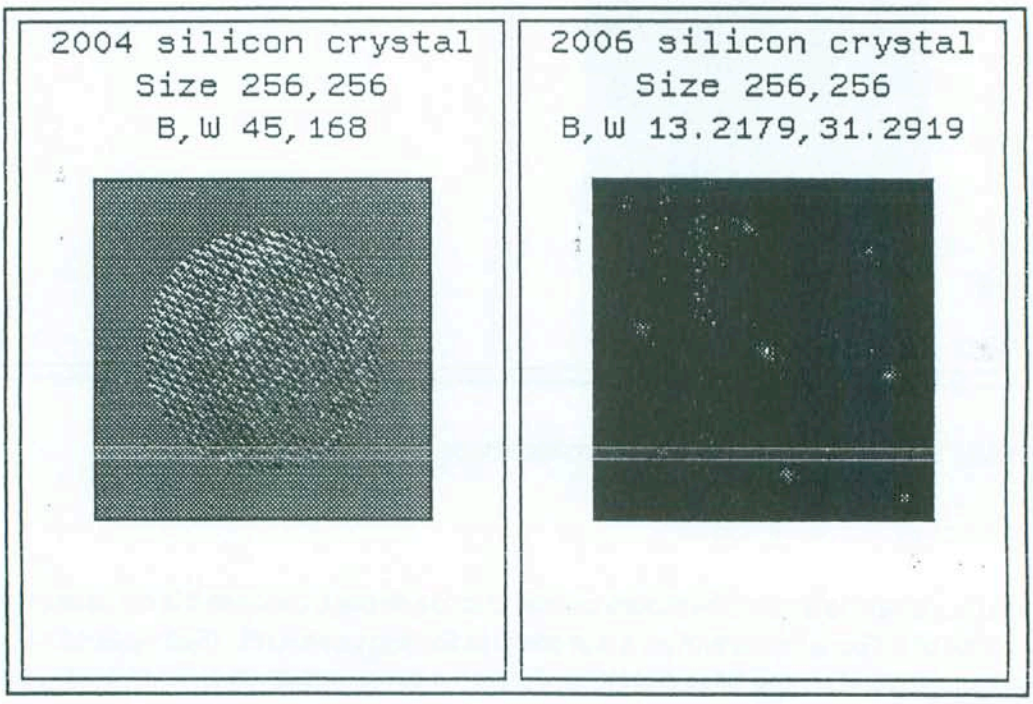
The **ln** key causes the logarithm of the power spectrum to be calculated because the dynamic range is too big to be displayed unless you compress it. (If you try and display it directly to the display without the **ln** key you will only see a white dot on the screen).

```
fullplane 6
```

As the power spectrum of a real image is symmetric, Semper only calculates the half plane to save space. The **fullplane** command reconstructs the full plane image.

```
survey full 6
min=mean; display 6 to display:2 preset
```

This sequence of commands causes the power spectrum to be displayed with dynamic range compression. The **survey** command is used to determine the mean of the picture and this value is then used to override the automatic display scaling by setting **preset**. This increases the contrast of the result, avoiding a washed out appearance in the picture. You will see the following result on the screen or display window. The peaks in the transformed picture give information about the properties of the crystal lattice; in particular they provide information about the unit vectors of the lattice which can be used to determine the type of crystalline structure you are viewing.



Performing arithmetic on an image

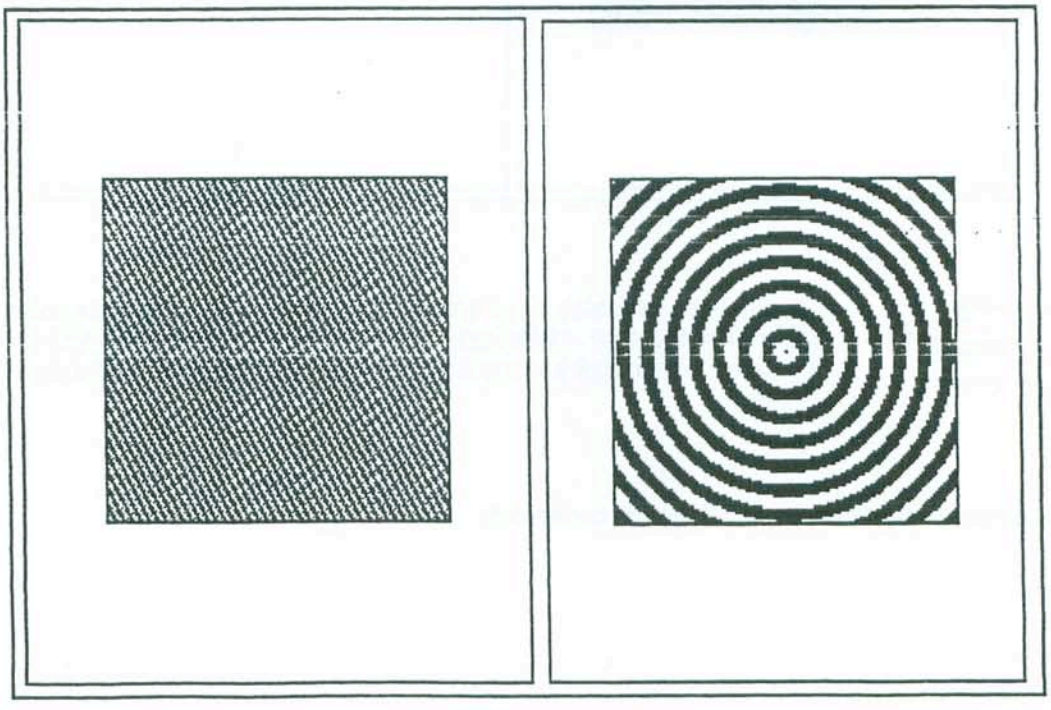
Semper allows you to perform arithmetical operations on pictures on a pixel by pixel basis, using the **calculate** command. This command allows you to generate synthetic images with intensity values which are an arbitrary mathematical function of pixel coordinates. The **calculate** command also allows you to add, subtract and divide pictures using the standard arithmetical expressions (+ − * /) and the arithmetical functions that are built into Semper, for example, sine, cosine, tangent. (For a full list of Semper functions, refer to *Chapter 2, Semper Elements* in the *Advanced Users' Guide).*

As a first step, you can use the **calculate** command to generate two synthetic images. When creating a synthetic image, Semper needs a defined destination to establish the size and coordinate system for the new picture, so the **create** command is used before **calculate** to create a new disk picture. Try typing the sequence of commands given below:

```
view lut 1; erase frame
min=-1 max=1; create 7 fp size 256
kx=1 ky=.5; calculate sin(kx*x+ky*y)
display 7 to display:1
```

This sequence of commands creates a sinusoidal grating, whose spacing and direction you can vary by changing the values of the variables *kx* and *ky*. You can create a second synthetic image by typing the following commands. The two images are shown below.

```
create 8 byte size 256
calculate (sin(root(x*x/8+y*y/8))>0)
display 8 to display:2
```

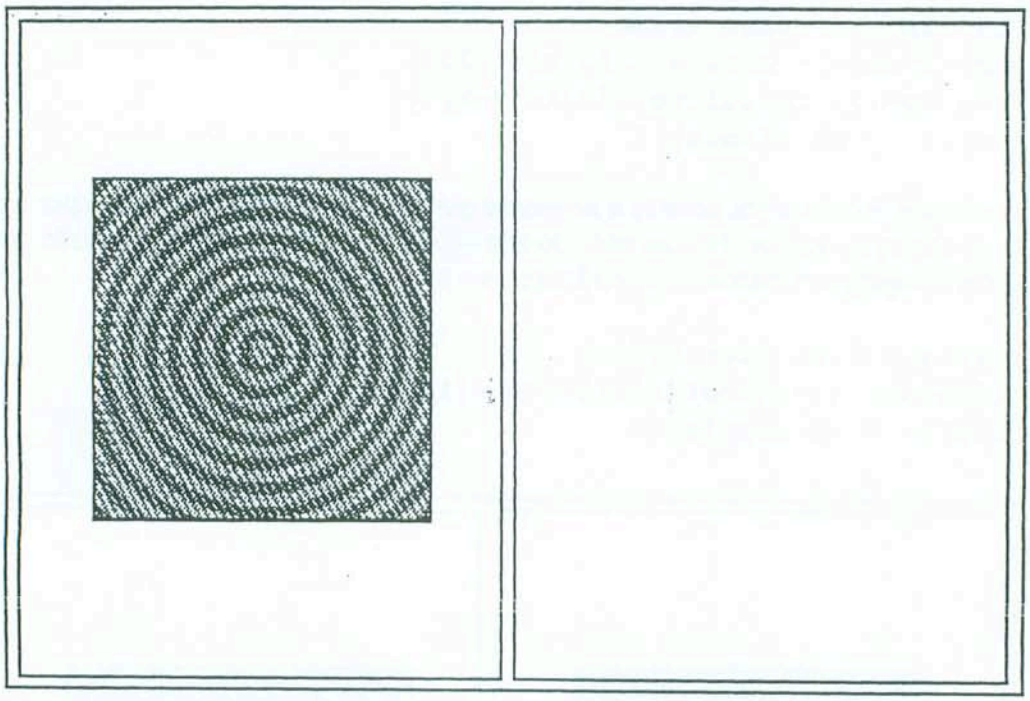


This type of synthetic image could be used as a test pattern for image processing algorithms or equipment.

