webcam Reference Manual

1.0

Generated by Doxygen 1.5.1

Thu Oct 25 12:35:12 2007

Contents

1 Philips SPC 900 NC - OpenCV Webcam demonstration

1.1 Introduction

These programs demonstrate how to use the Philips SPC 900 NC webcam with its features (gain control, shutter speed, ...) along with the OpenCV library. This could be useful to anybody who is doing computer vision and wants to run a low budget but high quality CCD web camera (the SPC 900 NC) under Linux with small efforts. **If you find this package useful please put a link to my site** http://www.rainsoft.de **and let me know for what you are using it.**

1.2 Installation

1.2.1 Step 1: Download the package

Download the package files from http://www.rainsoft.de/projects/pwc.html

1.2.2 Step 2: Extract the package

Extract the webcam package to your home directory.

1.2.3 Step 3: Check for OpenCV and ffmpeg

Check if you have installed OpenCV and ffmpeg properly. For details refer to http://rainsoft.de/projects/ffmpeg_opencv.html

1.2.4 Step 4: Correct paths

This package assumes that the OpenCV include files are accessible in /usr/local/include/opencv. If this is not the case you will have to modify the Makefile. cv.h, highgui.h must be accessible directly by the compiler. Likewise The libraries cv, cvaux, cxcore, highgui are assumed to be found. This should be Ok for normal OpenCV installations. For me it works with Ubuntu 7.04 and OpenCV 1.0.

1.2.5 Step 5: Build the project

Build the demo files grabber_demo, simple_demo and canny_demo using make. You can use make clean to clean the project.

1.2.6 Step 6: Have fun

For more details visit http://www.rainsoft.de/projects/pwc.html

2 webcam Class Documentation

2.1 CvButtons Class Reference

Class CvButtons

Implements functions to enhance the OpenCV GUI elements

by simple, platform-independet push buttons and toggle elements.

```
#include <cv-buttons.h>
```

Public Member Functions

- CvButtons ()
- ∼CvButtons ()
- void setMouseState (int e, int x, int y, int f)
- void paintButtons (IplImage ∗img)
- void addButton (PushButton pb)
- void delButton (int pos)

2.1.1 Detailed Description

Class CvButtons

Implements functions to enhance the OpenCV GUI elements

by simple, platform-independet push buttons and toggle elements.

Author:

Andreas Geiger
Karlsruhe Institute of Technology

Version:

1.0

Date:

16.07.2007

2.1.2 Constructor & Destructor Documentation

2.1.2.1 CvButtons::CvButtons () [inline]

Constructor creates button font

2.1.2.2 CvButtons::~CvButtons () [inline]

Deconstructor clears the button list

2.1.3 Member Function Documentation

2.1.3.1 void CvButtons::setMouseState (int *e*, int *x*, int *y*, int *f*) [inline]

Called by cvButtonsOnMouse() when button was pressed

2.1.3.2 void CvButtons::paintButtons (IplImage * *img*)

Paint all buttons to an image

2.1.3.3 void CvButtons::addButton (PushButton *pb*) [inline]

Add button to list

2.1.3.4 void CvButtons::delButton (int *pos*) [inline]

Delete button from list

The documentation for this class was generated from the following files:

- webcam/cv-buttons.h
- webcam/cv-buttons.cpp

2.2 CvCaptureCAM_V4L Struct Reference

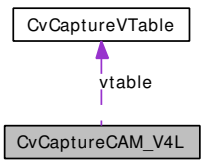
Struct CvCaptureCAM_V4L

Structure copied from OpenCV which is used to extract

the device handle each time a camera function is used.

`#include <pwc-wrapper.h>`

Collaboration diagram for CvCaptureCAM_V4L:



2.2.1 Detailed Description

Struct CvCaptureCAM_V4L

Structure copied from OpenCV which is used to extract

the device handle each time a camera function is used.

