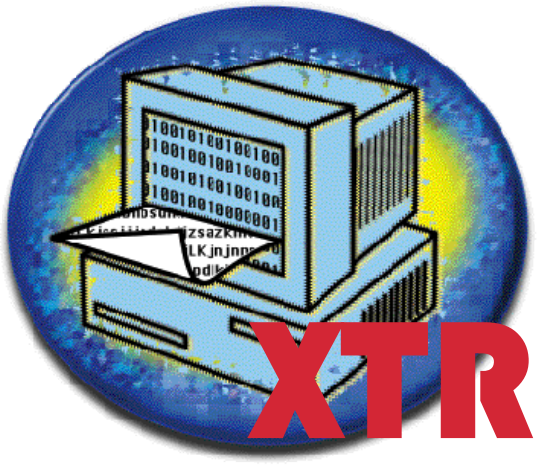
OCR Shop® XTR

Users Guide



OCR Shop XTR quickly and accurately converts document images into readable text in UNIX and Linux

For version 6.0

VIVIDATA
IMAGING SOFTWARE

Copyright Notices

use it at another location (except under special contract), it may do so giving prior written notice to Vividata LLC, specifying the type of computer and the new site. U.S. Government personnel using this Software, other than under DoD contract or GSA Schedule, are herby on notice that use of this Software is subject to restrictions which are the same as or similar to those specified above.

Request for Comments

In our effort to provide you with the best documentation possible, we welcome any comments and suggestions you may have about our products. Please direct communications to us at:

Vividata LLC
721 Cragmont Ave.
Berkeley, CA 94708
U.S.A.
Phone: (510) 658-6587
E-mail: http://www.vividata.com/support_contact.html
World Wide Web: http://www.vividata.com

Table of Contents

Chapter 1: Before You Begin

OCR Shop XTR Overview

OCR Shop XTR quickly and accurately turns printed pages and faxes into editable documents that you can use with your favorite programs.

The technology used is Optical Character Recognition (OCR). During OCR, OCR Shop XTR looks for and defines characters in an image to produce text that you can revise without retyping. You can export the recognized text from OCR Shop XTR for use in a wide variety of word processing, page layout, and spreadsheet programs.

General description

OCR Shop XTR utilizes Scansoft SDK 5.0 OCR technology to accomplish its fast and accurate optical character recognition.

OCR stands for optical character recognition: the process of recognizing text from images of printed pages into a computer file that can be indexed or edited without retyping.

A scanner is more than a copy machine that simply transfers an image into your computer. Rather, it translates a page into data by dividing up the image into millions of dots or pixels (usually from 40,000 to 90,000 per square inch) and then assigns a value to each, depending upon whether it is inked, partially inked, or blank.

The composite document stored in your computer is the map of these dots, that is, a bitmap. Your computer sees this data not as editable text, but as one bitmapped image.

OCR then, is the process of translating this bitmap into editable text. Text characters are designed by assigning a code corresponding to each key on the keyboard, be it a letter, number or symbol. There are a variety of different code sets in use, but the most common code set is the ASCII (American Standard Code for Information Interchange) table of character equivalents. ASCII is generally recognized as the universal code for most computers. Almost every program that uses text and/or numbers understands ASCII.

Prior to 1988, matrix-matching, the process by which a bitmap's shape is compared to a library of character shapes, was the only method of text recognition. Because matrix-matching required exact matches, it only worked for a small number of fonts and sizes. Thus, it was neither widely used nor recognized as a useful process. This changed when Caere released OmniPage, a page-recognition program that incorporated OCR technology based on feature-analysis. Now, the process involved individual character features being analyzed for recognition rather than matrix-matching for shapes as earlier OCR had done. Caere was acquired by ScanSoft in 2000 and its technology incorporated into ScanSoft's offerings.

Now in OCR Shop XTR, Vividata utilizes Nuance Communications' proprietary OCR Engine technology.

Type Conventions

Different kinds of typefaces used throughout this manual indicate text that will appear on the screen or need to be entered by the user.

Type:	Indicates text is:
`courier`	text generated by the computer
`courier bold`	text typed in by user
`<brackets>`	text to be replaced by user

When asked to enter commands preceded by a pound sign (‘#’), the user should be in super-user mode or ‘root’ first. (The command to be entered does not include the pound sign itself.)

System Requirements

OCR Shop XTR is available for a variety of Unix-based workstations. The following platforms are currently supported:

Manufacturer	Operating System / CPU
Sun/Oracle	Solaris SPARC (Solaris 10)
Linux: RedHat, Mandrake, etc.	Linux x86 (Kernel 3.2 and higher)

Table 1: Supported Platforms

If your platform is not listed above, you can contact Vividata, Inc. to see if your platform has been added since this printing of the manual.

Customer Support

You can reach the Vividata, Inc. technical support staff by:

- Online email form: http://www.vividata.com/support_contact.html
- Telephone: USA (510) 658-6587

Customer Service is available on regular business days from 8:00 AM to 5:00 PM (PST/PDT).

Chapter 2: Software Installation

Overview

This section describes the installation procedures for OCR Shop XTR, including the License Manager. Please consult the release notes supplied with the product for any last-minute information relevant to your particular system.

Installing OCR Shop XTR

Installing OCR Shop XTR on your system consists of a few simple steps. You may have obtained your OCR Shop XTR distribution either from the internet or from a CD-ROM. In both cases, you should have a OCR Shop XTR distribution file called, "**<product>-<platform>-<version>**". The file name will vary depending on the product name, operating system, release number.

Vividata's installer is a text-based installer and does not require a graphical interface or user interaction.

Installing OCR Shop XTR from the Distribution File

- For a CD distribution, mount the CD-ROM.
- As root, change to the directory containing the distribution. For a CD, this is the top level directory; for an internet download, it is whereever you saved the download.

```
# su              (become root)
# cd /mnt/cdrom   (or the saved location for a download)
```

- Run the self-extracting executable:

```
# ./<product>-<platform>-<version>
```

You will see output similar to this:

```
Extracting...
Installing...
Killing currently running licensing and <product>
processes...
```

Files are installed in /opt/Vividata, approximately 16 MB of space is needed there.

If you wish to install the software in a different directory, you may do so with most Vividata products by setting the environment variable VV_HOME to the desired directory prior to running the self-extracting executable. Please see "Configuring the Environment" on page 8 for more information on setting up your environment.

Installing the License Keys

Vividata normally distributes license keys through the Vividata website or by email. The encoded license key string is typically wrapped within a self-installing shell script. To install the license key using the self-installing script, run the script:

```
# sh vvkey.sh
```

The filename of the script may vary.

The license key will be installed in /opt/Vividata/config/vvlicense.dat. If you received a license key on paper, you must manually install it in this file.

If your Vividata software is installed in a directory other than the default /opt/Vividata, you must set the environment variable VV_HOME to the Vividata installation directory prior to installing the license key. Please see "Configuring the Environment" on page 8 for more information on setting up your environment.

Configuring the Environment

Please see "Configuring the Environment" on page 8 for details on setting up your environment.

Installation Complete

You are now ready to use OCR Shop XTR.

Removing OCR Shop XTR

Should it be necessary to remove OCR Shop XTR from your system, become root and execute the following commands:

```
# rm -r <vividata install directory>
```

Configuring the Environment

Environment Variables

A number of environment variables affect the operation of OCR Shop XTR. These are normally either unnecessary or set automatically during installation, but you may want to change their default values if you are customizing your system. If you would like to check on their settings, you can inspect the wrapper script(s) in $VV_HOME/bin. An explanation of each environment variable follows:

VV_HOME is the location where OCR Shop XTR is installed, by default /opt/Vividata. When run, OCR Shop XTR assumes the installation directory is the default; VV_HOME must only be set if the Vividata software is installed in an alternate directory.

VV_IGNORE_FILLORDER controls whether the TIFF fillorder bit is obeyed or ignored when OCR Shop XTR reads an input TIFF image file. OCR Shop XTR ignores the TIFF fillorder bit by default, but the user may set this environment variable to “y” or “n” if they wish to change the behavior. The command-line option, “-ignore_tiff_fillorder” controls the same behavior; see page 34. We recommend not setting either the environment variable or the command-line option, unless you know that the TIFF fillorder bit should be obeyed, a rare occurrence. If the TIFF fillorder bit should be obeyed, we recommend setting either the command-line option or the environment variable, but not both, in order to avoid confusion.

Setting the Environment Variables

You can set the appropriate environment variable(s) in your .cshrc or .profile file. When the next C or Bourne shell is started, its environment will be automatically configured for Vividata's environment variables.

You can also add the name of the directory that contains OCR Shop XTR to the PATH environment variable assignment in your .cshrc or .profile file. This will allow you to launch the application from any directory.

After modifying your .cshrc or .profile file, logout from the system and then login again to start your session with the modified initialization files.

Chapter 3: Starting Up OCR Shop XTR

Overview

Vividata's OCR Shop XTR provides sophisticated character recognition operations in Unix and Linux environments. The OCR Shop XTR engine ("OCR Engine") is based on the engine contained within the ScanSoft SDK 5.0. Vividata, under license from Scansoft and Nuance Communications, has ported the OCR engine to Unix and Linux environments and added additional image processing options.

In running OCR Shop XTR, there are a number of ways to speed up recognition, increase accuracy, and streamline OCR workflow. These are described in detail the following chapters of this manual.

Running OCR Shop XTR

First, to run OCR Shop XTR using the current settings and the environment variable $VV_HOME, invoke (the absolute path name of) the command ocrxtr followed by (zero or more) parameters followed by the files to recognize:

```
$VV_HOME/bin/ocrxtr [<-parameter=value>]* [<filename>]*
```

If the OCR Shop XTR bin directory "$VV_HOME/bin" was included in the variable assignment of your PATH environment variable, then you may begin running the program from any directory. simply by typing ocrxtr followed by parameters and names of document files to recognize.

For example,

```
ocrxtr -out_text_format=pdf -language=german test.tif
```

will perform recognition using the German language pack and output the results in pdf format for the file test.tif (in the current directory).

File names must be specified last on the command line.

Examples throughout this manual will assume that your PATH environment variable has been set to include "$VV_HOME/bin."

