

**It's no trick...
it's a vision system**



**Vision
Components®**

The Smart Camera People

VC Smart Finder Version 2.5 Manual

**VC's new Smart Finder Library Overview
and Description of VC Smart Finder Demo Program**

Revision 1.5 April 2011
Document name: VCSmartFinder.pdf
© Vision Components GmbH Ettlingen, Germany

Foreword and Disclaimer

This documentation has been prepared with most possible care. However Vision Components GmbH does not take any liability for possible errors. In the interest of progress, Vision Components GmbH reserves the right to perform technical changes without further notice.

Please notify support@vision-components.com if you become aware of any errors in this manual or if a certain topic requires more detailed documentation.

This manual is intended for information of Vision Component's customers only. Any publication of this document or parts thereof requires written permission by Vision Components GmbH.

Trademarks

Code Composer Studio and TMS320C6000, Windows XP, Total Commander, Tera Term, Motorola are registered Trademarks. All trademarks are the property of their respective owners.

References









Since the VC4XXX smart camera family employs a TI processor, the programming environment and functions for the VC20XX cameras can be used for this camera.

Further References under "Support + Download" on www.vision-components.com:

„**Support News**“ – for up to date information on VC Software and Documentation.

„**Knowledge Base / FAQ**“ - searchable Database with latest software developments, frequently asked questions and demo programs.

“**Download Areas**” for all documentation and Software downloads – refer to the following table:

Description	Title on Website	Download Area
Schnellstart VC – Deutsche Version dieses Handbuchs 	 Schnellstart VC Smart Kameras	Registered User Area ▶ Getting Started VC SDK TI
Introduction to VC Smart Camera programming	 Programming Tutorial for VC20XX and VC40XX Cameras	Registered User Area ▶ Getting Started VC SDK TI
Demo programs and sample code used in the Programming Tutorial	 Tutorial_Code	Registered User Area ▶ Getting Started VC SDK TI
VC4XXX Hardware Manual	 VC4XXX Smart Cameras Hardware Documentation	Public Download Area ▶ Hardware Documentation VC Smart Cameras
VCRT Operation System Functions Manual	 VCRT 5.0 Software Manual	Registered User Area ▶ Software documentation VC Smart Cameras
VCRT Operation System TCP/IP Functions Manual	 VCRT 5.0 TCP/IP Manual	Registered User Area ▶ Software documentation VC Smart Cameras
VCLIB 2.0 /3.0 Image Processing Library Manual	 VCLIB 2.0/ 3.0 Software Manual	Registered User Area ▶ Software documentation VC Smart Cameras



The Light bulb highlights hints and ideas that may be helpful for a development.



This warning sign alerts of possible pitfalls to avoid. Please pay careful attention to sections marked with this sign.

Author: Klaus Schneider, VC Support, <mailto:support@vision-comp.com>

Table of Contents

1 Introduction	1
1.1 General features	1
1.2 System Requirements	1
2 Operation of the VC Smart Finder Demo Program	2
2.1 Executing the VC Smart Finder Demo Program	2
2.2 Parameter Adjustment	3
2.2.1 Parameter Adjustment Menu	3
2.2.2 Input Parameter	5
2.2.3 Store Parameter	6
2.2.4 Output Parameter	7
3 Working with the VC Smart Finder Library	8
3.1 One Pattern Application	8
3.2 More Pattern Application (Example for two different patterns)	8
3.3 FindPattern() Parameter Description	9
4 Further features included in the VC Smart Finder Lib	9
5 Further Development in Progress	9
6 Error Messages	10
7 Changes since last version	10
Appendix A: Test Samples	A
Appendix B: User Interface VC Smart Finder Demo	C

1 Introduction

The new VC Smart Finder is a contour-based object recognition tool. The software allows users to identify structures by means of preset patterns, e.g. bottle labels and complex components. The real-time program, which operates with subpixel accuracy, ensures high-speed processes: it recognizes between 10 and 100 objects per second at a 640 x 480 Pixel resolution. Object recognition is not influenced by rotational position, object size or illumination. Moreover, the software reliably recognizes objects which are occluded by up to 80%. An intuitive, easy-to-use teach-in option allows users to include new objects.