Author:

Andreas Geiger
Karlsruhe Institute of Technology

Version:

1.0

Date:

25.10.2007

The documentation for this struct was generated from the following file:

- webcam/pwc-wrapper.h

2.3 CvCaptureVTable Struct Reference

Struct CvCaptureVTable

Structure used by CvCaptureCAM_V4L

`#include <pwc-wrapper.h>`

2.3.1 Detailed Description

Struct CvCaptureVTable

Structure used by CvCaptureCAM_V4L

Author:

Andreas Geiger
Karlsruhe Institute of Technology

Version:

1.0

Date:

25.10.2007

The documentation for this struct was generated from the following file:

- webcam/pwc-wrapper.h

2.4 PushButton Class Reference

Class PushButton

Implements a single push button object.

`#include <cv-buttons.h>`

Public Member Functions

- PushButton (int x, int y, int w, int h, int t, char *text, void(*cb)(int))

Public Attributes

- int x_pos
- int width
- int toggle
- char * text
- void(* cb)(int)

2.4.1 Detailed Description

Class PushButton

Implements a single push button object.

Author:

Andreas Geiger
Karlsruhe Institute of Technology

Version:

1.0

Date:

16.07.2007

2.4.2 Constructor & Destructor Documentation

2.4.2.1 PushButton::PushButton (int *x*, int *y*, int *w*, int *h*, int *t*, char * *text*, void(*)(int) *cb*) `[inline]`

Constructor takes parameters such as:

- **x** , **y** : x/y position of a push button
- **w** , **h** : width/height of a push button
- **t:** -1 if normal button or 0/1 as state of a toggle button

- **text:** button description
- **cb:** button callback function. Takes a function pointer. The argument will be the button toggle state when pressed.

2.4.3 Member Data Documentation

2.4.3.1 int PushButton::x_pos

x/y position of a push button

2.4.3.2 int PushButton::width

width/height of a push button

2.4.3.3 int PushButton::toggle

-1 if normal button or 0/1 as state of a toggle button

2.4.3.4 char* PushButton::text

button description

2.4.3.5 void(* PushButton::cb)(int)

button callback function. Takes a function pointer.

The documentation for this class was generated from the following file:

- webcam/cv-buttons.h

2.5 PwcWrapper Class Reference

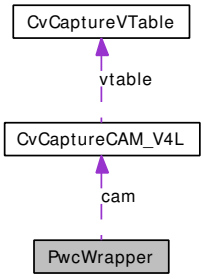
Class PwcWrapper

Implements functions to enhance the OpenCV camera driver

by the Philips Web Camera specific functions.

`#include <pwc-wrapper.h>`

Collaboration diagram for PwcWrapper:



Public Member Functions

- PwcWrapper (CvCapture *camera_device)
- ~PwcWrapper ()
- int GetFrameRate ()
- void SetFrameRate (int fps)
- void SaveUserSettingsToEEPROM ()
- void RestoreUserSettingsFromEEPROM ()
- void ResetEEPROM ()
- int GetCompressionMode ()
- void SetCompressionMode (int newmode)
- int GetAutomaticGainControl ()
- void SetAutomaticGainControl (int newagc)
- void SetShutterSpeed (int speed)
- void SetWhitebalance (int mode, int red, int blue)
- void SetAutomaticWhiteBalanceSpeed (int speed, int delay)
- void SetCameraLED (int on_time, int off_time)
- void SetElectronicSharpness (int newvalue)
- void SetBacklightCompensationMode (int newmode)
- void SetAntiFlickerMode (int newmode)
- void SetDynamicNoiseReductionMode (int newmode)
- void PrintRealImageSize ()

2.5.1 Detailed Description

Class PwcWrapper

Implements functions to enhance the OpenCV camera driver

by the Philips Web Camera specific functions.

Author:

Andreas Geiger
Karlsruhe Institute of Technology

Version:

1.0

Date:

25.10.2007

2.5.2 Constructor & Destructor Documentation

2.5.2.1 PwcWrapper::PwcWrapper (CvCapture * *camera_device*) [inline]

Constructor. The opened OpenCV capturing device (where an attached Philips SPC 900 NC is assumed) has to be passed.

2.5.2.2 PwcWrapper::~PwcWrapper () [inline]

Deconstructor.

2.5.3 Member Function Documentation

2.5.3.1 int PwcWrapper::GetFrameRate ()

Get frame rate (should be between 5 and 30).

2.5.3.2 void PwcWrapper::SetFrameRate (int *fps*)

Set frame rate (should be between 5 and 30).