Parameters

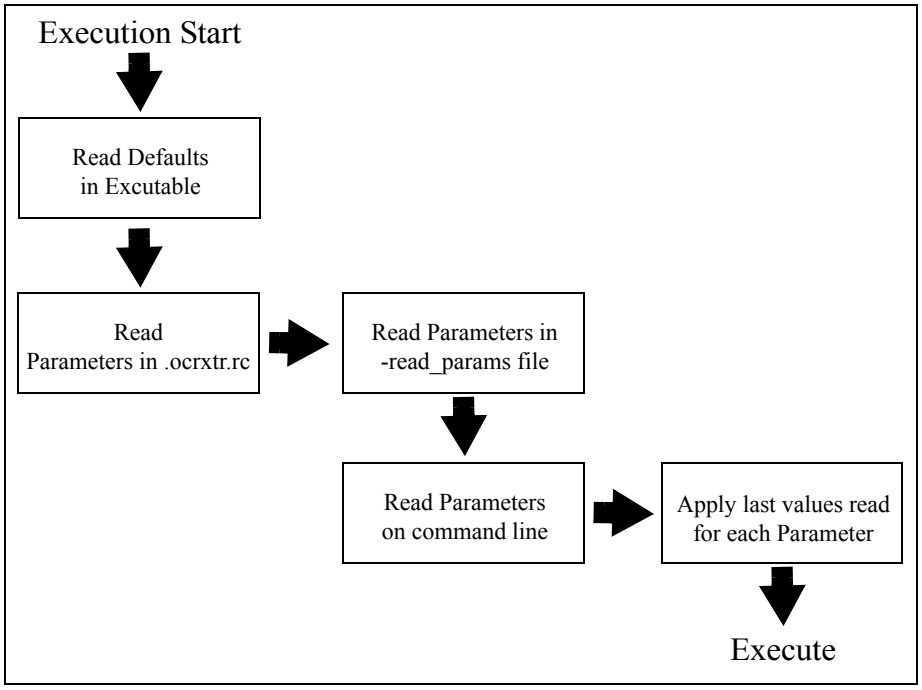
There are over 50 parameters that can be set and modified when running OCR Shop XTR. In most cases, the default settings for the parameters (as specified in the detailed documentation in the following chapters) can be used and the result will be high quality recognition.

Specifying invalid parameter names will cause the engine in most cases to immediately exit with a failure status.

Parameters are set in the following locations:

- Defaults within the executable
- ocrxtr.rc resource file, if any, in the home directory of the current user
- File specified by the read_params parameter, if any. Such a file can be written out by OCR Shop XTR using the write_params parameter.
- Parameters specified on the command line

Figure 3-1 Parameter Execution Order



The order of precedence of parameters is shown in the above diagram.

> Within each step, parameters are applied in the order specified, i.e., for files, parameters specified later in a file will override those specified earlier in a file. For command line parameters, parameters are applied left to right.

Parameter Files

Parameter files can have any name you choose. A parameter file contains one parameter per line. Parameter files must follow these rules:

- One parameter per line.
- Lines begin with the parameter name (not a hyphen "-")
- Lines beginning with "#" are ignored.
- Blank lines are ignored.

Using lines beginning with a pound sign, "#", you can add comments to your parameter files to remember setting choices that you've made.

Parameter Value Glossary

The following is a list of standard parameter values that are used throughout the description of parameters.

Table 3-1:Parameter Values

Parameter Value Name	Meaning
<char>	Single ASCII character; case sensitive.
<dpi> Example values: 200	Dots per inch (dpi). This is the dots per inch resolution of image data.
<filename> Examples: foo im.tif	This is the name of a file. The name can be either an absolute pathname or a relative pathname. Relative pathnames are relative to the current directory.
<float> Example values: 17.9 0.4	This is a floating point number of the format: [0-9]*[0-9].[0-9]*
info_log	This represents the information log to which information is output. Where used as a value indicates that output of the parameter is redirected to wherever the info_log parameter has directed output.

Table 3-1:Parameter Values (Continued)

Parameter Value Name	Meaning
<k1-k2> Examples: <0-100>; valid value: 40	This represents a range of integers, inclusive, from the first value k1 to the last k2. The example given indicates that the acceptable range of values is between 0 and 100 inclusive.
<value_1|value_2|…|value_n> Example: <full|preprocess_only|N>; valid values are full, preprocessing_only, and N	This represents a choice of possible values. Any of the values listed will be valid.
<out_filename> Example values: output.%s.%d.txt results%d	This is a filename used for output that can contain the special strings listed below to create the resultant filename(s). Special strings include: • %d anywhere in name to indicate the file number. The file number prints as 3 digits, with leading zeroes. The first file number is equal to the value of the parameter start_filenum (or 1 if start_filenum is not specified). With the default of 1, if multiple files to process are specified, the first is 001, the second 002, and so forth. • %s will insert the base name (exclusive of the last extension) of the input filename. So for instance if the input file name was "/images/my.data.tif" then the base "my.data" will be inserted into the output name where %s appears in the out_filename specification.
stderr	Standard error. This value is case insensitive.
stdout	Standard output. This value is case insensitive.

Table 3-1:Parameter Values (Continued)

Parameter Value Name	Meaning
<Y|N> Example values: Y no 0	The value "Y" is an abbreviation for "yes." Whenever it is specified either "Y" or "yes" is acceptable and will be handled the same. Similarly for "N" and "no." In addition, 1 or "y" will be accepted for Y, and 0 or "n" for N. **Parameters which take on the values Y and N (yes and no) are turned on when Y is specified and turned off when N is specified.**

Improving Accuracy

Typeset, high-quality printed pages return the best recognition accuracy. The following factors most affect text-recognition accuracy:

- Preprocessing Settings
- Recognition Parameters
- Line Art and Photographic Regions
- Document Quality
- Scanning Process

Preprocessing Settings and Recognition Parameters

No single combination of preprocessing settings and recognition parameters will always result in the quickest, most accurate recognition job. However, if you use the settings most appropriate to each document's page such as layout, printing, quality, font and size, OCR Shop XTR's speed and accuracy will be maximized.

Line Art and Photographic Regions

OCR Shop XTR may recognize some line-art graphics or areas of photographic regions as text if the artwork is poor and the lines resemble letter strokes or features look like in text in a light background. To avoid OCR Shop XTR incorrectly recognizing art of this type as text, create an rdiff file to designate the region containing the line art as an image region. Alternatively, adjusting the black_threshold parameter may change how the OCR Engine differentiates between photographic regions and text regions.

Document Quality

OCR Shop XTR recognizes characters in almost any font in sizes from 6 to 72 points. Following certain guidelines may improve recognition accuracy:

- The print should be as clean and crisp as possible. Characters should be distinct, separated from each other and not blotched together or overlapping.
- The document should be free of handwritten notes, lines and doodles. Anything that is not a printed character slows recognition, and any character distorted by a mark will be unrecognizable.
- Try to avoid highly stylized fonts. For example, OCR Shop XTR may not recognize text in the Zapf Chancery font accurately.
- Try to avoid underlined text. Underlining changes the shape of descenders on the letters q, g, y, p, and j.

Scanning Process

If you have control over the scanning process, you can improve recognition by taking a few steps during scanning to eliminate skew and background noise.

Make sure that the document is positioned correctly in your scanner and is not slanted. Even if you place a document in the scanner correctly, it can still shift enough as the lid is dropped to affect recognition. In such cases, the recognized text may contain missing characters, split lines of text, or unidentifiable words.

The sheet of glass on the flatbed of the scanner must be clean and clear. If it gets dirty, wipe it gently with a soft, damp, lint-free cloth or tissue. Be sure it is completely dry before you place anything on it.

Some paper is so thin that the scanner reads text printed on the back side of the scanned page. This is often the case with telephone book pages. To correct this problem, put a black piece of paper between the sheet and the lid of the scanner.

By eliminating any need for the OCR Engine to deskew an image, recognition processing speed will improve.

Spreadsheets and Tables

The following tips are useful when recognizing spreadsheets, charts, tables, single-column pages, or memos with page-wide text and tabs:

- Use the "-one_column=y" option to force a table to be recognized as one column, resulting in text output that preserves the table formatting. Otherwise, if "-auto_segment=y" and "-one_column=n" (the default behavior), the columns of the table might be recognized as separate regions, and written out sequentially in the text output. Which of these options you choose will depend on how you want the output formatted in the text file.
- With formatted output, such as PDF or formatted HTML, the best results will probably be obtained with "-auto_segment=y" and "format_analysis=y", which is the default behavior. In this case, the engine should recognize the columns of the table as separate regions and then will have the ability present the output in the original table format.
- For further customization of spreadsheet and table recognition, a rdiff file may be created to explicitly specify the region boundaries. See "rdiff Files" on page 36. In addition, if you know that all text contained in a table is numeric, then you could recognize only that region and restrict the character set to numbers only using the "-char_set=0123456789" and "-recognize_region=<region id>" options.

Foreign Language and Multilingual Documents

OCR Shop XTR has been configured with English as the default language, but the product is capable of recognizing over 50 different languages through the use of optional add-on language packs. If you have documents in different languages, consider using these add-on packs. Recognition of foreign language and

multi-language documents, then calls for setting the appropriate language(s) using the language parameter. In addition, you can specify your own lexicon of words in a user_lexicon file.

See “Supported Languages” on page 55 for more information on the languages supported by OCR Shop XTR and “-user_lexicon” on page 53 for details about user lexicons.

Chapter 4: Tutorials

Overview

Details of each OCR Shop XTR command line parameter are explained in more detail in the following section. In order to facilitate use of the OCR Shop XTR command line interface, this section presents a series of example commands that can be issued via the command line.

Please see the following chapters of this manual for more details about allowable parameter values and explanations of what each parameter does.

Each of the sample OCR Shop XTR command lines presented can be run as is (assuming the specified input files exist). Also, most examples demonstrate features that can be combined into more complicated commands.

The following tutorials guide the user through OCR Shop XTR's main functions and procedures of operation are presented:

- Tutorial 1 — Help and Version Information
- Tutorial 2 — Introduction to OCR Shop XTR
- Tutorial 3 — Controlling Pre-processing
- Tutorial 4 — Controlling Output
- Tutorial 5 — Language Settings
- Tutorial 6 — Resource Files and Settings
- Tutorial 7 — Log and Error Options

Tutorial 1 — Help and Version information

Before even getting started with the basics, you should know a few commands that can be used to get information about OCR Shop XTR and how to use it.

To get help:

```
ocrxtr -help
```

This command can be very handy when you quickly want to see the name of a given a parameter.

To get version information:

```
ocrxtr -version
```

See “Chapter 5: Help and Informational Parameters” on page 31 for more information.

Tutorial 2 — Introduction to OCR Shop XTR

This tutorial offers a brief introduction to OCR Shop XTR and optical character recognition.