The software works together with VC Smart Cameras from Vision Components based on Texas Instruments DSP's

1.1 General features

- Finds object in real-time
- Subpixel accuracy (+/- 0.1 pixel and +/- 0.3 degrees depending on pattern size)
- High robustness, occlusions of up to 80% possible
- On VGA modus (640x480 resolution) it's possible to recognize up to 100 objects/sec.
- Rotation invariant
- Intuitive and very easy learning
- Highly independent on illumination
- Scale invariant (+/- 50 %)
- Multi pattern search
- etc.

1.2 System Requirements

The VC Smart Finder has been optimized for all VC40XX, VC44XX and VCSBC40XX cameras (Standard Range, Entry Level Range, High End Range and VC's Board Cameras) as well as for the VisiCube Sensor camera.

For using the VC Smart Finder on the color camera, a Bayer to gray conversion is required prior to detection. Conversion functions are available in the VC Color Lib.

The VC Smart Finder Demo program displays the matching results on the VGA output. For cameras without video interface the results can be displayed using the parallel image transfer program "img3par".

2 Operation of the VC Smart Finder Demo Program

2.1 Executing the VC Smart Finder Demo Program

1. Download the VC Smart Finder Demo from the VC Website under:
[Download Area -> Customer Download Area -> Demo Code -> VC Smart Finder](#)
2. Upload the demo program "VCSFXX" into the camera flash memory (XX = version number).
3. Execute the program. (At cameras with no video output start "img3par &" prior to program call. Then use the ATX Client for image display).
4. The following terminal printout of the main program menu is displayed:

```
Start Smart Finder. Please wait...

--- Smart Finder Version 2.5 ---

Press ENTER to start single match
Press F1    to teach pattern
Press 'n'   next match
Press 'p'   to change search area and parameter
Press 's'   to store pattern at fd:
Press 'r'   to read pattern from fd:
Press 'q'   to quit program
```

5. Place a pattern inside the blue rectangle (teach area). If necessary you can adjust the pattern area size. Just follow the instructions on the terminal printout. After the complete pattern fits in the teach area press ENTER to detect the pattern contour.

```
Please select pattern area

Press ENTER to accept values
Press F1    step size
Press F2    toggle between Position and Size
Press CURSOR to change values
```

6. With the ENTER option you simply can use all contour edges displayed in blue for pattern recognition. Otherwise extract parts of the contour by pressing key F2. The selected contour segment is displayed in green and can be added by pressing F2 again. You can skip the contour segment with any other key. All selected contour edges are displayed in red. After the relevant segments are selected just press ENTER to teach all red marked contours.

```
Press ENTER to use all edges
Press 'F2' to select edges and ENTER when ready.
```

7. By pressing ENTER you can search for the pattern and the matching result will be displayed on the camera monitor. If you type 'n' for next pattern, the demo program will search for another pattern in the same image.

Result output on the monitor (example):

```
Error=0: X=96.8 Y=356.5 Phi=32.1 Scale=100.0% Corr=99.6% Time=37ms
```

8. Press "s" if you like to save the current pattern data to the camera flash memory. The pattern will be stored with the file name "Pattern.002". You later can read the stored pattern from the flash again with the item "r".
9. In order to improve or speed up the pattern matching, you can change the VC Smart Finder parameters from the menu.

2.2 Parameter Adjustment

2.2.1 Parameter Adjustment Menu

Select from main menu item "p" in order to change VC Smart Finder parameters. In the first step you can adjust the search area position and size or just accept the defined search area by pressing ENTER. Then the parameter menu appears at the telnet output:

```
-----
p:   change search parameter
a:   Change angle parameter.
s:   Change scale parameter.
e:   Change expert parameter.

q:   quit Menu
-----
```

1. Item 'p':
Change standard search parameter in order to improve search speed and accuracy. Standard parameters are for example pyramid zoom (search area initialization), search accuracy and speed level. These parameters are easy to use. The speed level will set the expert parameters in an advisable way and allows you to optimize your system in most cases. Use lower values for speed level if calculation time is available. This offers you extra safety. The speed level option is used, if the input parameter ExpertMode is zero. For special speed or search requirements you can manually adjust the expert parameters. Therefore set ExpertMode and optimize the expert parameters. The standard parameters pyramid zoom and search accuracy will still be used.