2.5.3.3 void PwcWrapper::SaveUserSettingsToEEPROM ()

Save settings to camera interal memory (Attention: works only about 10.000x). When the camera is plugged off and in again the parameters will be those which have been saved during this process. Note that not all parameters will be stored!

2.5.3.4 void PwcWrapper::RestoreUserSettingsFromEEPROM ()

Restore settings from camera interal memory.

2.5.3.5 void PwcWrapper::ResetEEPROM ()

Reset EEPROM to default values. (Attention: works only about 10.000x).

2.5.3.6 int PwcWrapper::GetCompressionMode ()

Get compression ratio of the videostream transported by USB. 0 means no compression, 3 means high compression. Each value in between is valid, too.

2.5.3.7 void PwcWrapper::SetCompressionMode (int *newmode*)

Set compression ratio of the videostream transported by USB. 0 means no compression, 3 means high compression. Each value in between is valid, too.

2.5.3.8 int PwcWrapper::GetAutomaticGainControl ()

Get the value which once was set by **PwcWrapper::SetAutomaticGainControl**.

2.5.3.9 void PwcWrapper::SetAutomaticGainControl (int *newagc*)

Philips webcams have features like auto-exposure and some additional circuitry to accomodate for changing light conditions. One of these features is an AGC circuit that amplifies or attenuates the signal that comes from the CCD/CMOS sensor. Normally this circuit is in auto mode, but you can set it to a fixed value if you like. The range is 0..65535. 0 means low gain, and 65535 sets it to highest gain possible. If you supply a negative number (i.e. -1), the AGC is set to automatic mode.

2.5.3.10 void PwcWrapper::SetShutterSpeed (int *speed*)

Set the shutter speed of a Philips capturing device. The range is: 0..65535. Setting shutter speed and agc to -1 means automatic adjustment.

2.5.3.11 void PwcWrapper::SetWhitebalance (int *mode*, int *red*, int *blue*)

The white balance is the ability to correct for different lighting conditions (outdoor, indoor, artificial lighting, etc), by adjusting the gains for the red and blue pixels (green is never affected). **Mode** can be one of the following: PWC_WB_AUTO, PWC_WB_-MANUAL, PWC_WB_INDOOR, PWC_WB_OUTDOOR, PWC_WB_FL. The last mode means "fluorescent lighting". Only in manual mode the red and blue parameters can be used to adjust the gain manually. The range is 0..65535.

2.5.3.12 void PwcWrapper::SetAutomaticWhiteBalanceSpeed (int *speed*, int *delay*)

Sets speed and delay which determine how fast the camera reacts to changes in lighting when it is in automatic mode. The range is 1..65535. 0 leaves the settings untouched. The higher the value speed, the slower is the reaction.

2.5.3.13 void PwcWrapper::SetCameraLED (int *on_time*, int *off_time*)

Controlling the leds is only supported by the ToUCam series. It doesn't work with the leds of the SPC 900 NC.

2.5.3.14 void PwcWrapper::SetElectronicSharpness (int *newvalue*)

You can electronically blur or sharpen frames coming from the web cam a little bit. 0 means blurring, 65535 means sharpening.

2.5.3.15 void PwcWrapper::SetBacklightCompensationMode (int *newmode*)

Compensate for a very bright background (where the object in the foreground is too dark). 0 means on, every other value means off.

2.5.3.16 void PwcWrapper::SetAntiFlickerMode (int *newmode*)

Due to the different frequencies of the webcam, monitor refresh rate and the electrical power supply, the intensity of the image may 'pulsate' which is quite annoying to watch. The Philips cams have a way to suppress this. This function turns that feature on or off. A value of 0 turns if off, any other value means that it is switched on.

2.5.3.17 void PwcWrapper::SetDynamicNoiseReductionMode (int *newmode*)

0 means no noise reduction filtering, 3 means highest noise reduction filtering switched on. Any value in between is valid, too.

2.5.3.18 void PwcWrapper::PrintRealImageSize ()

Plots the real frame size to the console to check if frame size has been changed correctly, etc...

The documentation for this class was generated from the following files:

- webcam/pwc-wrapper.h
- webcam/pwc-wrapper.cpp

Index