You can also use **calculate** to perform pixel by pixel operations *between* pictures. Try typing the following commands:

```
erase partition 2
calculate (:7+:8)/2 to display:1
```

This produces an average of pictures 7 and 8 in partition 1. Note that in this instance the current display picture *1001* is used to establish a picture size and coordinate system. The average of the two pictures is shown below.



If you use the variables *x*, *y*, *z* and *rr* in expressions in a **calculate** command, you will find that they are treated in a special way, meaning respectively the *x*, *y*, *z* pixel coordinates and the squared radius (distance from the coordinate origin).

To leave this Semper session, type the command:

```
stop
```

Summary

The table below presents a summary of the Semper commands that you used in this tutorial:

Action	Command Used
Applying a local mean filter	**lmean**
Sharpening an image	**sharpen**
Performing an edge detection operation	**edge/edge roberts**
Creating a masked image	**mask**
Calculating a *Fourier* transform	**fourier**
Calculating a power spectrum	**ps**
Reconstructing a full plane image	**fullplane**
Performing arithmetic on an image	**calculate**
Creating a Semper picture	**create**

Tutorial 5

PARTICLE ANALYSIS

- ❑ **Defining the limits of analysis**
- ❑ **Particle Analysis**
- ❑ **Editing particles**
- ❑ **Displaying the results of analysis**

Overview

This tutorial introduces you to the particle analysis capabilities of Semper. Many image processing applications require differentiation and analysis of particles, and Semper is provided with a number of commands to deal with this type of application. A typical particle analysis task might consist of the following steps:

- setting up a picture for analysis
- analysing a picture
- interpreting the analysis
- further analysis based upon the results, for example, measuring the parameters of an individual particle

The output produced by particle analysis requires a sizeable amount of disk space. Before starting this tutorial, it is a good idea to delete redundant pictures from your picture disk (that is, pictures created in the course of these tutorials) – leaving the write-protected pictures 1, 2 and 3 of Suzy, the crystal and blood cells. You may also try **compress**ing the disk, to reduce fragmentation of disk space.

Setting up a picture for analysis

Start the tutorial software by typing the command:

```
semper /run=tutorial
```

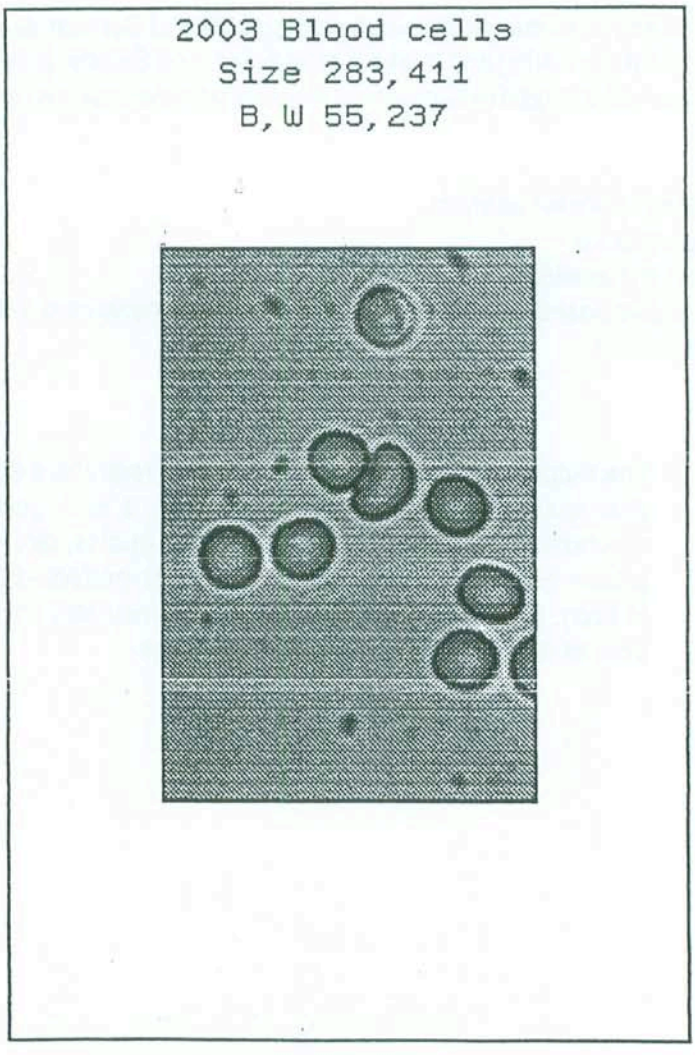
As a preliminary step, set the global option **erase**:

```
erase=yes
```

Your tutorial picture disk contains an image of particles. The next step is to display it on the screen.

```
display from 3 to display:1
```

This picture contains a number of blood cell particles, as illustrated in the image shown below.



The second step in setting up a picture for analysis is to use the **ladjust** command to define the particles, that is, to use the display look-up table to highlight all pixels that you require Semper to define as particles. Before using **ladjust**, ensure that you are using a false colour look-up table:

```
view lut 2
lut reset
```

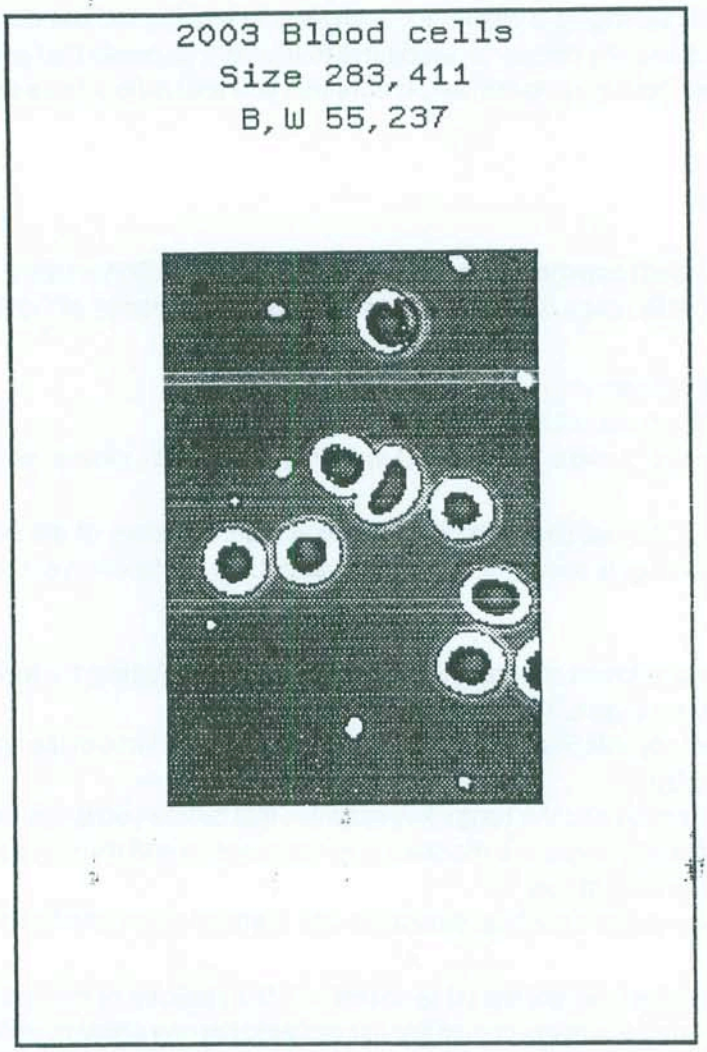
Now specify the **ladjust** command. Note that you need to define two variables *m* and *m2*, to use as the initial look-up table range, so that Semper highlights the midpoint of the range at the outset.

```
m=min+0.25*(max-min)
m2=min+0.75*(max-min)
ladjust upper lower scaled display:1 initial range m,m2 hsv 120,1,1
```

The purpose of the **ladjust** command is to highlight the perimeters of the blood cells, using the display look-up table, as is shown in the image overleaf. An explanation of this command is given below:

- the **upper** and **lower** options tell Semper that you are adjusting the lower and upper limit of modification range of the look-up table
- the **scaled** key tells Semper to scale the range values in terms of the black and white levels of the *display*
- the **initial** option and the **range** key specifies that before you begin adjusting the look-up table the display levels are modified over the range *m* and *m2* (defined as approximately the middle of the range)
- the **hsv** key defines the hue, saturation and intensity of the end limits of the look-up table

Use the mouse to adjust the look-up table so that only the particles of interest are coloured. At the end of the adjustment your image should appear similar to the one shown overleaf. When you press the mouse key, Semper sets the variables *r* and *r2* which are used by the **analyse** command as the threshold values in the next stage of the analysis. These variables contain the grey scale range that Semper will use in the analysis.