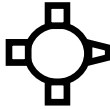
2. Item 'a':
Select the angel range (+/- degrees) from the taught pattern angle and select the angle step size. Possible ranges are for example [-180 ...180], [0 ...360], [-30 ... 30] or [0 ... 0] degrees. The search angle will only be inside this angle range and the find result is always inside the defined range. Smaller angle ranges lead to less time consumption and can speed up the system.
3. Item 's':
Select object scale range (in percent) from the taught pattern size and select the scale step size. Possible ranges are for example [-50 ... 50], [0 ... 30] or [0 ... 0] percent. The scale result is exactly in the defined range. A scale range takes much more calculation time.
4. Item 'e':
All parameter for the "search step" like image and pattern resolution, search mode and image pyramid. Use this option only, if you are familiar with the input parameters. Otherwise use the standard parameters (menu item "p") for general settings. The speed level parameter will set the expert parameters in an advisable way. Please remember, that the speed level parameter will overwrite the expert parameters, if parameter ExpertMode is zero.

For further details please refer to the chapter "Input Parameter"

2.2.2 Input Parameter

Input parameter	Default Value	Example / Meaning
LicenceCode1		VC Smart Finder license code 1 (hex value like 0x1234ABCD)
LicenceCode2		VC Smart Finder license code 2 (hex value like 0xABCD1234)
AngleMin	-180	Angle range in relation to the taught pattern [-360 .. 360]
AngleMax	180	Angle range in relation to the taught pattern [AngleMin .. AngleMin+360]
AngleStep	5	Angle step in degrees for rotation
ScaleMin	0	Scale range in relation to the taught pattern [-50 .. 0%]
ScaleMax	0	Scale range in relation to the taught pattern [0 ..+50%]
ScaleStep	10	Scale step in percent

2.2.2.1 Standard Input Parameter

PyrZoom	0	Number of pyramid zooms for search area (speed improvements) [0 = original size .. 2 = decrease input image, less accurate].
SearchAcc[0]	0	Accuracy during find step. [0 = Standard Search .. 1 = Accurate pattern search]. Accurate pattern recognition is necessary, if you have to detect pattern with small differences. For example: Detecting the correct rotation of slightly different symmetrical patterns.
		
SearchAcc[1]	1	Accuracy during adjust step. [0 = Low .. 1 = High sub pixel and angle accuracy].
SpeedLevel	5	Set the expert parameters in an advisable way. Depending on pattern or search area properties the speed steps are not necessary linear. [0 = Safe .. 9 = Fast].

2.2.2.2 Expert Input Parameter

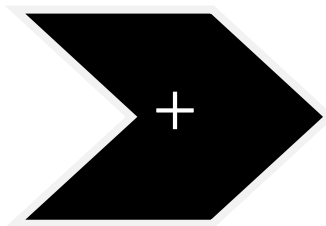
ExpertMode	0	Use SpeedLevel or expert parameters. In SpeedLevel mode, the expert parameters will be set depending on SpeedLevel value. [0 = standard .. 1 = expert parameters].
Type	4	VCLib edge() parameter. Use same parameters for teach and search.
Sigma	58	[Sigma / 10]
BinMode	2	
MinContr	5	
Thresh	20	[Thresh / 100]
AnchorTyp	1	Define Pattern Anchor Point Type Typ 1: Centre of Pattern Bounding Box (recommended) Typ 2: Centre of Pattern Contour Gravity Typ 0: Use defined AnchorX and AnchorY Points inside Pattern Area (experienced user only)
AnchorX	0	X Position of the Anchor point (only used at AnchorTyp 0)
AnchorY	0	Y Position of the Anchor point (only used at AnchorTyp 0)
PatRes[0];	4	define pixel distance between contour pixels for speed improvements:
SrcRes[0];	6	[0]: search step [1]: adjust step (final step after search step)
PatRes[1];	0	Large values reduce processor time, but are less robust.
SrcRes[1];	0	
MemLevel	2	Restrict memory consumption. [1 = high.. 3 = low memory level, but less accurate]
MemMode	1	Allow cache optimization if necessary (0 = no, 1 = low, 2 = high cache optimization)