Launching OCR Shop XTR

OCR Shop is started in the same way as other applications on the user's system. If the OCR Shop XTR executable directory ("$$VV_HOME/bin") is set in your PATH environment variable then you can invoke ocrxtr by simply typing

```
ocrxtr
```

Option parameters follow the program name then files to recognize are last on the command line.

What Is Optical Character Recognition (OCR)?

Optical Character Recognition (OCR) is the process of converting a text *image* file into a text file that a user can edit on screen. OCR is also referred to as *text* or *page recognition* software as it 'recognizes' imaged characters and turns them into type. OCR Shop XTR uses as its source document images stored in files.

OCR Shop XTR begins with an image that is really just a 'picture' of text and graphics and which cannot be edited directly by the user. The process of OCR transforms this 'picture' into separate characters of text and specific areas of graphics that can then be altered or edited individually by the user.

The recognized text from OCR Shop XTR can be exported to a variety of word-processing, page-layout and spreadsheet applications.

The OCR Process

OCR Shop XTR operates in a four-step process:

1. Acquire an image from a file
2. Perform pre-processing and segmentation of the image into zones
3. Recognize the image
4. Output the results

Running at the Command Line

The following command lines will output text to a file named “out.<input filename>.<doc num>”. The text will be in “iso” (simple ASCII text) format.

The simplest, fully automated way of running the OCR Shop XTR:

```
ocrxtr image.tif
```

The simplest run with all auto features turned off:

```
ocrxtr -auto_process=n -auto_filter=n image.tif
```

Fully automatic, overwrite any existing output file, and specify the output format and filename:

```
ocrxtr -overwrite=y -out_text_format=pdf \
 -out_text_name=out.pdf image.tif
```

Tutorial 3 — Controlling Pre-processing

By controlling the pre-processing activity of the OCR Engine, you can fine tune the OCR results. Many pre-processing activities can be controlled via the command line.

If you know your input image is a fax:

```
ocrxtr -fax_filter=Y image.tif
```

If you know your input image is output from a dot matrix printer:

```
ocrxtr -dotmatrix_filter=Y image.tif
```

If you know your input image is from a newspaper:

```
ocrxtr -newspaper_filter=Y image.tif
```

If you want to remove all graphics from output:

```
ocrxtr -remove_halftone=Y image.tif
```

See “Chapter 7: Pre-processing Options” on page 43 for details on all the pre-processing options.

Tutorial 4 — Controlling Output

OCR Shop XTR gives you many options for the type(s) of output from the OCR Engine. There are several text and raster output options available. In addition, with add-on features you can output compound documents in pdf and html formats.

To output a PDF file:

```
ocrxtr -out_text_format=pdf img.tif
```

Set the output filename to “out.pdf”, with PDF output:

```
ocrxtr -out_text_name=out.pdf -out_text_format=pdf img.tif
```

To output a basic html file:

```
ocrxtr -out_text_format=html image.tif
```

Use “wwhtml” or “thtml” for formatted html output.

See “Chapter 9: Output Functionality” on page 61 for details on all the output options.

Tutorial 5 — Language Settings

An important part of OCR is determining the language to be recognized in the text image being processed. Specific languages can be selected by using the OCR Shop XTR language parameter. Once set, the OCR Engine will then know what characters to expect and for several languages also have a lexicon of the most common words. For example, if you select German, OCR Shop knows that it will probably encounter such unique German language characters as 'ß' or double-ess.

For all languages, you can specifically also recognize characters from the English languages through use of the english_chars parameter. The OCR Engine supports many different languages and can output either ASCII or Unicode characters.

If the character set of the specified language does not fit into the standard ASCII range of characters you must specify an output text format that supports a wider range of characters.

To set the language to French and output in Unicode format:

```
ocrxtr -out_text_format=Unicode -language=french image.tif
```

When not explicitly specified, English is used as the default language.

Recognize a document with two languages which both use the Latin 1 code pages (see the section on Supported Languages in the Recognition Options chapter for more details).

```
ocrxtr -out_text_format=unicode -language=spanish,german \
 image.tif
```

Recognize a document with a non-Latin1 language and English characters:

```
ocrxtr -out_text_format=unicode -language=greek \
 -english_chars=Y image.tif
```

See "Supported Languages" on page 55 for a list of languages supported by OCR Shop XTR.

Tutorial 6 — Resource Files and Settings

If there are parameter settings that are often used in given situations they can be stored in a settings file. Also, the default parameters for operation of the OCR Shop XTR are stored in a resource file (.ocrxtr.rc) in the user's home directory.

To write the current settings to the user resource file (.ocrxtr.rc) in the user's home directory, use the write_resource_file option. The current settings written will combine the values of any existing .ocrxtr.rc file with the values of any specified parameter file with any values specified on the command line. For example, the line below will combine the current .ocrxtr.rc parameters with the values defined the parameter file parms.rc, with the command line parameter for the out_text_format and output all the parameters to a new .ocrxtr.rc file.

```
ocrxtr -write_resource_file=Y -read_params=parms.rc \
 -out_text_format=pdf img.tif
```

In future OCR sessions, the new .ocrxtr.rc will automatically be used.

To write the default user resource file (.ocrxtr.rc) to the user's home directory or reset that file to its default values:

```
ocrxtr -reset_resource_file=Y
```

Note: The above option may be specified with other parameters, but the default settings will not be used for that run of the OCR Engine, so it normally makes sense to run with this parameter alone on the command line as shown.

See “Chapter 10: Resource and Settings Files” on page 73 for more details.

Tutorial 7 — Log and Error Options

Write error messages to a file instead of standard error (STDERR):

```
ocrxtr -error_log=ERROR_MSGS.txt image.tif
```

Write informational messages to a file instead of standard output (STDOUT):

```
ocrxtr -info_log=INFO_MSGS.txt image.tif
```

Minimize diagnostic output:

```
ocrxtr -error_level=0 -info_level=0 image.tif
```

Maximize diagnostic output:

```
ocrxtr -error_level=5 -info_level=3 image.tif
```

See “Chapter 11: Debug and Log Options” on page 77 for more details.

Chapter 5: Help and Informational Parameters

Getting Help and Product Information

These are special, informational parameters, available from the command line only and take no values. If any values are specified they will be ignored.

Table 5-1:Help and Informational Parameters

Parameter	Value(s) & Default	Meaning
-help	none	Prints a summary of available command line options
-version	none	Prints the current version number of OCR Shop XTR.

If either of these special parameters is set, all other parameters are ignored and no pre-processing or recognition is performed. No file names should be specified when these parameters are invoked.

Chapter 6: Input Functionality

Overview

These parameters allow the list of input files to be specified in a file and also to set values specific to the input files. These parameters override whatever information may be contained in the file itself. For instance, setting in_res=100 will force the dots per inch resolution to 100 regardless of what is denoted in the input file. Parameters affecting the processing of input files apply to all input files specified.

Files in a number of different image formats (see “Supported Image Formats” below) are accepted by OCR Shop XTR. All input images are internally converted into 1-bit image data prior to the OCR Engine applying the pre-processing options and performing recognition.

Input Parameters

The following table lists the input parameters available in OCR Shop XTR.

Table 6-1:Input Parameters

Parameter	Value(s) & Default	Meaning
-black_threshold	<0-102> Default: 60	This is the threshold used to determine which pixels are black and which are white on a page when converting an image from multiple bits per pixel to the 1-bit per pixel image processed by the OCR Engine itself. Additional black_threshold options are provided for more sophisticated translations to a bi-tonal image. These include the value 101 which forces a random threshold to be used and the value 102 which directs the OCR Shop XTR to use the Floyd-Steinberg algorithm to determine which pixels are white and which are black. **Adjusting the black_threshold parameter can significantly affect the OCR Engine's recognition of image regions.**
-ignore_tiff_fillorder	<Y|N> Default: yes	Ignore the fillorder bit in input TIFF image files. Because the fillorder bit is frequently set incorrectly in TIFF image files, OCR Shop XTR ignores the fillorder bit by default. You only need to set this option in the rare instance that your input TIFF image has the fillorder bit set intentionally and correctly. If your input image looks readable when you view it and has a reasonable resolution, but OCR Shop XTR does an extremely poor job of recognizing the image, then the fillorder bit might be the reason. Try setting this option to "no" and running OCR Shop XTR again. See the environment variable VV_IGNORE_FILLORDER in "Configuring the Environment" on page 8. If the environment variable and this command-line option contradict each other, the TIFF fillorder bit will be obeyed.

Table 6-1:Input Parameters (Continued)

Parameter	Value(s) & Default	Meaning
-image_list	<filename> Default: none; no image_list is specified.	Specify a file containing a list of files to process. Each line of the image_list file should contain the name of a file to process. Blank lines and lines beginning with '#' are ignored. **Files specified in this parameter will be processed after any files specified on the command line itself.**

-image_rdiff_list	<filename>
	
	Default: none; no image_rdiff_list is specified.
		Input file containing a list of input images and rdiff files. See the discussion about rdiff files in section 3.1 below.
		
		Each line of the image_rdiff_list should contain the name of an image file to process followed by white space followed by the name of an rdiff file. Blank lines and lines beginning with '#' are ignored. The format is as follows:
		`#Files to process:`
		`<filename_1> <rdiff_filename1>`
		`<filename_2> <rdiff_filename2>`
		`<filename_3> <rdiff_filename3>`
		
		There is a space between the filename and the rdiff_filename. Both filename_n and rdiff_filename_n follow the same convention for <filename> as specified in the "Parameter Value Glossary" on page 14.

-in_res	<dpi> OR <dpi>x<dpi> default: 300x300 **unless** the document itself is in a format (e.g., tiff) which contains dpi resolution. If that is the case, the resolution specified in the document itself is used as the default.	Input image resolution. This overrides the resolution specified in the file itself. If only one value is specified then it is used for both x and y resolutions; if two values are specified (separated by an "x"), then the first value is used for the x resolution and the second value for the y resolution. **The minimum resolution recognized by the OCR Engine is 70 dpi and the maximum is 900 dpi.**

rdiff Files

An rdiff file is a “region description file”. As part of the recognition process, the document page layout is analyzed for regions of text and image. This information can be output at two different points in the process either before recognition via the out_prerec_rdiff parameter or, more commonly, after recognition via the -out_rdiff parameter.

Once created, an rdiff file can then be modified (with a text editor) to change which regions are to be recognized. Alternatively, a new rdiff file can be created independent of OCR Shop XTR and used. An rdiff file can then be passed to OCR Shop XTR via the image_rdiff_list parameter and used as the basis for the layout analysis performed by the OCR Engine.