Analysing the Picture

Use the following command to analyse the picture.

```
analyse 3 to 8 segment 9 ge r le r2 area 10
```

This command counts and measures particles with pixel values of an intensity range between the values held in the variables *r* and *r2* (defined by the **ladjust** command), ignoring particles with an area less than 10. The **analyse** command produces two forms of output:

- a particle parameter list (*ppl*) which contains measurements of the individual particles
- a segmentation map which is a visual representation of the particles selected by **analyse**

In this instance, you specify picture 8 as the particle parameter list and picture 9 as the segmentation map. At the end of the analysis, Semper displays a message similar to the following:

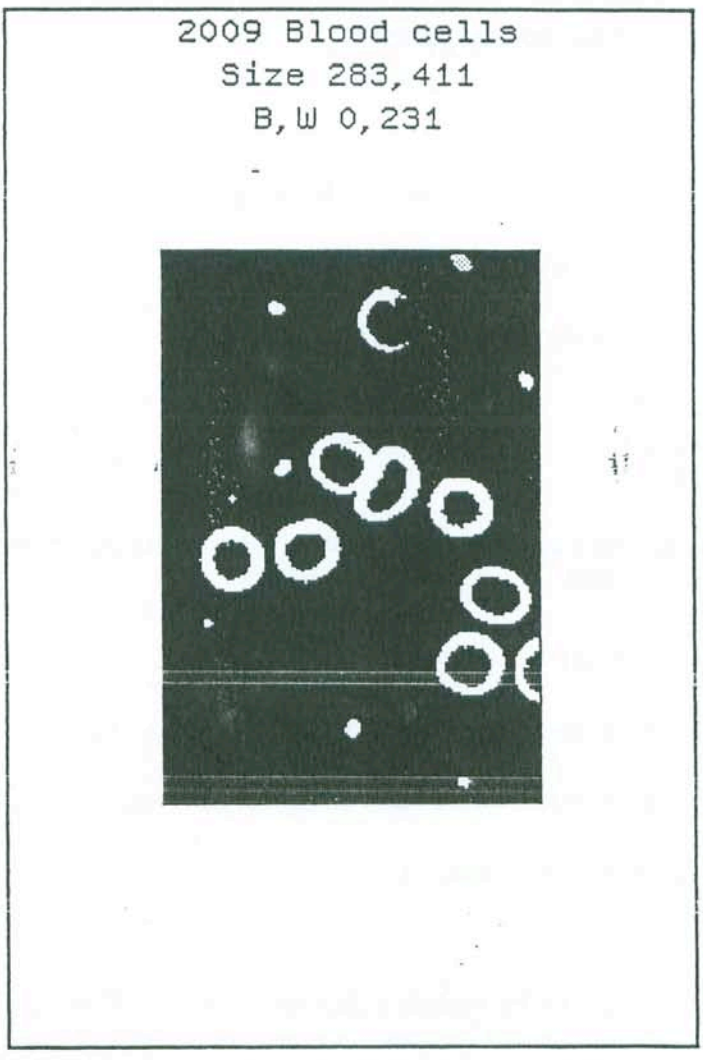
```
Number of particles found: 16
```

Interpreting the analysis (1)

The next step is to interpret the results of the analysis. Type the commands:

```
pshow; min=0 max=255 display noscale from 9 to display:1
```

This sequence of commands displays the segmentation map produced by **analyse**, which shows the particles that were found during the analysis. (The **pshow** command is used to alter the display look-up table to highlight the particles). This map is illustrated below:



From this segmentation map we can see that **analyse** has found the blood cell particles and also small background particles that we do not wish to measure in this application. To eliminate these particles from the analysis we can change the analysis limits and re-analyse the picture.

Changing the analysis limits

To eliminate the small background particles, we can specify a minimum particle area to be recognised by the **analyse** command. To discover a sensible minimum area, we can use Semper's particle analysis facilities. Type the following sequence of commands at the console:

```
xwires noverify; pid @xy
```

A cross-hair cursor appears on the image display (**xwires**). Use it to select a single (small) particle. (Note that if, by mistake, you do not select a pixel belonging to a particle, Semper displays the error message *Background hit*). The **pid** command identifies the particle at the specified position and stores its particle identifier in the variable *pid*. You can use the **pshow** command to alter the display look-up table to highlight the selected particle.

```
pshow if id=pid
```

To display the area for the test particle, type the following command:

```
ptype if id=pid area
```

You will see text on the screen similar to that given below:

```
Particle id: 1
Index     Area
1         186.00
```

From this we can estimate a sensible lower threshold value for the area parameter of the **analyse** command, for example, 250.

Re-analyse the picture using the command:

```
analyse 3 to 8 segment 9 ge r le r2 area 250
```

The **analyse** command now reports a smaller number of particles:

```
Number of particles found: 8
```

help

If you see the message *Disk fragmented* or *Disk full* when you try to re-analyse the picture, use the **compress** command to reduce fragmentation and delete the old segmentation map and particle parameter list, pictures 8 and 9, to create more disk space.

Interpreting the analysis (2)

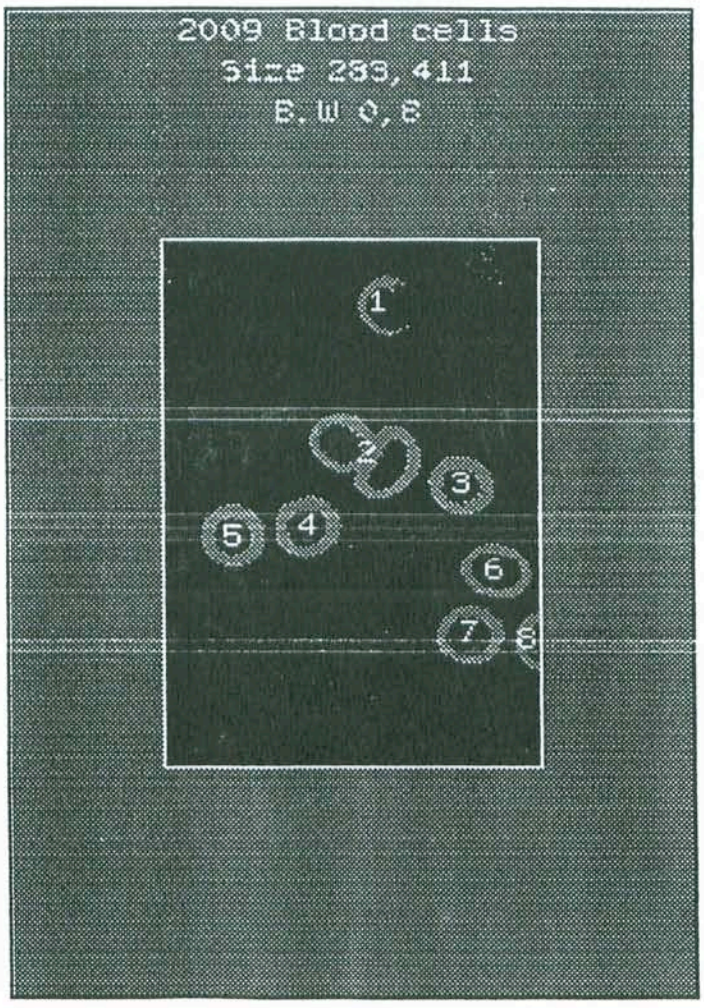
To discover which particles have been found by **analyse**, display the new segmentation map using the following command:

```
pshow;  min=0 max=255 display noscale from 9 to 1
```

As you can see, **analyse** has discounted the background particles of a smaller area and has successfully isolated the blood cells. It is now possible to query Semper about the different parameters of each particle. Before we do this, however, it is worth noting that the **analyse** command has counted two particles with edges that touch as a single particle. For example, try typing the following commands:

```
overlay red
pmark display cm id
```

These commands mark the centre of mass of each particle with its particle identifier, as shown below:



The following section describes how to separate the two touching particles using Semper.

Separating the particles

To define the two particles so that they are recognised as distinct by Semper we require a way of separating them. Semper provides the **pdraw** command which allows you to use the cursor to draw a line on the display that separates two particles by resetting the pixels along the line. (Note that you can also use **pdraw** to join a number of particles into one particle).