2.2.3 Store Parameter

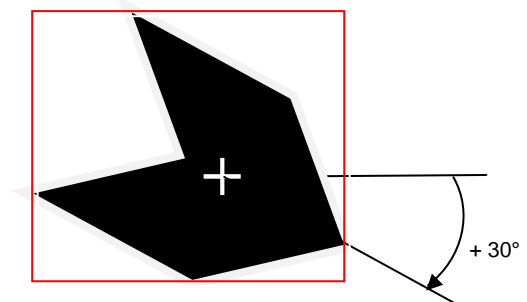
Store parameter	Example / Meaning
PatternLen	pattern data length
*PatternData	pattern data to store

2.2.4 Output Parameter

Output parameter	Example / Meaning
PosX	Anchor Point position X of detected pattern
PosY	Anchor Point position Y of detected pattern
Phi	Pattern rotation in degrees (counter clockwise)
Scale	[0.5 .. 1.5] Scale range +/- 50% from taught size
Match	[0.0 .. 1.0] Best match leads to a match result of 1.0. In case that only half of the pattern contour is available, the match result will be 0.5 (correlation of 50%).
BoundingX	Aligned pattern bounding box X
BoundingY	Aligned pattern bounding box Y
BoundingDx	Aligned pattern bounding box Dx
BoundingDy	Aligned pattern bounding box Dy
PatListLenXY	Pattern length for XY-List.
LibVersion	Smart Finder Library Version (25 = Version 2.5)
Error	Pattern matching result error



Pattern with Anchor Point



Found Pattern Position with 30 °rotation
and bounding box

3 Working with the VC Smart Finder Library

3.1 One Pattern Application

```

PatternPar PatPar[1];                                     // define pattern structure

SetParameter(PatPar);                                     // set pattern parameter
InitSmartFinder(PatPar);                                  // initialize pattern memory

TeachPattern(PatPar, &Pattern);                           // teach pattern
ReadPattern (PatPar, "fd:/Pattern.dat");                  // OR read pattern from file

LOOP 1
    FindPattern(PatPar, &SearchArea, FIND_PAT);           // find first pattern

LOOP 2
    FindPattern(PatPar, &SearchArea, DELETE_PAT);          // delete last pattern
    FindPattern(PatPar, &SearchArea, NEXT_PAT);            // find next pattern in same search image
END LOOP 2

ChangeParameter (PatPar);                                 // if necessary, change some parameter
TeachParameterChange(PatPar);                             // important: call this function after parameter changes
END LOOP 1

DeinitSmartFinder(PatPar);                                 // release pattern memory

```

3.2 More Pattern Application (Example for two different patterns)

```

PatternPar PatPar[2];                                     // define pattern structure

For (i=0; i<2; i++) SetParameter(&PatPar[i]);             // set pattern parameter
For (i=0; i<2; i++) InitSmartFinder(&PatPar[i]);          // initialize pattern memory

For (i=0; i<2; i++) TeachPattern(&PatPar[i], &Pattern[i]); // teach pattern

FindPattern(&PatPar[0], &SearchArea, FIND_PAT);           // find pattern 0
FindPattern(&PatPar[1], &SearchArea, NEXT_PAT);            // find pattern 1 in same search image

FindPattern(&PatPar[0], &SearchArea, DELETE_PAT);          // delete pattern 0
FindPattern(&PatPar[0], &SearchArea, NEXT_PAT);            // find next pattern 0 in same search image

FindPattern(&PatPar[0], &SearchArea, DELETE_PAT);          // delete pattern 0
FindPattern(&PatPar[0], &SearchArea, NEXT_PAT);            // find next pattern 0 in same search image

FindPattern(&PatPar[1], &SearchArea, DELETE_PAT);          // delete pattern 1
FindPattern(&PatPar[1], &SearchArea, NEXT_PAT);            // find next pattern 1 in same search image

For (i=0; i<2; i++) DeinitSmartFinder(&PatPar[i]);        // release pattern memory

```

3.3 FindPattern() Parameter Description

FIND_PAT	Initialize search area and find selected pattern. The search result will be stored in the output parameters PosX, PosY, Phi, Scale and Match.
DELETE_PAT	Delete pattern from initialized search area at the position defined in the output parameters PosX, PosY, Phi and Scale.
NEXT_PAT	Search for a pattern in the already initialized search area.
FREE_MEM	Release memory of the initialized search area. The parameter is only necessary, if you are short in memory. Otherwise FREE_MEM is not necessary to use. It will be called automatically at FIND_PAT or in the function DeinitSmartFinder() at the end of the program.