The rdiff file read in for a particular input image will influence how that image is recognized. Modifying the coordinates of regions, removing regions, and/or adding regions will affect the results. This mechanism allows the user to have more control over exactly what parts of a document image are recognized.

Table 6-2:Sample rdiff File

Line in rdiff File	Comment
BEGIN_IMAGE	
FILENAME letter.tif	
IMAGE_WIDTH 2592	
IMAGE_HEIGHT 3412	
IMAGE_XRES 300	
IMAGE_YRES 300	
END_IMAGE	
BEGIN_SUMMARY	Summary of entire image
TOTAL_REGIONS 5	Total number of regions
TEXT_REGIONS 2	Number of text regions
IMAGE_REGIONS 1	Number of image regions
ORDER_FLAGS ANY	Required line
END_SUMMARY	

Table 6-2:Sample rdiff File (Continued)

Line in rdiff File	Comment
BEGIN_REGIONS	Below the regions are described individually:
R_TYPE IMAGE	Region 2 is an image region.
R_SUBTYPE LINEART	
R_NUMBER 2	R_NUMBER is the region id number.
R_OUT_ORDER 3	
R_UOR_LIST 1	This number indicates how many rectangles there are in the union of rectangles (UOR) list below. In this case, there is only 1.
368 599 280 783	The one rectangle in the UOR. The origin is in the upper left. Coordinates correspond to: top,bottom,left,right.
R_TYPE TEXT	Region 3 is the first text region.
R_SUBTYPE UNFLAVORED	
R_NUMBER 3	
R_OUT_ORDER 1	
LINE_HEIGHT 36	Text attribute information follows.
N_LINES 1	
STROKE_WIDTH 6	
ITALICNESS 5	
ITALICNESS_CONFIDENCE 99	
R_UOR_LIST 3	Text regions are also described by a UOR list.
848 1167 296 863	This region is defined by 3 rectangles.
1168 1327 296 2279	
1328 2279 296 2247	
R_TYPE TEXT	
R_SUBTYPE FOOTER	
R_NUMBER 11	
R_OUT_ORDER 2	
LINE_HEIGHT 32	
N_LINES 1	
STROKE_WIDTH 4	
ITALICNESS 2	
ITALICNESS_CONFIDENCE 99	
R_UOR_LIST 1	
2808 2855 304 1527	

Table 6-2:Sample rdiff File (Continued)

Line in rdiff File	Comment
R_TYPE IGNORE	The last two regions are neither text nor image; so they do not appear in the output. If they were not included in the rdiff file there would be no change to the output.
R_NUMBER 12	
R_UOR_LIST 1	
3360 3375 16 1391	
R_TYPE IGNORE	
R_NUMBER 13	
R_UOR_LIST 1	
3368 3375 1824 2575	

To use an rdiff file to manipulate how the engine will recognize an image, do the following:

1. Run OCR Shop XTR to create an rdiff file for the input image file.

   Execute the following command to create an rdiff file named "out.rdiff" based on the input image sample.tif:

   > ocrxtr -out_rdiff=sample.rdiff sample.tif

2. Modify the rdiff file.

   Modifying an rdiff file is most useful for:

   - Restricting which regions are included in the output file.
   - Modifying the region boundaries.

   To eliminate a region, remove that region's description and modify the region count.

   To modify a region's boundaries, edit the UOR information. Use an image viewer to view the input image and map its coordinates. Add or remove rectangle descriptions from the UOR list, and make sure to modify the rectangle count (the number after “R_UOR_LIST”.)

You can also add regions or modify other information in the rdiff file.

3. Create a image list file that associates the modified rdiff information with the input file.

   Create a text file named "list.txt" that contains this single line:

   ```
   sample.tif sample.rdiff
   ```

4. Run OCR Shop XTR a second time with the same input file and the text file created above.

   Enter the following command line:

   ```
   ocrxtr -image_rdiff_list=list.txt
   ```

   The list.txt file points to sample.tif, therefore you should not list sample.tif on the input line.

A good way to intuitively understand how using an rdiff file will affect output is to create PDF output. Create and modify an rdiff file for an image, then compare the results from these runs:

```
ocrxtr -image_rdiff_list=list.txt -out_text_format=pdf \
-auto_segment=Y
ocrxtr -image_rdiff_list=list.txt -out_text_format=pdf \
-auto_segment=N
```

Auto segmentation affects how the engine uses the rdiff input file.

Supported Image Formats

OCR Shop XTR supports input files in a number of different image formats. OCR Shop XTR reads an input file and automatically determines the input file format.

Some formats are supported by add-on options that may be purchased.

Supported formats are listed below:

Table 6-3:Supported Image Formats

Format name	Typical file extension	Comments
Graphics Interchange Format (GIF)	.gif	
Joint Photographic Experts Group File Interchange Format (JPEG)	.jpg	JPEG formats compatible with libjpeg, in particular JPEG File Interchange Format (JFIF). See http://www.ijg.org/files/
Portable BitMap (PBM)	.pbm	Bi-level (1-bit per pixel) bitmap with a simple header
Portable document format (pdf)	.pdf	By default, OCR Shop XTR renders a PDF input file into 24-bit raw image data, which can be quite large for multipage PDF input files. The user may improve processing speed and reduce memory usage by restricting the bit depth at which the PDF file is rendered. See "-out_depth" on page 71 for instructions.
Portable network Graphics Format (PNG)	.png	
Portable PixMap (PPM)	.ppm	RGB-encoded bitmap with a simple header
Postscript	.ps & .eps	Levels 1, 2, and 3
Rasterfile	.ras	Native bitmap of Sun; see /usr/include/rasterfile.h or man rasterfile(5) on a Sun system.
Silicon Graphics image file format (SGI-RGB)	.rgb, .sgi, and .iris	As per Silicon Graphics' library libimage.a. Type "man 4 rgb" on an IRIX machine for more details.

Table 6-3:Supported Image Formats (Continued)

Format name	Typical file extension	Comments
Tagged image file format (TIFF)	.tif	Both single page and multipage formats are supported. Images within a TIFF file may be uncompressed or compressed in Group 3, Group 4, LZW, or JPEG compression.
X Window Dump (XWD)	.xwd	Screen dump from an X Window System. See “man xwd” on most Unix and Linux systems for details.
X11	.xpm, .xbm, and .bm	X Consortium created format. It is a relatively simple bitmap format. The extension of .xpm indicates a pixel map (see “Portable PixMap (PPM)”. above) and .xbm indicates a bi-level bitmap (see “Portable BitMap (PBM)” above).

Chapter 7: Pre-processing Options

Overview

The OCR Engine can be fine-tuned to meet your specific needs. A large number of pre-processing options are supported. These are divided into basic and advanced options. The basic pre-processing options are the more commonly used options; the advanced pre-processing options are likely to be used only in very specific cases. Defaults have been established (in particular auto_process=full) so that in most situations good recognition results can be achieved even if no pre-processing options are explicitly set.

Basic Pre-processing Options

The following table lists the basic pre-processing parameters available in OCR Shop XTR. For a list of advanced parameters that are available see the following section “Advanced Pre-processing Options”.

Table 7-1:Basic Pre-processing Options

Parameter	Value(s) & Default	Meaning
-auto_process	<full|preprocess_only|N> full: turns on auto-preprocessing and automated layout analysis preprocess_only: turns on auto-preprocessing. Default: full	Automate much of the pre-processing activities. Full will fully automate the processing; preprocess_only will partially automate the processing; N will turn off auto-pre-processing. **If the auto_process parameter is set to full or preprocess_only, then a number of individual pre-processing flags are set to auto (as documented below). Individually setting these parameters will then override the auto_process value. For example if auto_process=full and fax_filter=N then fax filtering will be turned off.**
-auto_filter	<Y|N> Default: Set to Y if auto_process=full or auto_process=preprocess_only; else N	This parameter controls the pre-processing options related to filtering separate from other pre-processing options. When this is set to Y, auto filtering is turned on, which includes preprocess dot detection and auto fax filter. If auto_segment=Y, auto_filter=Y will also set segment_lineart=Y and reverse_video=Y. **auto-filter is applied after the auto-process parameter and before any individual settings. Thus auto_filter overrides auto_process and then, in turn, individually set filter parameters will override the auto_filter setting.**

Table 7-1:Basic Pre-processing Options (Continued)

Parameter	Value(s) & Default	Meaning
-rotate	<0|90|180|270> These are degrees of rotation. Default: 0	Explicitly rotate the input image during pre-processing. Non-orthogonal rotation by an arbitrary angle is not supported by this parameter. **If auto_orient is set to Y and rotate is set to a non-zero value, the image will be rotated so that it is upright before recognition, even if this differs from the rotate value specified. The only effect of specifying both parameters is that the OCR Engine will favor the suggested rotation of the rotate parameter in determining the orientation of the page.**
-auto_orient	<detect|correct|N> Default: Set to correct if auto_process=full or auto_process=preprocess_only; else N	When this is set to correct then the image is automatically rotated to correct its orientation. See the note for the rotate parameter above. When set to detect, “Landscape” or “portrait” is written to the info log; no correction is performed.
-fax_filter	<Y|Nauto> Default: Set to auto if auto_process=full or auto_process=preprocess_only; this will be overridden by the auto_filter parameter setting fax_filter=auto when auto_filter=Y.	When set to Y, the image is considered to be a fax and special fax filtering is performed. When set to auto, fax filtering is done if necessary.
-newspaper_filter	<Y|N> Default: N	When set to Y, the image is considered to be a newspaper page and special newspaper filtering is performed. **This filter is not turned on by the auto_process or auto_filter parameters.**

Table 7-1:Basic Pre-processing Options (Continued)

Parameter	Value(s) & Default	Meaning
-dotmatrix_filter	<Y|N|auto> Default: Set to auto if auto_process=full or auto_process= preprocess_only; this will be overridden by the auto_filter parameter setting dotmatrix_filter=auto when auto_filter=Y.	When set to Y, the image is considered to be a page from a dot matrix printer and special dot matrix filtering is performed. When set to auto, dot matrix filtering is done if necessary.
-deskew	<correct|N|manual> Default: set to correct if auto_process=full or auto_process= preprocess_only; otherwise N	The deskew parameter, when set to manual, works in conjunction with three other parameters: • deskew_confidence • deskew_upper_angular_thresh • deskew_lower_angular_thresh See descriptions of these below.
-deskew_confidence	<0-100> This is a percentage from 0 to 100. Default: 10	This is a confidence level that must be exceeded in order for deskew to be performed. This parameter applies only if deskew=manual.
-deskew_upper_ angular_thresh	<float> This is a number of degrees. Default: 10	This is the maximum number of degrees that a document will be deskewed. This parameter applies only if deskew=manual.
-deskew_lower_ angular_thresh	<float> This is a number of degrees. Default: 0.25	This is the minimum number of degrees that a document must be skewed before any deskewing is performed. This parameter applies only if deskew=manual.