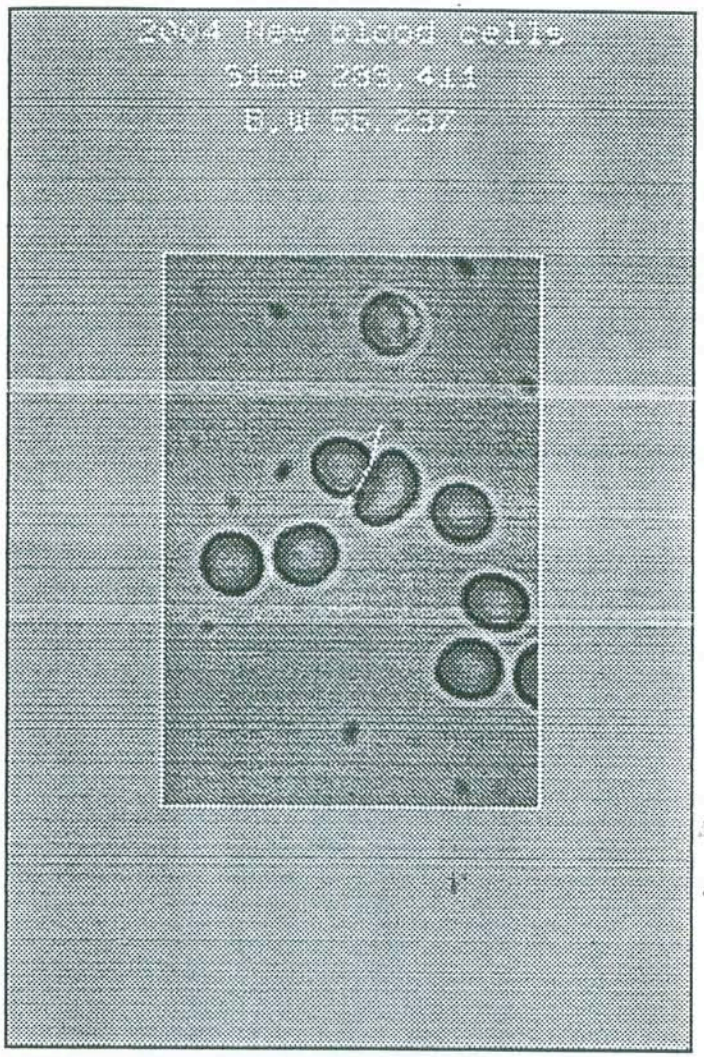
As the **pdraw** command alters the pixels of a picture, you need to make a copy of the original picture. (The original picture cannot be altered as it is write-protected and may be required by other users of the tutorial software). The following commands make a copy of picture 3, *Blood cells*, rename the new picture, reset the look-up table to monochrome and display the picture on the screen:

```
copy from 3 to 4
title 4 text 'New blood cells'
lut reset
display from 4 to display:1
```

To separate the two joined particles, type the following command and draw a line between the particles on the display, as illustrated overleaf. To start the line, click the left mouse button once at the required start position and end the line by clicking twice with the mouse button.

```
pdraw display:1 4
```

This command separates the two particles and writes the result to disk picture 4.



You can also use the Semper morphology commands **erode** and **dilate** to separate particles. The **erode** command removes a one-pixel wide border from each object on the display that is made up of non-zero pixels (that is, particles) to help clarify the relationship between particles. The **dilate** command adds a one-pixel border to objects made up out of non-zero pixels and is used to clean up an object by filling holes and points in an object without affecting its real edge.

You can also specify the **separately** option with **dilate** so that Semper dilates objects whilst preserving their separation from other objects. If you would like to try these commands, you need to use a segmentation map on the display as a source picture, as **erode** and **dilate** require a binary picture as source.

To verify that **pdraw** has efficiently separated the two joining particles we can run the following check by repeating the **analyse** command using picture 4 as the source, noting the new number of particles found and displaying the new segmentation map with suitable annotation.

```
analyse 4 to 8 segment 9 ge r le r2 area 250
```

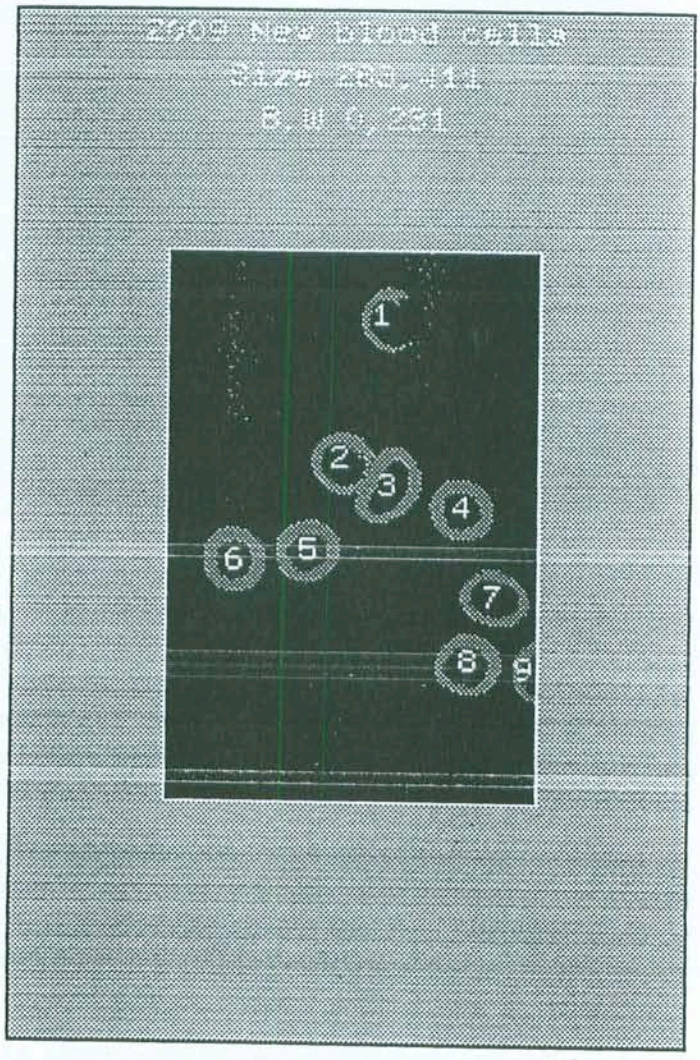
Semper should report:

```
Number of particles found: 9
```

To display the annotated segmentation map, type:

```
pshow; min=0 max=255 display noscale from 9 to display:1
pmark display cm id
```

The new segmentation map is shown below.



Further analysis

We can now perform further analysis on the differentiated particles. The **analyse** command records 25 individual parameters for each particle, these parameters are stored in Semper variables. This list of parameters is held in the particle parameter list (*ppl*) produced by **analyse** which in this example is held in picture 8. The table below lists the recorded parameters and their corresponding variables.

Name	Var	Description	Name	Var	Description
xref, yref	*xr, yr*	reference point	aferet, bferet	*af, bf*	feret diameters – 45° 135°
id	*pid*	particle identifier	hproj, vproj	*hp, vp*	horizontal and vertical projections
parent	*pa*	parent identifier			
holes	*h*	number of holes	perimeter	*p*	perimeter
background	*bg*	background flag	area	*a*	area
contact	*ec*	edge contact flag	xcen, ycen	*xc, yc*	centre of area
xmin, xmax	*x1, x2*	limits – min, max x	mmin, mmax	*m1, m2*	principal second moments of area – min, max
ymin, ymax	*y1, y2*	limits – min, max y	angle	*theta*	orientation
hferet, vferet	*hf, vf*	horizontal and vertical projections	circularity	*c*	circularity

In this instance we will find the results for the area, perimeter, circularity and orientation (angle) of selected particles. Type the following command at the console:

```
ptype id area perimeter circularity angle
```

This command displays a table showing the values for these parameters for all the particles found by **analyse**:

Index	Particle id	Perimeter	Area	Angle (degs)	Circularity
1	1.	237.693	829.0	86.395	0.18439
2	2.	268.930	1229.0	40.105	0.21354
3	3.	283.442	1556.0	58.585	0.24338
4	4.	227.633	1372.0	-27.358	0.33273
5	5.	255.037	1378.0	14.555	0.26623
6	6.	239.008	1384.0	-48.281	0.30445
7	7.	242.673	1345.0	-16.550	0.28701
8	8.	284.556	1333.0	1.239	0.20687
9	9.	145.795	616.0	87.489	0.36417

To leave Semper, type the command:

```
stop
```

Summary

The table below presents a summary of the Semper commands that you used in this tutorial:

Action	Command Used
Highlighting a threshold for analysis	**ladjust**
Performing particle analysis on an image	**analyse**
Displaying a segmentation map	**pshow**
Identifying a particle at a specific position	**pid**
Typing parameters for a particle	**ptype**
Marking particles on the display	**pmark**
Drawing a line to separate or join particles	**pdraw**

Tutorial 6

PROGRAMMING WITH SEMPER

- ❑ Creating a Semper macro
- ❑ Creating a Semper program
- ❑ Adding a program to a program library
- ❑ Running a program
- ❑ Creating a program as a text file
- ❑ Creating your own program library
- ❑ Programming tips

Overview

This tutorial describes how to combine individual Semper commands to suit your particular application. The ability to create a *program* or a *macro file* from a sequence of Semper commands allows you to devise your own solutions to image processing problems and to store these solutions for further use.

