CamShift Documentation

Generated by Doxygen 1.8.5

Sun Jun 15 2014 14:50:07

Contents

1 Namespace Index **1**

1.1 Namespace List 1

2 Class Index **3**

2.1 Class List 3

3 Namespace Documentation **5**

3.1 camShift Namespace Reference 5

3.1.1 Detailed Description 5

4 Class Documentation **7**

4.1 camShift::CamShift Class Reference 7

4.1.1 Detailed Description 8

4.1.2 Member Function Documentation 8

4.1.2.1 getBackprojection 8

4.1.2.2 getParameter 8

4.1.2.3 getRotatedTrack 9

4.1.2.4 getTrack 9

4.1.2.5 runCamShift 9

4.1.2.6 setCapturedRawFrame 10

4.1.2.7 setParameter 10

4.1.2.8 setSelection 11

Index **12**

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

camShift
Contains the CamShift class 5

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

camShift::CamShift
Carries out the CAMShift algorithm, utilizing OpenCV libraries 7

Chapter 3

Namespace Documentation

3.1 camShift Namespace Reference

Contains the CamShift class.

Classes

- class CamShift

  Carries out the CAMShift algorithm, utilizing OpenCV libraries.

3.1.1 Detailed Description

Contains the CamShift class.

Author

Andrew Powell

Date

June 14th, 2014

Chapter 4

Class Documentation

4.1 camShift::CamShift Class Reference

Carries out the CAMShift algorithm, utilizing OpenCV libraries.

`#include <CamShift.h>`

Public Types

- enum { **THRESHOLD_MAXI** = 255 }

  Largest Threshold value.
- enum Parameter {
  HUE_BINS_C, **SAT_BINS_C**, **VAL_BINS_C**, **MEDIAN_BLUR_C**,
  THRESHOLD_C }

  An enumerator type used to specify a parameter to change and view with the setParameter() and getParameter() methods, respectively.

Public Member Functions

- CamShift ()

  Constructor.
- ~CamShift ()

  Destructor.
- void setCapturedRawFrame (cv::Mat &capturedRawFrame)

  Sets the captured raw frame.
- void setSelection (cv::Rect &selection)

  Sets the selection window.
- void runCamShift ()

  Executes the CAMShift algorithm and other operations intended to optimize the results.
- cv::Mat & getBackprojection ()

  Gets the backprojection.
- cv::Rect & getTrack ()

  Gets the track window.
- cv::RotatedRect & getRotatedTrack ()

  Gets the rotated track window.
- void setParameter (Parameter parameter, long newParameter)

  Sets a specified parameter.
- long getParameter (Parameter parameter)

  Gets the value of a specified parameter.

4.1.1 Detailed Description

Carries out the CAMShift algorithm, utilizing OpenCV libraries.

The CamShift class relies on Gary Bradsky's Continuously Adaptive Meanshift (CAMshift) algorithm implemented within OpenCV libraries. More information on the algorithm itself can be found at the webpage whose hyperlink is located below. A brief description of the CAMShift algorithm is presented here.

The CAMShift algorithm is, in essence, an adaptive version of the Meanshift algorithm. Say we have a set of points and a window that encapsulates a subset of the set of points. The set of points could be a set of pixels of an image, and the window could be a circle or another shape. The Meanshift algorithm takes the window and shifts the window such that the maximum point density is achieved in the window. The CAMShift algorithm extends the Meanshift algorithm by also changing the size and rotation of the window.

By continuously producing a different image in which the desired object to track has the highest pixel density (i.e. the backprojection) and applying the CAMShift algorithm, the resulting windows represent where the object is located in each image.

In addition to the operations related to the CAMShift algorithm, the CamShift class also carries out several filtration methods so as to optimize the results of the algorithm. Please keep in mind the the CamShift generates backprojections highly based on the color of the desired object.

See Also

Meanshift and Camshift

Author

Andrew Powell

Date

June 14th, 2014

4.1.2 Member Function Documentation

4.1.2.1 cv::Mat & camShift::CamShift::getBackprojection ( )

Gets the backprojection.

Returns

Returns a reference to the backprojection

Exceptions

runtime_error	The runtime error is thrown in the event the backprojection has not been set.

Warning

runCamShift() should be called prior to calling getBackprojection().

4.1.2.2 long camShift::CamShift::getParameter (**Parameter** *parameter*)

Gets the value of a specified parameter.

Parameters

parameter	Specifies which parameter to return

Returns

Returns the value of a specified parameter

4.1.2.3 cv::RotatedRect & camShift::CamShift::getRotatedTrack ( )

Gets the rotated track window.