Table 7-1:Basic Pre-processing Options (Continued)

Parameter	Value(s) & Default	Meaning
-invert	<Y|N> Default: N	When set, the 1-bit per pixel image is inverted, so that white becomes black and black becomes white for use of the image in the recognition process. **This parameter affects the recognition process; it does not affect output of individual graphics elements (via the out_graphics_name parameter).**
-analyze_layout	<Y|N> Default: set to Y if auto_process=full; otherwise N	This is an important step in determining where text and images are on a page. When this is set to N, the entire page is assumed to be text.

Advanced Pre-processing Options

Several of these parameters are considered advanced options, because they provide a level of control that is unlikely to be utilized except in very specific circumstances.

.

Table 7-2:Advanced Pre-processing Options

Parameter	Value(s) & Default	Meaning
-double_dimension	<Y|N> Default: N	This parameter is used for non-square images (e.g., faxes which were transmitted at 200x100 dots per inch). When set, the dimensions of each image will be examined and pixels will be doubled in the dimension with the lower resolution. **If the image is already square (i.e., x-dpi equals y-dpi, no doubling is performed.**
-auto_segment	<Y|N> Default: Set to Y if auto_process=full; otherwise N	When set, the OCR Engine performs the analysis to divide the different image and text areas of the document.
-segment_lineart	<Y|N> Default: Y if auto_segment=Y and auto_filter=Y or if auto_process=full; otherwise N	When set, this causes the OCR Engine to distinguish between line art image regions and halftone image regions. **This will only have an effect if auto_segment is set to Y or the advanced output option remove_halftone is set to Y.**
-single_col_autoseg	<Y|N> Default: Y	When this is set the OCR Engine performs additional processing after it determines that the page is a single column.
-one_column	<Y|N> Default: N	When set, forces a page to be interpreted as containing only one column.
-two_page_mode	<Y|N> Default: N	When set, considers documents to be two facing pages as in the result of a flat scan of an open book.

Table 7-2:Advanced Pre-processing Options (Continued)

Parameter	Value(s) & Default	Meaning
-photometric_interp	<detect|correct|N> Default: N	When set, photometric interpretation is performed, i.e., the OCR Engine determines if the input document is overall reverse or normal video. If the document is mostly reverse video, then the OCR Engine will invert the entire image.
-reverse_video	<Y|N> Default: Y if auto_segment=Y and auto_filter=Y otherwise N.	When set to Y, this parameter detects which regions of the page are reverse video so that the OCR Engine will know to invert the image before recognition. This option will affect the output on pages where reverse-video text exists and is detected. **Text will not be recognized in reverse video regions unless this option is set.**

Chapter 8: Recognition Options

Overview

This set of options allows for control of how recognition itself is performed by the OCR Engine.

Recognition Options

The following table lists the recognition parameters available in OCR Shop XTR.

.

Table 8-1:Recognition Options

Parameter	Value(s) & Default	Meaning
-language	<language> or <language1 [,language2] … [,languageN]> Possible Values: See the following section below. Default: english	Loads the language pack for the specified language. Multiple languages may be specified (separated by commas). **If multiple languages are specified, then they must all use the same set of shapes (glyphs). See the section "Supported Languages" on page 55 for details.** **Some languages produce output which is incompatible with some output formats. For example Russian cannot be represented by ASCII text.** **Language names are case-insensitive.**
-english_chars	<Y|N> Default: N	If set, the OCR engine will recognize, in addition to the languages specified in the language parameter, the letter shapes associated with the English language. This option should be used when documents in a non-English language contain embedded English words.

Table 8-1:Recognition Options (Continued)

Parameter	Value(s) & Default	Meaning
-char_set	<string> equal to the set of characters to be recognized; e.g. char_set=0123456789 will cause only numbers to recognized. Default: not set, not constrained.	If set, the OCR engine will only recognize the characters specified in the char_set string. If the document being recognized only contains a few characters specified by this parameter, the resulting text file will contain characters from the char_set along with the reject_char character for any characters not in the char_set. **Setting to the null string, i.e., char_set= will cause the OCR Engine to not be constrained.** **char_set is case-sensitive. For instance, to recognize both uppercase and lowercase d, both must be specified.** **Because "s" and "5" have similar character shapes, if "s" (or "S") is specified then the number 5 will also be recognized in the output, even if it is not specified.**
-user_lexicon	<filename> Default: none	If set, the OCR Engine will use the words listed in the file in addition to any defined for the languages specified by the language parameter. If there are any proper names, technical terms, acronyms, and so on, that are common to the documents being recognized, then defining a user_lexicon that includes these words will improve the recognition results. The file should consist of lines as follows: **word<whitespace>lexclass<newline>** where <whitespace> is one or more space and/or tab; and lexclass is a number from 1 to 26 which simply categorizes the word into a group of similar words. It would be sufficient to set lexclass to 1 for all words.
-min_point	<5-72> This value is expressed as a point-size. Default: 5	Sets the minimum point size of text that the OCR Engine will recognize.

Table 8-1:Recognition Options (Continued)

Parameter	Value(s) & Default	Meaning
-max_point	<5-72> This value is expressed as a point-size. Default: 72	Sets the maximum point size of text that the OCR Engine will recognize.
-format_analysis	<Y|N> Default: Y	This performs analysis of the format that will affect pdf and formatted html output.
-recognize_region	<region id> Default: none specified, i.e., the entire document will be recognized.	Only the specified text region will be recognized. The text region must already be defined. Text regions are numbered in an rdiff file.
-timeout	<int> Default: no timeout	Set a timeout in seconds for recognition.

Supported Languages

The OCR Engine supports over 50 different languages. Languages are divided into groups by the shapes or glyphs of characters contained within the language. The OCR Engine follows the glyph conventions of Microsoft Code Pages. See http://msdn.microsoft.com/en-us/goglobal/bb964653 for more details.

Several languages have ".lng" file associated with them. These languages contain word information in addition to information about the allowable shapes (glyphs) in the language. The .lng files are stored in the /lib/lang sub-directory of the OCR Shop XTR installation directory. For other languages only information about permissible shapes is used.

> If multiple languages are specified in the language parameter then they all must use the same set of shapes (same Code Page). To recognize languages that use different sets of shapes on the same page, OCR Shop XTR must be run multiple times on a region-by-region basis. In these cases, each time OCR Shop XTR should be run with a different set of languages specified and only the relevant regions specified in an rdiff file.

Languages supported by the OCR Engine are listed in the table below. Your licensed copy of OCR Shop XTR may not contain support for all these languages.

Table 8-2:Supported Languages

Language	<language name>	".lng" file available	Shape Pack/Code pages
Afrikaans	afrikaans	no	Latin I (1252)
Albanian	albanian	no	Central Europe (1250)
Aymara	aymara	no	Latin I (1252)
Basque	basque	no	Latin I (1252)
Breton	breton	no	Latin I (1252)
Bulgarian	bulgarian	no	Cyrillic (1251)

Table 8-2:Supported Languages

Language	<language name>	".lng" file available	Shape Pack/Code pages
Byelorussian	byelorussian	no	Cyrillic (1251)
Catalan	catalan	no	Latin I (1252)
Croatian	croatian	no	Central Europe (1250)
Czech	czech	yes	Central Europe (1250)
Danish	danish	yes	Latin I (1252)
Dutch	dutch	yes	Latin I (1252)
English	english	yes	Latin I (1252)
Estonian	estonian	no	Baltic (1257)
Faroese	faroese	no	Latin I (1252)
Finnish	finnish	yes	Latin I (1252)
Flemish	flemish	no	Latin I (1252)
French	french	yes	Latin I (1252)
Frisian - West	frisianw	no	Latin I (1252)
Friulian	friulian	no	Latin I (1252)
Gaelic	gaelic	no	Latin I (1252)
Galician	galician	no	Latin I (1252)
German	german	yes	Latin I (1252)
Greek	greek	yes	Greek (1253)
Greenlandic	greenlandic	no	Latin I (1252)
Hawaiian	hawaiian	no	Baltic (1257)
Hungarian	hungar	yes	Central Europe (1250)
Icelandic	icelandic	no	Latin I (1252)
Indonesian	indonesian	no	Latin I (1252)
Italian	italian	yes	Latin I (1252)

Table 8-2:Supported Languages

Language	<language name>	".lng" file available	Shape Pack/Code pages
Kurdish (Latin)	kurdishlat	no	Turkish (1254)
Latin	latin	no	Latin I (1252)
Latvian	latvian	no	Baltic (1257)
Lithuanian	lithuanian	no	Baltic (1257)
Macedonian (Cyrillic)	macedonianc	no	Cyrillic (1251)
Malaysian	malaysian	no	Latin I (1252)
Norwegian	norsk	yes	Latin I (1252)
Pigin English	piginenglish	no	Latin I (1252)
Polish	polish	yes	Central Europe (1250)
Portugese	port	yes	Latin I (1252)
Romanian	romanian	no	Central Europe (1250)
Russian	russian	yes	Cyrillic (1251)
Serbian	serbian	no	Cyrillic (1251)
Serbo-Croatian	sbcroatian	no	Central Europe (1250)
Slovak	slovak	no	Central Europe (1250)
Slovenian	slovenian	no	Central Europe (1250)
Sorbian - Lower	sorbianl	no	Central Europe (1250)
Sorbian - Upper	sorbianu	no	Central Europe (1250)
Spanish	spanish	yes	Latin I (1252)
Swahili	swahili	no	Latin I (1252)
Swedish	swedish	yes	Latin I (1252)
Tahitian	tahitian	no	Latin I (1252)
Turkish	turkish	yes	Turkish (1254)

Table 8-2:Supported Languages

Language	<language name>	".lng" file available	Shape Pack/Code pages
Ukranian	ukranian	no	Cyrillic (1251)
Welsh	welsh	no	Latin I (1252)
Zulu	zulu	no	Latin I (1252)

The languages listed by language pack are presented in the following table.