There are two ways of combining a sequence of Semper commands:

- as a macro
- as a program

A *macro* is a sequence of Semper commands that is contained within a single input line. A *program* is a sequence of commands that is stored in a file and may run over several lines. These two methods of combining commands are described in the following sections.

Start Semper by following the usual procedure, but you do not need to display a picture at this stage.

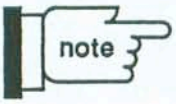
Creating a Semper macro

Use the **macro** command to tell Semper to store the rest of the input line as a macro. Macros are a cheap and cheerful way of storing a sequence of commands that you perform often, as they help to save time and typing. Use a semi-colon to separate commands on a single input line. For example, type the following command at the Semper prompt:

```
macro 100; ask 'Picture no? 'n; display n; negate n to display
```

To run this macro, type:

```
@100
```



Note that Semper stores macros on disk as pictures, using the picture class *Macro*. So if you were now to create a picture 100 you would, in effect, overwrite macro 100. To confirm that your macro is stored as a picture, type the command **examine all**. You will see a list of disk pictures, including the following:

```
2100  Size 42,1,1    0.0kb    Macro    Byte
```

For further information on the different classes of Semper pictures, refer to *Appendix B, Picture Types*.

Semper has a number of standard macros that you can use – type **show macros** for a list of these. Standard Semper macros are *named* to distinguish them from numbered macros of your own creation. For example, the macro called *@xy* is shorthand for **position x, y**.

Note that you can use the **edit** command to alter the text of an existing numbered macro or to create a new macro. Use the **examine macros** command to see all the numbered macros on your current device and **list** to list the text of a specified macro.

Creating a Semper program

Semper also allows you to store sequences of commands in a program file, which is not restricted to a single input line. There are two ways of creating a Semper program:

- interactively, within Semper
- using a text editor, outside Semper

These two methods are described in the following sections.

Creating a program within Semper

Semper allows you to create a program interactively, using the **add** command. It is only advisable to use this method if you are creating short, simple programs because, unlike a file created within a text editor, mistakes cannot be corrected and you would need to create a new program if you made a simple typing mistake.

To create a Semper program interactively, type the following command at the console:

```
add
```

Semper responds by asking for a name for the new program – type a suitable name within quotes. (Note that in the following examples, Semper prompts are shown in **bold**).

```
Program name (as textstring): 'test1'
```

Semper responds with a prompt which allows you to enter the text of a program, line by line:

```
Program Input->
```

At this prompt type the following command and press the *<return>* key at the end of the line:

```
Program Input-> type 'Hello world!'
```

To end the program and return to the usual Semper prompt, type the command **end.**

```
Program Input-> end
```

Semper now displays the message: *Program 'test1' inserted on device n,* where *n* corresponds to your program library device number. (Semper adds programs to a special device called a *program library).* To run the program, type the command:

```
library test1
```

Creating a program outside Semper

For more serious applications, you would probably need to write a Semper program in an environment where you can edit text and add lines. You can do this by creating a text file outside Semper, using the line editor of your operating system, for example, or a popular wordprocessor such as *Wordstar.* The procedure is similar to that described above, with two exceptions:

1. In order for Semper to recognise your text file as a program you need to give it the extension *.spl.* For example, *test1.spl, test2.spl,* where *spl* is an abbreviation for *source program library.*

2. The text file must begin with the name of the program, followed by a pair of empty brackets, for example:

```
test2()
```

The following example shows a program created using a standard text editor. In this program a sequence of Semper commands is used to cut a number of pictures and paste them into a single picture to create a montage of images. This can be useful if you are transferring images to a desktop publishing system or to line up images taken from a microscope.

Leave the tutorial software by typing:

```
stop
```

Using a text editor with which you are familiar, create a text file containing the following Semper commands:

```
montage()
! Create a montage of pictures
local t
! Set the display partitions and create a display picture
erase frame
```

```
cls
partition 1 size 768, 512
create 1001 size 768, 512 byte
! Ask for user input
ask 'How many pictures would you like to place in the montage? ' n
type ''
for i=1,n
type 'Place the cursor where you would like the centre of the image to be'
type ''
! Use the cursor to mark the centre of the picture
xwires display:1 noverify
ask 'What is the picture number of the image to be placed here? ' p
type ''
! Paste the picture onto the display
trap=30 paste p to display:1 @xy
! Trap the error return code (rc) in the local variable t
t=rc
if t=30 type 'This picture does not exist. Terminating program'
if t=30 return
cls
loop
! Store the montage or exit the program
ask 'Do you wish to store the montage? (yes/return) 'n
unless set(n) return
unless n=yes return
type ''
ask 'In which picture number would you like to store the montage? 'q
trap=11 copy display:1 to q
if rc=11 type 'There is not enough disk space for the picture'
end
```

Save this file under the name *montage.spl* and restart the tutorial software by typing:

```
semper /run=tutorial
```

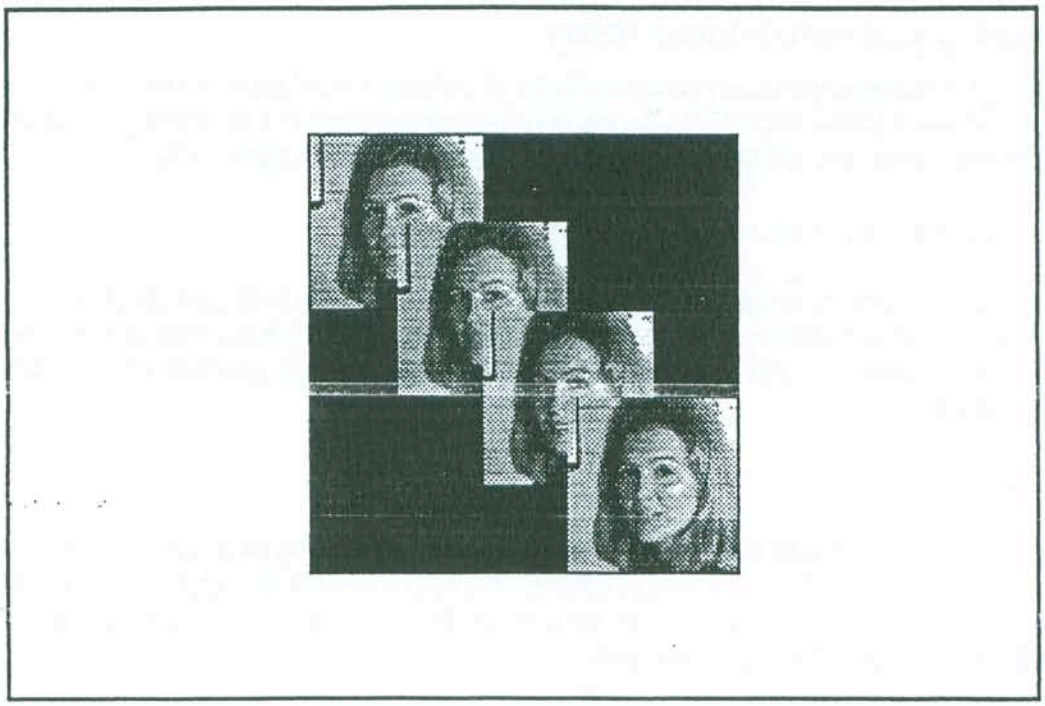
To use the montage program within Semper, you need to add the program to a Semper program library, using the command:

```
add name 'montage.spl'
```

To run the program, type the command:

```
library montage
```

On running the program you can create a montage picture that is similar to the one shown overleaf.



A note about error trapping

Note that in the *montage* program a number of error traps are employed to trap forseeable error situations, for example, lack of disk space to store a picture. If an error does occur in a program and it is not trapped, Semper ends the program abruptly, displaying the corresponding error message. To prevent this situation when programming, use the **trap** command to trap errors and to supply more helpful error messages to the user of the program. For example, the command **trap=30** traps the error *Picture <n> does not exist* which, without a trap, would cause the program to abandon its processing.