In the context of the CamShift class, selection refers to the window manually set with the setSelection() method, whereas track refers to the window calculated as a result of the runCamShift() method.

The difference between track and a rotated track is as follows. The rotated track window is the true window produced by running the CAMShift algorithm implemented within OpenCV. The track is the rotated track's bounding rectangle, which is never rotated.

Returns

Returns a reference to the rotated track window

Exceptions

runtime_error	A runtime_error is thrown in the event track has not been set.

Warning

runCamShift() should be called prior to calling getRotatedTrack().

4.1.2.4 cv::Rect & camShift::CamShift::getTrack ( )

Gets the track window.

In the context of the CamShift class, selection refers to the window manually set with the setSelection() method, whereas track refers to the window calculated as a result of the runCamShift() method.

The difference between track and a rotated track is as follows. The rotated track window is the true window produced by running the CAMShift algorithm implemented within OpenCV. The track is the rotated track's bounding rectangle, which is never rotated.

Returns

Returns a reference to the track window

Exceptions

runtime_error	A runtime_error is thrown in the event track has not been set.

Warning

runCamShift() should be called prior to calling getTrack().

4.1.2.5 void camShift::CamShift::runCamShift ( )

Executes the CAMShift algorithm and other operations intended to optimize the results.

For every new captured raw frame, the runCamShift() should be called in order to determine a new window.

Exceptions

runtime_error	The runtime error is thrown if the captured raw frame has not been set.

Warning

The methods setSelection() and setCapturedRawFrame() should be called at least once prior to calling runCamShift().

4.1.2.6 void camShift::CamShift::setCapturedRawFrame (cv::Mat & *capturedRawFrame*)

Sets the captured raw frame.

The captured raw frame is the image over which the CAMShift algorithm is executed. If the CamShift class is being employed to determine the location of an object in real-time, the captured raw frame should be set to every new frame.

Parameters

capturedRawFrame	A reference to the image over which the CAMShift algorithm is executed

Warning

setCapturedRawFrame() should be called prior to calling setSelection() and runCamShift().

4.1.2.7 void camShift::CamShift::setParameter (Parameter *parameter,* long *newParameter*)

Sets a specified parameter.

Parameters:

Parameter	Description	Valid Range
HUE_BINS_C	Sets the number of hue bins in the histogram	0 to 179
SAT_BINS_C	Sets the number of saturation bins in the histogram	0 to 255
VAL_BINS_C	Sets the number of value bins in the histogram	0 to 255
MEDIAN_BLUR_C	Sets the size of median blur	odd values greater than 1
THRESHOLD_C	Sets the threshold value	0 to 179

Description:

setParameter() can configure parameters that affect how the backprojection is generated. The backprojections are each generated from a histogram produced once the selection window is set. The histogram is calculated from the selection window and the captured raw frame that has been converted from RGB (i.e. red, green, and blue) to HSV (i.e. hue, saturation, and value). Hue is indicative to color, saturation is indicative to where the color is on the gray scale, and value refers to brightness. The number of bins for each channel (i.e. hue, saturation, and value) effectivly changes how well and how poorly the backprojections capture the desired object.

Parameters

parameter	Specifies which parameter to modify
newParameter	The new value to which the specified parameter is changed

Exceptions

runtime_error	A runtime error is thrown if an attempt is made to set the specified parameter to an invalid value.

See Also

OpenCV's median blur

4.1.2.8 void camShift::CamShift::setSelection (cv::Rect & *selection*)

Sets the selection window.

In the context of the CamShift class, selection refers to the window manually set with the setSelection() method, whereas track refers to the window calculated as a result of the runCamShift() method.

Parameters

selection	A reference to the rectangle that acts as the new window

Exceptions

runtime_error	A runtime error is thrown if the selection is invalid. Specifically, the selection's width and height both must be greater than 0. Moreover, a runtime error is thrown if the captured raw frame has not been set.

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

- Camshift/CamShift.h
- Camshift/CamShift.cpp

Index

camShift, 5
camShift::CamShift, 7
    getBackprojection, 8
    getParameter, 8
    getRotatedTrack, 9
    getTrack, 9
    runCamShift, 9
    setCapturedRawFrame, 10
    setParameter, 10
    setSelection, 11

getBackprojection
    camShift::CamShift, 8
getParameter
    camShift::CamShift, 8
getRotatedTrack
    camShift::CamShift, 9
getTrack
    camShift::CamShift, 9

runCamShift
    camShift::CamShift, 9

setCapturedRawFrame
    camShift::CamShift, 10
setParameter
    camShift::CamShift, 10
setSelection
    camShift::CamShift, 11