Table 8-3:Languages by Language Pack

Shape Pack/ Code pages	Languages		
Baltic (1257)	Estonian	Latvian	
	Hawaiian	Lithuanian	
Central Europe (1250)	Albanian	Polish	
	Croatian	Romanian	Slovenian
	Czech	Serbo-Croatian	Sorbian - Lower
	Hungarian	Slovak	Sorbian - Upper
Cyrillic (1251)	Bulgarian	Macedonian (Cyrillic)	Serbian
	Byelorussian	Russian	Ukranian
Greek (1253)	Greek		

Table 8-3:Languages by Language Pack

Shape Pack/ Code pages	Languages		
Latin I (1252)	Afrikaans	French	Malaysian
	Aymara	Frisian - West	Norwegian
	Basque	Friulian	Pigin English
	Breton	Gaelic	Portugese
	Catalan	Galician	Spanish
	Danish	German	Swahili
	Dutch	Greenlandic	Swedish
	English	Icelandic	Tahitian
	Faroese	Indonesian	Welsh
	Finnish	Italian	Zulu
	Flemish	Latin	
Turkish (1254)	Kurdish (Latin)	Turkish	

Chapter 9: Output Functionality

Overview

Following recognition, the OCR Engine can output the results in a number of different ways. Through the output options, you can control what output is made.

Basic Output Options

The following table lists the basic output parameters available in OCR Shop XTR. See the following section on page 68 for advanced output parameters that are available.

.

Table 9-1:Basic Output Options

Parameter	Value(s) & Default	Meaning
-out_text_name	<out_filename> or info_log or none Default: out.<filename>.<file num> where <filename> is the name of the file being processed (without its extension) and <file num> is its file number. <file num> will be included even if there is only one input file specified.	This is the template of the output filename for the text recognized. See the description of out_filename in the Parameter Glossary for more details. If %d is not included in the out_filename, and there are multiple input files, then the file number is automatically appended to the filename unless the parameter combine_docs=Y is set. This is done to avoid overwriting files created during a given execution of OCR Shop XTR. If there is only one input file then %d is not added unless it has been explicitly specified in the out_filename. Note that %d refers specifically to the input file number, so even if an input file contains multiple pages, e.g., a multipage tiff input file, OCR Shop XTR will only produce a single output text file. If out_text_name= info_log is set then output is redirected to the information log (see the info_log parameter). **If out_text_name= none, no text output will be generated.** Also see the comments under the out_text_format parameter.

Table 9-1:Basic Output Options (Continued)

Parameter	Value(s) & Default	Meaning
-start_filenum	<0-999> Default: 1	Starting file number to use in conjunction with the out_text_name parameter. **The file number loops, i.e., if a file number is set to 999, then the next file number will be 000.** **If a negative number is specified, then the default of 1 is used.**
-combine_docs	<Y|N> Default: N	This option applies to the out_text_name and out_text_format parameters. If combine_docs=Y, then only one output text (or compound document) file will be created, with one page per input file. If combine_docs=N, then multiple input files are treated as separate documents, with one output file created per input file.

Table 9-1:Basic Output Options (Continued)

Parameter	Value(s) & Default	Meaning

-out_text_format	<format>
	
	Possible Values:
	none
	iso
	8bit
	unicode
	html
	wwhtml (= WYSIWYG html,
	with use of styles)
	thtml (= html with frames)
	pdf
	
	Default: iso
		This is the output format for either text or, when the format supports it, both text and graphics.
		
		The format and bit depth of PDF output may be controlled with two additional command-line options. Please see “-out_depth” on page 71 and “-pdf_format” on page 68.
		
		If out_text_format=html (or wwhtml or thtml), the out_text_name will be used to construct the names of associated files created for each input document. If the out_text_name ends in ".htm" or “.html”, then the stem of the html output filenames will not include the ".htm" or “.html” extension. The html output files will be named based on the stem plus the appropriate extension, ".htm", ".css" or ".jpg" depending on the output file type. For html format output, ancillary files for each output document include a stylesheet and, for each image region in the input, a JPEG image.
		
		For html output, if combine_docs=Y, the output is modified in the following way:
		• Only one stylesheet is created for all the output html files.
		• An index html file named <out_file_name>_ndx.htm is created. The file contains links to each html file.
		• Each html file will contain a header and footer allowing navigation between pages of the output.
		• Four GIF images (but-blank.gif, index.gif, but-next.gif, and but-prev.gif) are copied into the output directory for use in the headers and footers. Note these images are always written if system permissions allow it.
		• A ‘%s’ included in the out_text_name will refer to the first input file only.
		• The html and JPEG files are all named based on the same stem. All but the first have an underscore followed by a number (the output page number minus 1) appended to the stem.
		
		For example, this command line:
		`ocrxtr -out_text_name=out_%s.htm \`
		`-out_text_format=html -combine_docs=N \`
		`test.tif sample.pdf`
		will produce these output files for the first input file, assuming

Table 9-1:Basic Output Options (Continued)

Parameter	Value(s) & Default	Meaning
-out_text_format continued		the second and fourth regions on the page are image regions: out_test001.htm, out_text1.css, out_test001.002.jpg, and out_text001.004.jpg. And for the second input file, these output files (assuming its first two regions are images: out_sample002.htm, out_sample002.css, out_sample002.001.jpg, and out_sample002.002.jpg.

-out_graphics_name (continued)		For html format, a similar example with combine_docs=Y,
		`ocrxtr -out_text_name=out_%s.htm \`
		`-out_text_format=html -combine_docs=Y \`
		`test.tif sample.pdf`
		will produce these output files:
		• For the index page: out_test_ndx.htm and the four gif files but-blank.gif, index.gif, but-next.gif, and but-prev.gif.
		• A Stylesheet for all pages: out_test.css,
		• For the first input file: out_test.htm, out_test.002.jpg, and out_test.004.jpg
		• For the second input file: out_test_1.htm, out_test_1.001.jpg and out_test_1.002.jpg

Table 9-1:Basic Output Options (Continued)

Parameter	Value(s) & Default	Meaning
-out_graphics_name	<out_filename> Default: none; no graphics output to separate files is produced unless this parameter or the out_graphics_format is specified. If out_graphics_format is specified and this parameter is not, then the value "outgraphics%d" is the default.	This is used (in conjunction with out_graphics_format) to output each image region of the document(s) to separate graphics files. This parameter is the template of the output filename for graphics and is specified similar to the out_text_name parameter. **If %d is not included in the out_graphics_name, and there are multiple input files, then the file number is automatically appended to the filename. This is done to avoid overwriting files created during a run of OCR Shop XTR.** **The region_id will be automatically appended to the output file name of each graphic. The region_id is represented by a three-digit number with leading zeroes. The first region id is 001.** **The output file type extension is also automatically added to the end of each of the output graphics filenames. For instance, if out_graphics_name= graph%d and the output format is tiff, then for the first region of the second image the graphics filename will be "graph002.001.tif".** **In order to produce this graphics output, the parameter auto_segment must be set to Y (which will be the case if auto_process=full).**

Table 9-1:Basic Output Options (Continued)

Parameter	Value(s) & Default	Meaning
-out_graphics_format	<format> Possible Values: (file extensions automatically used for each are in parenthesis) tiff (tif) ras (ras) epsf (eps) x11 (x11) tiff-pack (tif) tiff-g31d (tif) tiff-g32d (tif) tiff-g42d (tif) tiff-lzw (tif) pal-tiff (tif) gif (gif) jpeg (jpg) png (png) xwd (xwd) rgb (rgb) rgb-rle (rgb) pdf (pdf) – image-only pdf Default: tiff (used only if out_graphics_name is specified)	This specifies the format for graphics data when graphics are output to separate files. The bit-depth of an output graphics file may be set by the user; see "-out_depth" on page 71. The bit depth of the output file is further restricted by the chosen output graphics format. For example, the JPEG format does not support 1-bit image data, and the tiff-g31d format does not support 24-bit image data. If the chosen output bit depth is not in accordance with the bit depth of the output graphics format, the bit depth used will be the minimum required to create a valid image file in the specified format. **If pdf is specified here, then the graphics are output to separate files which are each in an embedded-image pdf format.**
-overwrite	<Y|N > Default: N	Indicate whether to overwrite existing files.

Advanced Output Options

Several of these parameters are considered advanced options, because they provide a level of control that is unlikely to be utilized except in very specific circumstances.

Table 9-2:Advanced Output Options

Parameter	Value(s) & Default	Meaning
-pdf_format	<img_text|normal| img_only> where img_text: full-page image with invisible text behind it normal: text, pictures, and embedded images img_only: only the original image Default: img_text; only applies if out_text_format=pdf.	If the out_text_format is specified as PDF, then this option specifies which type of PDF output to create. For more control over PDF output, the user may also set the bit-depth of the output PDF image data. Using a 1 or 8-bit depth over a 24-bit depth has the advantage of improving processing time and reducing memory usage and output filesize. Please see "-out_depth" on page 71 for more information.
-out_image_scale	<1-500> This is a percentage of the original. Default: 100, i.e., no scaling is done.	Scales the output image by the indicated percentage. **This option applies only to separate graphics output; it does not apply to graphics embedded in pdf output files.**

Table 9-2:Advanced Output Options (Continued)

Parameter	Value(s) & Default	Meaning
-reject_char	<char> default: '~', i.e., the tilde character (ASCII 126)	Character used to represent characters that are rejected, i.e., do not meet the recognition criteria of any of the characters in the language set. **If a string of more than one character is specified, only the first character of the string is used as the reject character.**
-out_rdiff	<out_filename> Default: none	This specifies an rdiff file, i.e., an output file containing region definition information. Format of the out_rdiff parameter value matches that of the out_text_name parameter value. See "rdiff Files" on page 36.for more information.
-out_prerec_rdiff	<out_filename> Default: none	This specifies an rdiff file, i.e., a file to which to output region definition information. Format of the out_prerec_rdiff parameter value matches that of the out_text_name parameter value. See "rdiff Files" on page 36 for more information on rdiff files. **The output of this parameter differs from that of the out_rdiff parameter in that the regions are determined prior to recognition, i.e., prior to any modifications that might be made in the regions during the recognition step. Generally, out_rdiff should be used rather than out_prerec_rdiff.**

Table 9-2:Advanced Output Options (Continued)

Parameter	Value(s) & Default	Meaning

-out_regions_as_graphics	<Y|N>
	
	Default: N
		Outputs each text region as an image in the format specified by the out_graphics_format parameter. The output files follow the naming pattern of the out_text_name with the additional extension:
		.%r.<x-value>.<y-value>.<fmt_ext>
		where %r is the region_id as an integer; x-value and y-value are the pixel location of the upper left corner of the bounding rectangle of the text region; and fmt_ext is the format extension of the specified out_graphics_format.
		