Note that the command:

```
trap=-1
```

traps all errors within a program. The **trap** command also records the number of any error that occurs in the Semper variable *rc* (return code), so that you can test for an error condition.

To find out about an error, type the following command:

```
help ?n
```

where *n* corresponds to the error number, or simply type:

```
help errors
```

Creating your own program library

Although Semper automatically assigns a library at the beginning of each session in which you can store programs (called *SEMPER.LIB*), you may like to create your own library for your collection of programs. To do this, use the **assign** command to assign a new program library:

```
assign program name 'mylib' new size 1
```

Semper returns the message *Device n assigned, Device n initialised with a 64 slot directory*, where *n* is the number of the next free Semper device – for the purposes of this example, device 5. Now set the current device to this device containing the newly created program library, using the assignment:

```
cd=5
```

You can now use the **add** command to create a program interactively or to add a program that you have created as a text file, but you need to specify the device number of your program library so that Semper can differentiate between the libraries *SEMPER.LIB* and *MYLIB.LIB*. For example, to create a new program interactively, type:

```
add device 5
```

or to add an existing program to the library, type:

```
add name 'montage.spl' device 5
```

To end this Semper session, type the command:

```
stop
```

Programming tips

1. There are a number of commands that specifically help you to program in Semper. These commands are:

• **for, loop, next**	to repeat a group of commands
• **break, jump**	to branch to part of a program
• **if, uniess**	to make an action conditional
• **list**	to list a program text
• **rename**	to rename a program within a program library
• **trap**	to trap errors within a program
• **wait**	to suspend program execution

2. Semper only reports errors in a program when you run the program using the **library** command. (It does not report an error at the time of typing a program line or when you store a program using **add**). For this reason, it is a good idea to keep programs short and simple in the first stages of devising a solution to a problem and to split a complex application into a number of small programs, that can be combined into a larger program once they are tested.

3. There are a number of pre-defined Semper programs in *SEMPER.PLB* that contain routines for common applications. Type **show programs** for a list of these programs. You may like to **list** one of these programs to see an example of programming in Semper.

4. Text that is written after an exclamation mark in a program is treated as a comment (expanatory text) and will not be a executed by Semper. Note that if you place a comment on a line that contains a command you need to place a semi-colon before the exclamation mark that introduces the comment, for example:

```
assign display
assign name 'mydisc';    ! assigns the picture disc 'mydisc'
```

Summary

The table below presents a summary of the Semper commands that you used in this tutorial:

Action	Command Used
Creating a macro	**macro**
Displaying a list of standard Semper macros	**show macros**
Creating/editing a macro	**edit**
Displaying a list of numbered macros	**examine macros**
Listing the text of a macro	**list**
Creating a program interactively	**add**
Running a program	**library**
Trapping errors within a program	**trap**
Adding a program to a Semper program library	**add...name**
Assigning a new program library	**assign...program**
Displaying a list of standard Semper programs	**show programs**

Conclusion

Running Semper

You have now completed the tutorials and can proceed with confidence to run a standard Semper session – without the constraints of working in the type of controlled environment provided by the tutorial software. To run Semper type the following command at your operating system prompt:

```
semper
```

Before you begin to use Semper to display pictures and perform image processing operations you will need to assign a display and various devices, such as a picture disc, help library and program library. You will also need to define display partitions and set the viewing conditions. (These functions are performed automatically by the tutorial software). The following section describes how to create a start-up file that will perform these functions for you each time you start Semper. This means that you do not need to type a series of similar commands each time you start a session.

Creating a Semper environment

You may like to set up your own environment for Semper by writing a short start-up file that is run each time you start Semper. An example of a typical start-up file is given below. You can modify it to suit your own requirements. (You may also like to list the file *TUTORIAL.RUN* to see the start-up file used by the tutorial software).

```
! Semper 6 -  set up commands

assign name 'pictures.dsk';          ! assign picture disk
cd=n;                                ! set the current device to the
                                     ! assigned device
assign display;                      ! assign the display device
assign help name 'semper.hlb';       ! assign help library
assign program name 'semper.plb';    ! assign shared program library
partition 1;                         ! define partition 1 (frame 1,
                                     ! look-up table 1)
lut 1 create; lut 2 create false;    ! create monochrome and false
                                     ! colour look-up tables for
                                     ! viewing
erase=yes;                           ! erase before displaying picture
```

Save this file as *SEMPER.RUN*, so that is it executed automatically at the **semper** command.

What next?

You may also like to experiment with the Semper visual user interface, which allows you to interact with Semper using icons such as panels and menus. An introduction to this interface is given in the manual:

Tutor User Guide

You can also adapt the user interface to your own requirements, by creating your own menus to suit a particular application. For details, refer to the manual:

User Interface Guide

These manuals are contained in the *Semper 6 Guide*.

Appendix A

TROUBLESHOOTING

Overview

This appendix provides general advice in case you encounter problems when you begin to use Semper. This appendix is divided into three sections. The first section describes what to do if you see an error message. The second section details common problems and suggests solutions. The third section includes a report form that you can use to register any complaints, suggestions or enhancements.

Semper error messages

If Semper displays an error you can find an explanation of the error using the on-line help, by typing **help** followed by a question mark and the error number, for example:

```
help ?41

help ?72
```

Alternatively, you can refer to *Appendix E, Error Messages* in Part 2 of the manual:

Semper 6 Command Reference

If you see the following message when you try to use the on-line help:

No help library assigned

you need to assign the standard Semper help library, by typing the command:

```
assign help name 'semper.hlb'
```

You will now be able to access the Semper on-line help.

Common problems

This section details common error situations and suggests possible solutions.

Display errors

A display error may be caused by:

- hardware failure in the display device or interface
- using false or full colour look-up tables in installations where the hardware does not support them
- using zoom or pan settings that are not supported by your display hardware

If your display is *rolling* or *flickering*, it may be because your system is configured for the wrong country. Type one of the following commands:

```
assign display ntsc
```

This causes the framestore to switch to American display standard.

```
assign display ccir
```

This causes the framestore to use the European TV standard.

If you are using a PC AT and your *display is blank* you may have a speed problem. Certain PC AT compatibles have design faults that prevent a Synergy framestore from working correctly at high speed and produce a display error. If you suspect this problem, try running your PC at slow speed. (Refer to your manufacturer's documentation for details). *Synoptics* can advise you of suitable AT compatibles which work correctly at high speed.

Disk storage problems

If you find that your *disk is full*, there are a number of ways round this problem:

- delete unwanted pictures using the **delete** command
- store existing pictures in a more compact form, for example, **copy 41 byte**
- create a new disk device with sufficient space, for example, using the command:
  ```
  assign new name 'newdisk' size 5000
  ```
- create new *temporary* disk space (available only for the session), using the command:
  ```
  assign new name 'tempdisk' size 5000 scratch
  ```

If you find that your *disk is fragmented*, you can use the **compress** command to compact the free space. You can also use the **directory** command to list the available free space on your current device.

If you find that you have *no free disk workspace*, you can also try **compress**ing the device. In this case, also ensure that you have **assign**ed a device or that your device is not write-protected (**wp off**).

Device errors

Problems with devices are common and may be concerned with:

- setting the current device. To make a device current you can assign a number to the Semper variable *cd* (current device). For example, the command:

  ```
  cd=4
  ```

  makes device 4 the current device.

- assigning a device. For example, you may receive an error message saying *'Device <n> is not assigned'*. In this case, you need to assign the specified device, for example:

  ```
  assign name 'mydevice'
  ```

- write-protected devices. You may find that a device that you would like to work on is write-protected (if your operating system only permits read access to a file). To assign the required device use the **assign** command with the **wp** option:

  ```
  assign wp 'arcdev'
  ```

To find out details of each device that you are using (number, name, size, blocks and directory size) type the command:

```
show devices
```

Partition problems

You may find that you see the error message *Display partition number does not exist* when you try to display a picture. This is because you need to define at least one display partition at the start of a Semper session so that pictures can be sent to an area of the display. (Note that if you are running the tutorial software, partitions are automatically set up for you). Type the command:

```
show partitions
```

to see if any partitions are defined for your display. To define a partition, use the **partition** command, for example:

```
partition display:3 size 256 top left
```

A standard Semper program, called *partitions*, also allows you to define a number of partitions easily. To run this program, type the command:

```
library partitions
```

Display picture vs. disk picture

When you use the tutorial software you may see the following error message:

Write protected device or picture number

This is because the pictures on your disk are write-protected so that they can be used repeatedly for tutorial purposes. Semper does not allow you to alter a write-protected picture. If it was not your original intention to change a tutorial picture, you may find that you are confused between *disk pictures* and *display pictures*. For example, the commands:

```
negate 3 to display

negate 3
```

are different in their effect. The first command shows a negated form of picture 3 on the display. The second command negates picture 3 that is stored on disk. If in doubt, always explicitly specify a source and output using the keys **from** and **to**. It also helps to be aware of the difference between, for example, the words *3* and *dis:3* in a command. The first specifies disk picture 3, the second specifies a defined area of the display.