4 Further features included in the VC Smart Finder Lib

	Features Demo Program	Features Library
Multi Pattern Search (same pattern)	✓	✓
Multi Pattern Search (different pattern)		✓
Data export via TCP/IP or serial RS232		✓
Display of parameters and results on VGA output: <ul style="list-style-type: none"> - Pattern Position - Pattern Angle - Scale Factor - Correlation Value - Processing time (no image acquisition) - Anchor Point - Bounding Box - Full Error code output 	<ul style="list-style-type: none"> ✓ ✓ ✓ ✓ ✓ ✓ ✓ 	<ul style="list-style-type: none"> ✓ ✓ ✓ ✓ ✓ ✓ ✓

5 Further Development in Progress

Additional features will soon be available in the VC Smart Finder Lib:

- Further improvements in matching speed, robustness and accuracy
-

6 Error Messages

Error Code	Meaning
0	No Error
-10xx	Memory allocation Error during program initialisation
-11xx	Memory allocation Error during running time
-2000	licence code error
-2001	VC libraries doesn't fit to SmartFinder lib
-2002	No Pattern Contour defined
-2003	Search Area not ready (Missing initialized search area. Use FIND_PAT first)
-2004	Image Format Error (the input teach image is no edge() image)
-2005	InitSmartFinder() missing. Please call this function first.
-2006	No Pattern taught.
-2007	Pattern requests different Pyramid Zoom. The PyrZoom Level in the initialized search area is different from the one selected in the pattern. Use either FIND_PAT for a new initialisation of the search area or change the pattern input parameter PyrZoom so that it fits to the initialised search area (occurs only in Mode NEXT_PAT or DELET_PAT).

7 Changes since last version

Version 23:

- scale independent
- accurate bounding box
- speed improvements
- easy setup parameters [0..9]
- cache optimization
- flexible anchor point
- generate xy list for pattern position

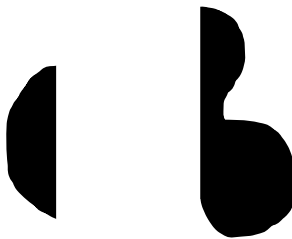
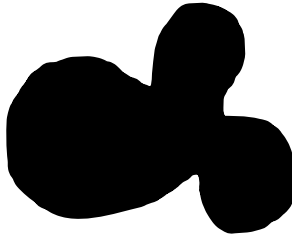
Version 24:

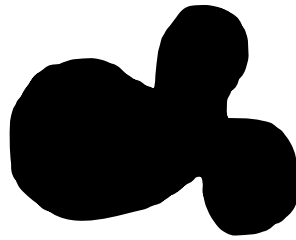
- scale mode: detecting pattern with better correlation values first, independent of pattern size

Version 25:

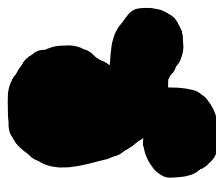
- 1000 runs and 90 min time limit in demo mode
- include lib version in PatPar struct (PatPar->LibVersion)