		For example:
		`outxtr -out_text_name=out.%d.txt \`
		`-out_graphics_format=jpeg`
		`-regions_as_graphics=Y \`
		`sample.tif`
		
		creates these files:
		out.001.txt
		outgraphics.2.280.368.jpg
		outgraphics.3.296.848.jpg
		outgraphics.11.304.2808.jpg
		
		The upper left corner of the page is (0,0).

-output_text_by_region	<Y|N> Default: N	Creates an output text file for each separate text region instead of one text file for an entire document (or all documents when combine_docs=Y). The output files follow the naming pattern of the out_text_name with the additional extension ".region<region_id>" where region_id is an integer. For compound document output, e.g., out_text_format=pdf or out_text_format=html all embedded images on a page will be contained in the output file for every region on that page.
-remove_halftone	<Y|N> Default: N	When set to Y, the OCR Engine removes all image regions from output including halftone and line art regions.

Table 9-2:Advanced Output Options (Continued)

Parameter	Value(s) & Default	Meaning
-photometric_interp	<Y|N> Default: N	When set, inverts photometric interpretation regions, i.e., on a region-by-region basis, the OCR Engine determines whether the region is reverse video and flips it back to normal video if that is the case.
-out_depth	<input|1|8|24> Default: input "input" instructs OCR Shop XTR to use the depth of the input image for PDF, HTML, or graphics output; for PDF or PS input, 1-bit is used as the default since the bit depth is unknown.	Set the bit depth of output image data in PDF output, HTML output, and graphics output. By setting the output bit depth to a lower value than the input bit depth, the user may reduce output filesize. When a multipage PDF or PS file is passed as input, by default OCR Shop XTR renders the input files to 1-bit per pixel in order to optimize processing time and reduce memory usage. When 8 or 24-bit PDF, HTML, or graphics output is desired, and PDF or PS input is used, out_depth should be set to 8 or 24. Because OCR Shop XTR uses the out_depth to determine how to render input PDF and PS files, setting out_depth to 8 or 24 will increase memory usage and processing time. For all input file types, if out_depth is set to a lower value than the bit depth of the input image, by definition graphical information will be lost in PDF, HTML, or graphics output. Setting out_depth only affects PDF, HTML, and graphics output. Only for PDF and PS input does setting out_depth have an effect on how the input is rendered. Setting out_depth does not affect the quality of the OCR results.
-xdoc_word_confidence	<Y|N> Default: N	Include word confidence values in XDOC output.
-xdoc_char_confidence	<Y|N> Default: N	Include character confidence values in XDOC output.
-xdoc_word_coords	<Y|N> Default: N	Include word coordinates in XDOC output.

Table 9-2:Advanced Output Options (Continued)

Parameter	Value(s) & Default	Meaning
-xdoc_char_coords	<Y|N> Default: N	Include character coordinates in XDOC output.

Chapter 10: Resource and Settings Files

Overview

A resource file is stored in a user's home directory and named ".ocrxtr.rc". This file is used automatically to load parameters. An additional parameter file with the same format as a resource file, may be specified on the command line.

Parameters for OCR Shop XTR are read first from the user's .ocrxtr.rc file, then from the parameter file (if specified), then from the command line. See Figure 3-1, "Parameter Execution Order," on page 13.

Only one parameter file can be specified and it can only be specified on the command line. The last one specified on the command line will be used.

When the resource file or parameters are written to a file, all other processing e.g., pre-processing and recognition will still take place as specified by the parameters.

Resource and Settings File Parameters

The following table lists the parameters available in OCR Shop XTR to set and read resource and setting files.

Table 10-1:Resource and Settings File Parameters

Parameter	Value(s) & Default	Meaning
-read_params	<filename> Default: none	Read parameters from the specified file. **The filename must end in the extension ".rc."**
-write_params	<filename> Default: none	Writes all current parameters to the specified file. Only explicitly set values are written, i.e., no default values are written with the following exceptions: • Parameters error_level and info_level • Both out_text_name and out_text_format if one but not both parameters are specified. • Both out_graphics_name and out_graphics_format if one but not both parameters are specified. **No Resource and Settings Files parameters are written to the file.** **The extension ".rc" is automatically appended to the filename (unless it already ends in ".rc").**
-reset_resource_file	<Y|N> Default: N	Resets the resource file to the initial default settings (as specified in this document). **The changes will take effect on the next OCR Shop XTR run.**

Table 10-1:Resource and Settings File Parameters (Continued)

Parameter	Value(s) & Default	Meaning
-write_resource_file	<Y|N> Default: N	Writes the current settings (a combination of the user resource file, any parameter files specified, and the command line parameters) to the user's resource file ".ocrxtr.rc" in the user's home directory. **No Resource and Settings Files parameters are written to the file.**

Chapter 11: Debug and Log Options

Overview

You can control the level of informational and debug data output during the recognition process.

Debug and Log Parameters

The following table lists the parameters available in OCR Shop XTR for logging and debugging.

Table 11-1:Debug and Log Parameters

Parameter	Value(s) & Default	Meaning
-info_log	<filename> or stdout Default: stdout	This controls where diagnostic, status, and debugging information is written and facilitates redirecting this status information to a file.
-error_log	<filename> or stderr Default: stderr	Redirects standard error to a file. Normally, error information is directed to standard error.Writes all current parameters to the specified file.
-error_level	<0-5> Default: 5	Level to filter error messages. The most detailed is 5. **This level refers to what data is written to the error_log not whether or not OCR Shop XTR immediately exits on a fatal error condition.**
-info_level	<0-3> Default: 3	Level to filter informational messages. 0 is the least amount of detail; The most detailed is 3; it includes debug information.
-debug	<Y|N> Default: N	Generate detailed debug output to stderr and stdout, appropriate for sending to Vividata when reporting a bug.

Appendix A: Troubleshooting

Overview

This chapter offers some troubleshooting hints as well as brief pointers to maximize operation efficiency.

Getting Help

Read this section of the manual

This section of the manual contains useful information on common problems and troubleshooting. If you do not find an answer please go to our website's support section.

Submitting a Question to the Support Department

At the Vividata Website http://www.vividata.com you will see a link "*Contact Support*". Go to this page to fill out information and submit a support request.

Identifying the Problem

There are several status files and options you can set in OCR Shop XTR to help you identify the cause of your problem.

Step 1: Verify Licensing is Working

Run vvlmstatus to check if the license manager daemon is running:

```
$VV_HOME/bin/vvlmstatus
```

This command will display a list of the license keys you have installed and number of licenses available for each key. It also displays the license manager process id and process name, if it is running.

If you have not run Vividata software before, the license manager will not be running. This is normal. The license manager is started automatically the first time you run OCR Shop XTR.

Try running OCR Shop XTR to start the license manager, then run vvlmstatus again. If the license manager is still not running or shows errors, verify that a license key is installed on your system. The file `$VV_HOME/config/license.dat` contains your license key(s). It should be an ASCII text file with 644 permissions. If it is not there, then your license key has not been installed. Install the license key via the distributed shell script, `key.sh`, or contact Vividata if you can not find your key or have questions.

If something still seems wrong with the license manager, set an environment variable called `VV_DEBUG` to 1000 and then run OCR Shop XTR. A large amount of debug information will print to the console, including any error messages regarding licensing.

If the license manager seems to be in a bad state, stop it by running the command `$VV_HOME/bin/vvlmstop`. Then verify that the license manager process is no longer running using the `ps` command. The license manager will start again the next time you run OCR Shop XTR.

You may also need to restart the license manager. For details, see "License Manager Commands" on page 83.

Step 2: Check Log Files

Vividata software generates various log files that can be useful for determining the cause of many problems.

vividata.log

When your system reboots, a file called vividata.log is created in the /tmp directory. This file contains information from the license manager used with our products. This log is useful in determining if licensing is starting at boot time correctly.

How to Get a License

Installing the Keys

To enter your license keys, use the Installer. Please see “Installing the License Keys” on page 7. Then, try restarting OCR Shop XTR.

You must have a valid license before you can use OCR Shop XTR. If you haven’t yet received a license key from us, you need to get one. You can get one by contacting Vividata Support or Vividata Sales through our website http://www.vividata.com, or by email. If you are certain that you have a valid license, verify that your licensing is set up correctly (See Appendix B, “License Manager Commands”, for license manager information.)

Installing the Keys

OCR Shop XTR license keys are normally distributed within a shell script installer, named “key.sh” or something similar. To install a key, run the script on the command-line as root, “sh key.sh”, and the license key will be placed in `$VV_HOME/config/vvlicense.dat`.

Patches

It is suggested that the operating system be maintained by installing the most current patches available from the platform vendor, as certain (possibly known) bugs can affect the operation of OCR Shop XTR. Check Vividata’s release notes and the support areas of our website for mention of any specific known problems which can be fixed with certain patches.

Appendix B: License Manager Commands

Overview

Publicly distributed versions of Vividata products use a proprietary license manager. This section will describe the usage of the license manager as it pertains to Vividata products, including determining the lmhostid necessary for license keys to be issued, diagnosing the license keys and license server, and additional configuration information.

License Manager Utilities

You will find the various license manager utilities discussed below in the $VV_HOME/bin directory after you have installed a Vividata product containing the license manager. This set of utilities currently includes vvlmhostid, vvlmreread, vvlmstatus, vvlmstop. The license manager daemon, found in the same directory, is lmgrd. (Note: in some Vividata products an alternate license manager is used. The commands are lmutil and six links to it: lmdiag, lmdown, lmhostid, lmremove, lmreread, and lmstat. In such installations, the license manager daemon is vv_d2)

The License Daemon

A license daemon runs in the background in order for the license manager to operate properly. The daemon is started automatically by the Vividata software and the process is named either “vvlicense” or has the same name as the software binary, depending on the system. The user never starts the license manager or daemon by hand. If for some reason the license daemon needs to be stopped, the “vvlmstop” utility (described below) should be used to stop the program gracefully.

License File Format

The license file is plain text and contains a long encrypted string that encodes the license(s). Usually a short text string is concatenated to the right of the license string, along with the serial number(s). More than one license for the same product may be stored in one license key. License keys for more than one Vividata product may be included on separate lines in the same license file.