Emergency exits from Semper

You may need, from time to time, to abandon a process within Semper (for example, a lengthy calculation). This is called an *abandon request* and can be performed by typing the following control characters at the keyboard:

CNTRL/C

Other characters which may be used, depending on your machine, are:

CNTRL/BREAK

or the interrupt character that is local to your machine.

In extreme circumstances, it may be necessary to make an emergency exit from Semper and return to the operating system, but this is not recommended and is to be avoided wherever possible. To leave Semper in this way, first try typing the control characters *CNTRL/C* sixteen times or more at the keyboard. If this does not take you to the operating system you could try the following characters:

- *CNTRL/ALT/DELETE* or *RESET* button (PC only)
- *CNTRL/Y* (VAX only, excluding DEC windows versions)
- CNTRL\ (control backslash) (VAX running DEC windows and Hewlett-Packard running X-windows)

Crashing out of Semper may mean that temporary files are left behind – for example scratch files and work files. If you have a PC these are easily recognisable as *ZZZZ* files and can be deleted. (If you are working on a network, take care not to delete a file that is attached to another user's current Semper session).

On installations other than PCs these temporary files will be cleared by the operating system after the next reboot or as part of its internal monitoring.

For information on the abandon process that is specific to your installation, type the command:

```
help abandon
```

Which framestore/installation?

You will often find references to your *framestore* or *installation* in this set of manuals. In particular, you may be directed to read *installation specific* information on, for example, Semper commands. If you are unsure of the type of framestore or installation you are using, try typing the following command:

```
help framestore
```

This command provides a general overview of your display, detailing any special features. For an overview of the extra commands that are provided with your version of Semper, type the command:

```
help extras
```

To find out more about your system, for example, input/output limits, the number of possible devices, you can use the Semper command:

```
show system
```

Customer report form

Please fill in and detach the form given overleaf if you find a fault with the following:

- software
- hardware
- documentation

Also use this form to record technical enquiries, enhancement requests, suggestions or comments. Send the form to the appropriate address given below:

Synoptics Ltd
271 Cambridge Science Park
Milton Road
Cambridge
CB4 4WE
UK
Tel: (0223) 423223
Tel: (0223) 420020

Synoptics Ltd
Paragon Towers
233 Needham Street
MA 02164
USA

Tel: 617 527 4461
Fax: 617 527 4084

The telephone number of the **Semper Support Line** is:

(0223) 420267

Appendix B

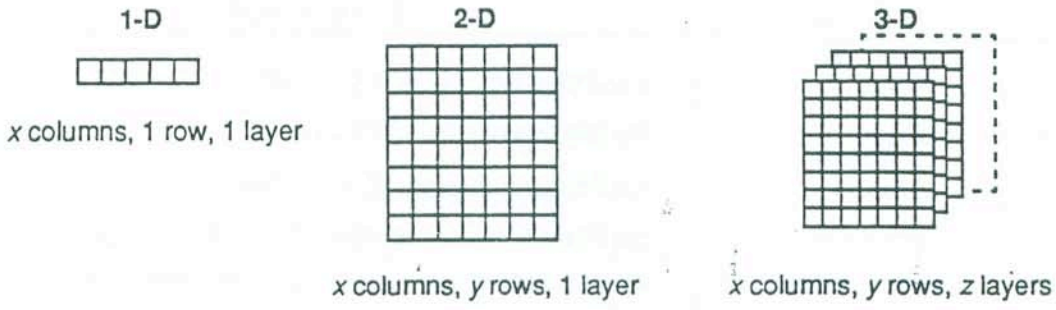
PICTURE TYPES

Overview

This appendix gives a summary of Semper picture types and explains some of the terms used in this manual. It details picture dimensions, class and form.

Picture dimensions

A Semper picture can have 1, 2 or 3 dimensions, as is illustrated in the following diagram:



Note that only some Semper commands support multi-layer (3-D) and 1-D as well as 2-D operations.

Creating a multi-layer picture

You can create a mulit-layer (3-D) picture, using the **create** command, for example:

```
create 4 size 768,512,3 fp
```

If you have a true colour system and have three separate 2-D pictures that can be used as the red, green and blue components of a picture, you can create a multi-layer picture from these using the **stack** command, for example:

```
stack from 1,3 to 4
```

This command stacks all the layers of pictures 1 to 3 as picture 4.

Picture classes

Semper assigns a picture to a class according to its data type. The class indicates what kind of information a picture contains, for example, three pictures are supplied on the tutorial disk and these are classed as *Images*. As you work through the tutorials you will also note that you can create different classes of pictures, for example, *Histograms* in *Tutorial 2* or *Macros* in *Tutorial 6*. These are referred to by a picture number, even though a macro, for example, is not a picture as such.

Most of Semper's commands will work on any class of picture, but some are more selective and will give an error message if you try to apply them to a picture of an inappropriate class. Also, some commands do different things according to the class of picture they are given. For example, the **display** command portrays a 2-D image on the screen as a block of grey-level pixels, but draws a line graph to represent a histogram. For further information on picture classes, type the command:

```
help classes
```

The table below lists the ten different classes of Semper picture.

No	Class	Description
1	*Image*	a sampled picture (most data fall into this class)
2	*Macro*	the text of a numbered macro
3	*Fourier*	the Fourier transform (*ft*) of an image
4	*Spectrum*	the intensity (modulus squared) of the *ft* of a picture
5	*Correlation*	the cross or auto-correlation function between pictures
6	*Undefined*	any data not otherwise classifiable
7	*Walsh*	the Walsh/Hadamard transform of a picture
8	*Plist*	a list of position coordinates with associated information
9	*Histogram*	the histogram (of the pixels) of a picture
10	*Lut*	a framestore look-up table

Picture forms

Semper can store picture data in one of four different forms. Each form offers a different trade-off between storage and precision requirements. For example, at one extreme, the *Byte* form provides only 256 possible values for each pixel but is sufficient for display purposes and requires little disc space; at the other, *Complex* floating point requires eight times as much disc storage but is essential for work such as *Fourier* transforms. For further information about picture forms, type the command:

```
help forms
```

The table given overleaf describes the four types of picture storage. Note that the values of integer, fp and complex forms are, to some extent, machine dependent. The values given here may be considered as typical.

No	Form	Description
0	*Byte*	1byte storage. 0 to 255, unsigned integers only. A compact but restrictive form of storage as negative values can easily arise in processing
1	*Integer*	2 byte storage. –32768 to 32767 or wider, integers only. *Integer* storage is less restrictive than *Byte* but still compact
2	*Fp*	4 byte storage usually.1–35 to 1+35 or wider, either sign, integral or fractional (floating point). Typically one part in a million precision over the full range
3	*Complex*	8 byte storage usually. Pairs of *fp* values representing real and imaginary parts of a complex value

Further information

To see the dimension, class and form of a picture, for example picture number 2, type the following command:

```
examine 2
```

or type the command **examine all** for details of all pictures on a current device.

For further detail about how Semper stores its pictures, refer to *Chapter 4, Pictures* of the following manual:

Advanced Users' Guide

contained in the *Semper 6 Guide.*

Appendix C

GLOSSARY

Overview

This glossary provides definitions of the technical terms that are used in this manual to describe image processing operations and computer functions. The glossary is arranged in alphabetical order. Words that are in *italic* within the text indicate that an additional glossary entry exists for that term.

1-D picture

A picture with only one row, for example, a graph (x=>1, y=1, z=1).

2-D picture

A picture with rows and columns but only one *layer*, for example, an image (x=>1, y=>1, z=1).

3-D picture

A picture with rows, columns and more than one *layer*, for example, a colour image (x=>1, y=>1, z=>1).

Abandon request

A request made by a special keyboard character that tells Semper to abandon the current process and seek fresh instructions. An abandon request can also be made from Semper to the *operating system* – this type of request returns you to the *operating system prompt*.

Assignment

A statement that assigns a new value to a *variable*, for example, *cd=4*, *min=1*.

Byte

Pixel form using 8 bits, with possible values in the range 0 to 255.

Class

Name and number that codifies the kind of data held by a picture, for example, *Image*, *Histogram* and *Macro* are all different classes of picture.

Command

Single instruction to Semper, involving a *command* name with optional arguments, label, conditional clause and/or *assignment*.

Complex

Pixel form using two *floating point* values, used for imaginary as well as real parts, such as *Fourier transforms*.