Appendix A: Test Samples

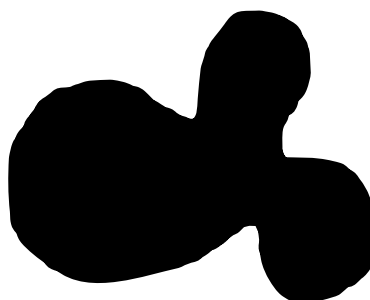




Size 100 %

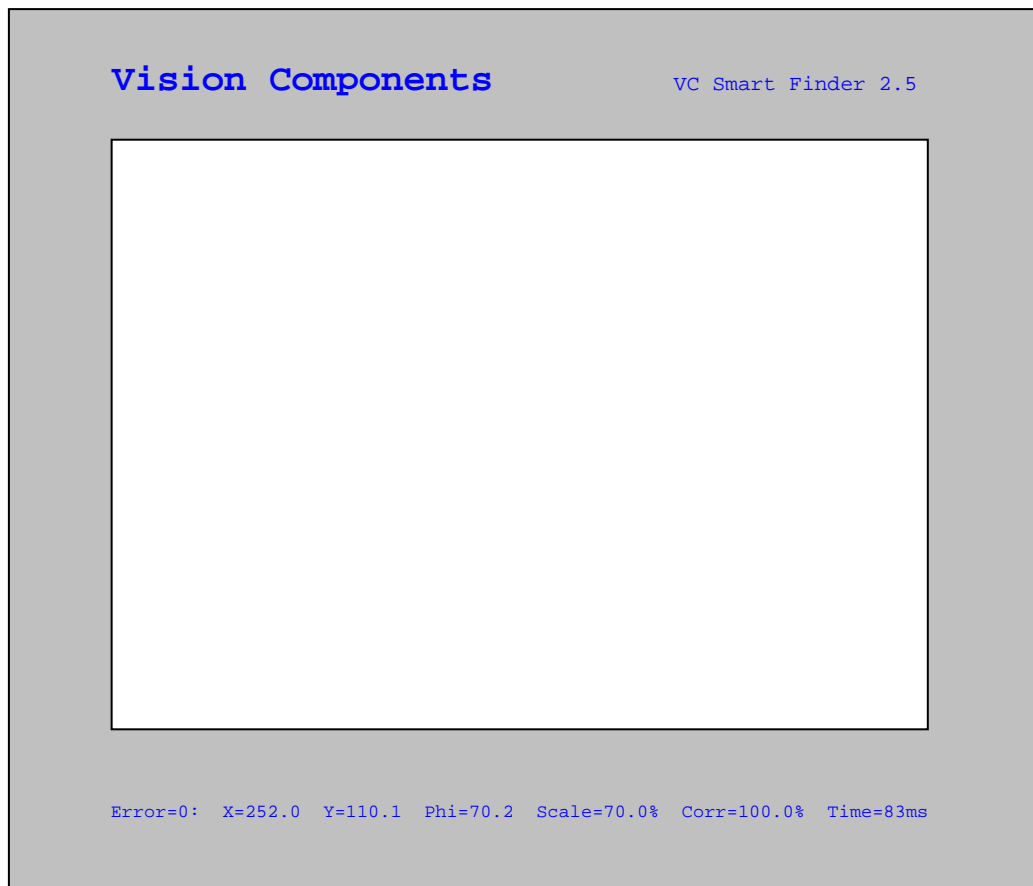


Size 70 %



Size 130 %

Appendix B: User Interface VC Smart Finder Demo



It's no trick... it's a vision system

Visit the Vision Components site www.vision-components.com for further information and documentation and software downloads:

Web Site Menu Links	Content
Contact	Distributor list / Enquiry forms
Home	Latest News from VC
Our Company	VC Company Information
News and Events	Trade Show dates VC Publications Sign in for free VC Seminars
VC Network	Description of Partner Companies Application Overview 3 rd Party Hard- and SW Products for VC Smart Cameras
Products VC Smart Camera Overview	Product Overview: VC44XX High End Camera Series VC40XX Standard Camera Series VC4016 / 18 Entry Level Cameras VC4002L Line Scan Camera VCSBC Single Board Cameras VC20XX Smart Cameras VCSBC Board Cameras VCM + Viscube Camera Sensors
VC Smart Camera Software VC Software Development Kit Ti:	VCRT Operating System VCLIB Image Processing Library
VC Special Libraries:	M200 Data Matrix Code Finder VCOCR Text Recognition Library Color Lib
Support: Support News (User Registration required) Knowledge Base / FAQ (User Registration required) Download Area	Tech News – new SW and Documentation Searchable FAQ Database with programming Examples and Demo Code Download of:
Public Download Area (free Access) Registered User Area (User Registration required) Customer Download Area (User- and SW License Registration required) RMA Number Form	<ul style="list-style-type: none"> - Product Brochures    - Camera Manuals - Getting Started    - Programming Manuals - Training Manuals and Demo Code - Software Updates - Demo Code Form for Allocation of Repair Numbers.