Obtaining your lmhostid

If you have a Vividata product installed on your system, you can simply run "vvlmhostid" to determine the lmhostid on your system. If you need the lmhostid prior to installing the software, or if the "vvlmhostid" utility does not return a valid lmhostid, please see the following table to determine your lmhostid manually.

Table 1: lmhostid derivations

Platform	Source	User command	Example
OSF/1 Digital Unix	ethernet address	netstat -i	080020005532
HP-UX	32-bit hostid	uname -i and convert to hex or prepend with #	778DA450 or #2005771344
Linux	ethernet address	/sbin/ifconfig eth0 and remove colons from HWaddr	00400516E525
AIX	32-bit hostid	uname -m then remove last 2 digits, and use remaining last 8 digits	02765131
IRIX	32-bit hostid	/etc/sysinfo -s and convert to hex, or prepend with #	69064C3C or #1762020412

Table 1: lmhostid derivations

Platform	Source	User command	Example
SunOS and Solaris	32-bit hostif	hostid	170a3472
Windows NT	ethernet address	Programs: Administrative Tools (common): Windows NT Diagnostics: Network: Transports:Address	Programs: Administrative Tools (common): Windows NT Diagnostics

Command Reference

vvlmstat

NAME

vvlmstat – Displays the current status of the license manager.

SYNOPSIS

```
vvlmstatus
```

DESCRIPTION

vvlmstatus checks the current state of the license manager and reports how many keys are available for each product for which you have a license key.

vvlmstop

NAME

vvlmstop – Shuts down the license daemon

SYNOPSIS

```
vvlmstop
```

DESCRIPTION

vvlmstop shuts down the license manager process if it is running.

vvlmhostid

NAME

vvlmhostid – Prints the lmhostid of the system

SYNOPSIS

```
vvlmhostid
```

DESCRIPTION

vvlmhostid prints the machine id (lmhostid) of the system, usually used for generating license keys.

vvlmreread

NAME

vvlmreread – Forces the license daemon to reread the license file

SYNOPSIS

```
vvlmreread
```

DESCRIPTION

vvlmreread causes the vendor daemon to reread the license file and update itself on any new feature licensing information.

Key Read program

NAME

<product>KeyRead – Utility that decodes the features from a license key; the name varies with the product.

SYNOPSIS

```
<product>KeyRead -k [key string]
```

DESCRIPTION

The key read program permits you to view what options and licenses are encoded within the license key string. Pass the key string listed in your license file to the key read program to verify the features, number of licenses, and product enabled by that key string.

Appendix C: Glossary

Glossary term	Term definition
ADF	Automatic document feeder
ASCII	An acronym for American Standard Code for Information Interchange. A code in which the numbers from 0 to 127 stand for text characters. ASCII code is used for representing text inside a computer and for transmitting text between computers or between a computer and a peripheral device.
auto segmentation	The process in which the OCR Shop XTR determines where on a page different elements are such as where pictures are and where columns of text are.
binary image	A image that is represented using only one bit per pixel. Such images are also called black and white, monochrome, bi-level, or 1-bit.
bit-depth	The bit-depth refers to the number of bits used to describe each pixel in a bitmap. For example, an image with a bit-depth of 1 contains only black-and-white or binary image data. An image with a bit-depth of 8 appears as a grayscale image, with 8 bits of grayscale data per pixel. An image with a bit-depth of 24 appears as a color image, with 8 bits of data for each red, green, and blue sample of each pixel.
bit-mapped image	A collection of bits (dots) in memory that represent the scanned image. The display on the screen is a visible bit-mapped image.

Glossary term	Term definition
Code Page	"Code Page" is a Microsoft® term. A code page is a particular mapping of a set of unsigned bytes to a set of visible characters (and space characters). Different code pages are used to represent in memory the characters in different languages. See http://msdn.microsoft.com/en-us/goglobal/bb964653 for more details.
compound document	A compound document is a set of one or more pages that consists of a mixture of text and images, for example pdf or html.
conversion filter	A program that translates one file format into another. For example, the 'mpage' conversion filter can translate an ASCII file into a PostScript file.
device driver	A program that manages the transfer of information between the computer and a peripheral device such as a scanner.
digital image	A digital image is the way a picture or visual image of some object is represented in computer memory. A digital image consists of a number of pixels and a description of how the pixels are arranged to form the image. In addition, information about how each pixel stores the color of the original image is included.
dithering	A method of representing an image using fewer colors than the image actually has.
document	A document is a set of pages that are related usually because the sense of the text on one page flows into the next as in a book. For OCR Shop XTR, it is best to arrange for documents to be sets of pages that have the same font or set of fonts continuing from one page to the next. This best takes advantage of the internal font learning system that is built into the OCR Shop XTR recognition system.

Glossary term	Term definition
dpi	An abbreviation for dots per inch. This is the number of dots per linear inch that a printer can print or a scanner can produce. See also resolution. Sometimes its usage is as “pixels per inch”.
driver	See device driver
frame	A frame is a way to represent the maximum extent of some page element in the horizontal and vertical direction (X and Y coordinates respectively). A frame can be thought of as a rectangle that is lined up with the X and Y axes. Frames are represented by four numbers, which can be top, left, bottom, right or top, left, height, width. Also see UOR.
language pack	A language pack is a data file supplied with the OCR Shop XTR that includes information about how the characters of a given language are put together to write words and sentences in the language. Language packs contain information about the common words used in a language, rules for punctuation and the conventions used when writing things such as numbers, money amounts and dates.
language set	A language set is that set of supported languages that can be recognized with a given shape pack loaded. Each of the supported languages in a language set may or may not have an available language pack associated with it. Languages without an available language pack can still be recognized but accuracy for these languages will not be as high as for languages for which a pack exists

Glossary term	Term definition
lexical constraints	A lexical constraint is a set of restrictions on how the characters on a given page or region within a page can be recognized. Constraints can include the set of languages allowed, and/or a character set that recognition is restricted to. A lexical constraint can be a weaker preference or a stronger absolute. A custom word list can be used as an additional lexical constraint on the recognition.
lexicon	A lexicon is a list of words used in a given language and perhaps in a special setting. Language packs supplied with OCR Shop XTR contain built in general purpose lexicons. Users may specify a custom lexicon with the user_lexicon parameter.
monospaced font	Any font in which all characters have the same width. For example, in Courier New (a monospaced font), the letter “M” is the same width as the letter “l”. Thus, “MMMMM” is the same width as “lllll”.
orient	To orient a page is to rotate the page in memory so that it is better positioned for display to the user and/or recognition by the OCR Engine. A page is oriented for recognition when the text flows left to right (from low X to high X coordinates) and from top to bottom (lowY to high Y coordinates).
page	A page is the unit that makes up a document. Within OCR Shop XTR, a page is usually the representation of one side of a single piece of paper if that was input from a scanner. In addition it may be a single image, input from a file, fax machine, digital camera or other digital image input device. For purposes of licensing, a page size equivalent to a US letter size or ISO A4 is used.
peripheral	At or outside the boundaries of the computer itself, either physically (as a peripheral device) or logically (as a peripheral card).

Glossary term	Term definition
pixel	Pixel is short for picture element. A point (dot) on the graphics screen. It is the smallest definable unit of a digital image. Each pixel represents a single point in the image. The number of pixels per unit distance (dot-per-inch or DPI for instance) within a digital image is referred to as the resolution of the image. A pixel can be binary, gray, or color, or can be an index into a palette. Binary pixels require only one binary digit or bit of computer memory to store; gray, color and indexed pixels use more bits with 4, 8, and 24 being common values for the number of bits used.
point	A typographic unit of measurement equal to 1/72 inch, measured vertically. Points are used to describe font size.
proportional font	Any font in which characters differ in width. For example, in the proportional font used here, the letter “M” is wider than the letter “l”. Thus, “MMMMM” is wider than “lllll.”
rdiff file	In OCR Shop XTR a file which contains descriptions of the image and text regions of a document image.
recognize	In the context of the OCR Shop XTR, when an image is recognized, it is processed using the OCR Engine that is part of the OCR Shop XTR. During this process the pixels making up a digital image are processed by the OCR engine to determine which pixels are parts of visible text characters within the image. The identities of those characters are also determined and stored in memory using the code page representation of the given character. The result of recognition is used to create output based on the user settings.

Glossary term	Term definition
region	A region is an area of a page that usually contain either all text or all picture. Regions can be determined by the OCR Shop XTR during auto-segmentation or specified by a user in an rdiff input file.Regions on a page can overlap. Regions can be simple rectangles in shape or they can be more complex (see UOR).
resolution	The fineness with which a scanner, printer, or other device produces information. It is expressed in dots per inch (dpi). A higher dpi produces a sharper image.
shape pack	A shape pack is a data file supplied with the OCR Shop XTR that describes the shapes of the characters that can be recognized by the OCR engine when that shape pack is loaded. Each shape pack corresponds to a particular code page that will be used for output when that shape pack is loaded. For each shape pack there is an implied language set that represents the supported languages that can be recognized with that shape pack loaded.
skew	Skew is the amount of tilt in an input image.Skew is generally used to describe the tilt in images including text. In such images the tilt is more apparent and affects recognition and layout analysis.
swap file	An area of the hard disk that is used for temporary data storage when RAM is low or used up. This is also known as virtual memory. A swap file lets you run more programs than you could with actual memory, but it is slower than using regular memory.
text file	A file containing information in text form; its contents are interpreted as characters encoded using the ASCII (or comparable) format.
TIFF	An abbreviation for tagged image file format. This is a standard graphic file format for grayscale and high-resolution bit-mapped images.

Glossary term	Term definition
TrueType™ fonts	One of the major types of scalable fonts. These can be printed or displayed on the screen at any size.
Unicode	UNICODE is a standard for representing visible characters using a stream of bytes in computer memory or on some other digital storage medium. Unlike code pages where each code page can only be used to describe a subset of the known written languages, Unicode is a single standard way to represent all of the world's common written languages. Whereas the code page representation uses a single byte to represent each character, Unicode uses a 16-bit word for each character. The OCR engine that is part of OCR Shop XTR does recognition internally based on a single selected code page. During output however, the text data can be converted to Unicode for use with other applications that expect text data in Unicode format.
UOR (Union of Rectangles)	A UOR or Union Of Rectangles is the data used to represent the position and shape of a region in a region descriptor. A UOR is a list of rectangles contained in an rdiff file. The area described by the UOR is the sum or OR-ing of all the areas described by each rectangle in the list.
zone	See Region.

Index

R

S

T

U

V

W

X

Z