Compiler

A number of *high-level languages*, for example, 'C' or Fortran, must be run through a *program* that translates them into machine code (instructions that a computer understands). This *program* is called a *compiler*.

Compress

A way of packing the separate sections of a disk to avoid *fragmentation*. Compression is used to create contiguous free space on disk.

Correlation

A *class* of Semper picture that stores the cross or auto-correlation function between pictures.

Current directory

The default directory. A *directory* is a storage area in which files are kept.

Data range

The numerical band occupied by the grey levels of a picture.

Destination

A picture which contains the *output* of a Semper process.

Device

A storage area which may contain Semper pictures or other data, for example, *programs* or help files. A Semper device can be a disk file, magnetic tape or the *display* and is identified by a number, for example, device 2.

Dimensions

Set of three values giving row length, column length and number of *layers* in a picture (x, y, z).

Directory

Storage area on magnetic media, usually disk, in which files are kept.

Display

An area on a monitor or in a window that is used to display Semper pictures. ***Display*** is also a *variable* for identifying the current display picture.

Dynamic range

Range of *grey scale* data in a picture that is capable of change.

Edge detection

Image processing technique used to delineate the boundary lines of features in an image.

Error trap

A method used to prevent an error from abruptly halting the processing of a Semper *program.*

Factorisable

A picture size that can be divided exactly by 64, for example, a picture of size 256 x 256.

False colour look-up table

Intensity transformation table mapping the *pixel* values stored in the *framestore* to different colours on the *display.*

Filter

Image processing technique used to remove unwanted elements from a picture, for example, noise.

Floating point

Positive or negative integral or fractional values.

Form

Name and number coding *pixel* representation (*Byte*, *Integer*, *Fp* or *Complex*).

Fourier picture

A *class* of Semper picture that contains the *Fourier transform* (*ft*) of an image.

Fourier transform

Two-dimensional transform used for a variety of image processing purposes, such as image enhancement and recovery. A *Fourier transform* creates the *power spectrum* of an image.

Fp

Floating point form allowing positive or negative integral or fractional values, very small or very large.

Fragment

Disk fragmentation is caused by parts of files scattered across the disk space. A fragmented disk may take a longer time to access and have less available free space than a disk which is *compressed.*

Frame

A surface of the Semper *display* that can contain images.

Framestore

A piece of hardware used to store and translate images. Certain types of framestores (called framegrabbers) are able to capture and digitise images, when connected to a camera.

Global key

Key accepted by all processing *commands.*

Global option

Option accepted by all processing *commands.*

Grey scale

The range of gradations of *intensity* (grey levels) in a picture.

Grey scale histogram

A histogram showing the distribution of *intensity* ranges of *pixels* in a picture.

Hardware

The items of physical equipment that make up a computer.

Help library

A disc file from which Semper retrieves information for use by the **help** command.

High-level language

A programming language designed to facilitate the creation of specific applications.

Histogram

A *class* of Semper picture containing a picture *intensity* histogram.

Icon

A graphic symbol representing an element of the *user interface* system, for example, a *menu* or *panel*.

Image

A Semper picture containing sampled data.

Initialise

To initialise a disk is to clear the contents of the disk.

Integer

Pixel form normally using 16 or 32 bits, allowing positive or negative integer values.

Intensity

The brightness or colour of a *pixel*.

Interface

A shared boundary between computer *hardware* or between the computer and its users.

Interpreter

An interpreter is a *program* that translates a *high-level programming language* into machine readable instructions.

Key

Assignment to a processing *command* with a numerical value.

Kilobyte

1024 bytes. A byte is a unit of information storage used by a computer.

Layer

A component of a *3-D picture*. A *2-D picture*, which comprises *pixels* organised in rows and columns, may be stacked to form a *3-D picture*, with *layers* numbered 1,2... from the back.

Library

Collection of Semper files, for example, *programs* or help entries.

Look-up table

Intensity transformation table, mapping the *pixel* values stored in the *framestore* to the colours/*intensities* presented on the *display*.

Lut

A *class* of Semper picture storing a *framestore look-up table*. *Lut* is also used as an abbrieviation for *look-up table*.

Macro

A stored sequence of Semper *commands* (the length of a single input line) that can be executed by reference to a name or number.

Macro picture

A *class* of Semper picture containing the text of a numbered *macro*.

Megabyte

1048576 bytes (1024 Kilobytes). A byte is a unit of information storage used by a computer.

Menu

A list of functions that are available to the user. An element of the *user interface* system.

Monochrome look-up table

An *intensity* transformation table, mapping the *pixel* values stored in the *framestore* to the *grey scale* presented on the *display*.

Morphology

An area of image processing that is concerned with the shape characteristics of features of an image.

On-line help

Help information that is available at the terminal screen by typing the Semper **help** *command*.

Operating system

A *program* or number of *programs* which creates the environment in which a user can communicate directly with a computer.

Option

Argument to a *command* expecting the value ***yes*** or ***no***.

Output

The result of a Semper process.

Overlay

Additional *display* memory 1 bit deep, allowing display text and graphics to be defined independently of the picture storage.

Panel

A basic building block of the *user interface* system. An area of the interface *display* that can contain *menus* and other *user interface* elements.

Particle Parameter List

A list recorded by the **analyse** command that contains 25 individual measurements for each particle found during the analysis.

Partition

A defined area of the *display* that is used to display an image.

Peak

The highest point of *intensity* in an image.

Peripheral

An additional *hardware* item, for example, a printer, that can be connected to a computer.

Pixel

Picture element. One of the numerical values that make up a picture to represent brightness or colour.

Plist

A *class* of Semper picture that contains a list of position coordinates with associated information.

Portable

A *program* that is easy to transfer between computers.

Power spectrum

The result of a *Fourier transform*, where parts of the image are arranged according to the *intensity* of *pixels* in the original image.

Primitive routine

Low-level segment of programming code in a language that is machine readable.

Program

Self-contained sequence of Semper *commands* invoked from the terminal by a **library** *command*.

Program library

Disc file containing *programs*, assigned as a *device*.

Prompt

Characters on the terminal, followed by a cursor, which indicates to the user that the computer will respond to a typed *command*. The Semper prompt is ***S$***.

Return code

Error code produced by the previous *command*, stored in the Semper variable ***rc***.

Segmentation map

A binary picture produced by Semper as a result of particle analysis. The map shows the particles that were found during analysis, removing other detail from the picture.

Sharpen

An image processing technique to throw into relief the fine detail of a picture.

Software

The *programs* that are run on a computer.

Start-up file

File of *commands* used to initialise the environment for a Semper session.

Subregion

A defined area of a Semper picture.

Subroutine

A section of a programming language that is used as a procedure to perform repetitive tasks.

Syntax

A definition of a programming language that conforms to "grammatical" rules.

Synthetic image

A mathematically generated image created by an arrangement of *pixels*.

Threshold

A limit beyond which particles are not identified by the **analyse** command.

True colour look-up table

Intensity transformation table, mapping the *pixel* values stored in the *framestore* to three separate *display* frames which are presented as the red, green and blue components of a colour image.

Undefined

A *class* of Semper picture that contains any data which is not otherwise classifiable.

User account

Details stored on a computer about a user, including available free space and access rights to files.

User interface

A computer *program* that communicates with a user in a way that is easy to understand and use, for example, using *menus* or *icons*.

Variable

Name with an associated (usually changeable) numerical value.

Walsh

A *class* of Semper picture that contains the *Walsh/Hadamard* transform of a picture.

Write-protected, wp

A picture or *device* which is locked so that data can be read and used, but not modified.

Synoptics — Customer Report Form

For Synoptics use only		
Reference number		
	Name	Date
Received / Taken by		
Acknowledge		
Start		
Finish		
Reply		

Name _____ Date _____

Company _____ Your ref. _____

Address _____

_____

_____

_____

Telephone _____ Ext. _____

Installation: Software _____ Serial number _____

Computer _____ Operating system _____

Framestore _____ Serial number _____

Please indicate the nature of this report:

- ☐ Software fault
- ☐ Hardware fault
- ☐ Documentation fault
- ☐ Technical enquiry
- ☐ Request for enhancement
- ☐ Suggestion or comment

If reporting a fault, please indicate its priority:

- ☐ High • System does not start up or aborts
- ☐ Medium • System works but fault cannot be avoided
- ☐ Low • Fault can be avoided
- ☐ None • Fault is mostly harmless

☐ Supporting information is attached (please itemise below)

Description (please be as precise and informative as possible)

☐ Description continues overleaf

Synoptics Ltd. 271 Cambridge Science Park Milton Road Cambridge CB4 4WE Telephone (0223) 423223 Fax (0223) 420020

Description (continued)