

IMAR-C

0.1

Generated by Doxygen 1.6.3

Wed Jul 17 23:46:47 2013



# Contents

<b>1</b>	<b>Class Index</b>	<b>1</b>
1.1	Class Hierarchy . . . . .	1
<b>2</b>	<b>Class Index</b>	<b>3</b>
2.1	Class List . . . . .	3
<b>3</b>	<b>File Index</b>	<b>5</b>
3.1	File List . . . . .	5
<b>4</b>	<b>Class Documentation</b>	<b>9</b>
4.1	activitiesMap Class Reference . . . . .	9
4.1.1	Detailed Description . . . . .	9
4.2	Box< T > Class Template Reference . . . . .	10
4.2.1	Detailed Description . . . . .	19
4.3	Cache Class Reference . . . . .	20
4.3.1	Detailed Description . . . . .	20
4.4	decision_function Struct Reference . . . . .	21
4.4.1	Detailed Description . . . . .	21
4.5	DescInfo Struct Reference . . . . .	22
4.5.1	Detailed Description . . . . .	22
4.6	DescMat Struct Reference . . . . .	23
4.6.1	Detailed Description . . . . .	23
4.7	exec_time Struct Reference . . . . .	24
4.7.1	Detailed Description . . . . .	24
4.8	IplImagePyramid Class Reference . . . . .	25
4.8.1	Detailed Description . . . . .	27
4.8.2	Constructor & Destructor Documentation . . . . .	27
4.8.2.1	IplImagePyramid . . . . .	27
4.8.2.2	IplImagePyramid . . . . .	27

4.8.2.3	IplImagePyramid	27
4.8.2.4	IplImagePyramid	27
4.8.2.5	IplImagePyramid	27
4.8.2.6	IplImagePyramid	28
4.8.2.7	IplImagePyramid	28
4.8.2.8	IplImagePyramid	28
4.8.2.9	IplImagePyramid	28
4.8.2.10	IplImagePyramid	28
4.8.3	Member Function Documentation	28
4.8.3.1	getImage	28
4.8.3.2	getImage	28
4.8.3.3	getImage	28
4.8.3.4	getImage	28
4.8.3.5	getImage	29
4.8.3.6	getIndex	29
4.8.3.7	getIndex	29
4.8.3.8	getIndex	29
4.8.3.9	getIndex	29
4.8.3.10	getIndex	29
4.8.3.11	rebuild	29
4.8.3.12	rebuild	29
4.8.3.13	rebuild	29
4.8.3.14	rebuild	30
4.8.3.15	rebuild	30
4.9	IplImageWrapper Class Reference	31
4.9.1	Detailed Description	33
4.10	KCleaf Class Reference	34
4.10.1	Detailed Description	34
4.11	KCnode Class Reference	35
4.11.1	Detailed Description	36
4.12	KCsplrit Class Reference	37
4.12.1	Detailed Description	37
4.13	KCtree Class Reference	38
4.13.1	Detailed Description	38
4.14	Kernel Class Reference	39
4.14.1	Detailed Description	39

4.15	KMcenters Class Reference . . . . .	40
4.15.1	Detailed Description . . . . .	41
4.16	KMdata Class Reference . . . . .	42
4.16.1	Detailed Description . . . . .	42
4.17	KMfilterCenters Class Reference . . . . .	43
4.17.1	Detailed Description . . . . .	44
4.18	KMlocal Class Reference . . . . .	45
4.18.1	Detailed Description . . . . .	46
4.19	KMlocalEZ_Hybrid Class Reference . . . . .	47
4.19.1	Detailed Description . . . . .	47
4.20	KMlocalHybrid Class Reference . . . . .	48
4.20.1	Detailed Description . . . . .	49
4.21	KMlocalLloyds Class Reference . . . . .	50
4.21.1	Detailed Description . . . . .	50
4.22	KMlocalSwap Class Reference . . . . .	51
4.22.1	Detailed Description . . . . .	51
4.23	KMorthRect Class Reference . . . . .	52
4.23.1	Detailed Description . . . . .	52
4.24	KMterm Class Reference . . . . .	53
4.24.1	Detailed Description . . . . .	54
4.25	ONE_CLASS_Q Class Reference . . . . .	55
4.25.1	Detailed Description . . . . .	55
4.26	Point< T > Class Template Reference . . . . .	56
4.26.1	Detailed Description . . . . .	57
4.27	PointDesc Class Reference . . . . .	58
4.27.1	Detailed Description . . . . .	58
4.28	QMatrix Class Reference . . . . .	59
4.28.1	Detailed Description . . . . .	59
4.29	Size< T > Class Template Reference . . . . .	60
4.29.1	Detailed Description . . . . .	62
4.30	Solver::SolutionInfo Struct Reference . . . . .	63
4.30.1	Detailed Description . . . . .	63
4.31	Solver Class Reference . . . . .	64
4.31.1	Detailed Description . . . . .	65
4.32	Solver_NU Class Reference . . . . .	66
4.32.1	Detailed Description . . . . .	66

4.33	SVC_Q Class Reference . . . . .	67
4.33.1	Detailed Description . . . . .	67
4.34	svm_model Struct Reference . . . . .	68
4.34.1	Detailed Description . . . . .	68
4.35	svm_node Struct Reference . . . . .	69
4.35.1	Detailed Description . . . . .	69
4.36	svm_parameter Struct Reference . . . . .	70
4.36.1	Detailed Description . . . . .	70
4.37	svm_problem Struct Reference . . . . .	71
4.37.1	Detailed Description . . . . .	71
4.38	SVR_Q Class Reference . . . . .	72
4.38.1	Detailed Description . . . . .	72
4.39	Tactil Class Reference . . . . .	73
4.39.1	Detailed Description . . . . .	73
4.39.2	Member Function Documentation . . . . .	73
4.39.2.1	init . . . . .	73
4.39.2.2	onFrontHeadTouched . . . . .	73
4.40	temps_exec Struct Reference . . . . .	74
4.40.1	Detailed Description . . . . .	74
4.41	Track Class Reference . . . . .	75
4.41.1	Detailed Description . . . . .	75
4.42	TrackerInfo Struct Reference . . . . .	76
4.42.1	Detailed Description . . . . .	76
<b>5</b>	<b>File Documentation</b>	<b>77</b>
5.1	naodensetrack.cpp File Reference . . . . .	77
5.1.1	Detailed Description . . . . .	78
5.1.2	Function Documentation . . . . .	78
5.1.2.1	extractSTIPs . . . . .	78
5.2	naokmeans.cpp File Reference . . . . .	79
5.2.1	Detailed Description . . . . .	79
5.2.2	Function Documentation . . . . .	79
5.2.2.1	createTrainingMeans . . . . .	79
5.2.2.2	exportCenters . . . . .	80
5.2.2.3	importCenters . . . . .	80
5.2.2.4	importSTIPs . . . . .	80
5.2.2.5	kmIvanAlgorithm . . . . .	81

5.3	naokmeans.h File Reference	82
5.3.1	Detailed Description	82
5.3.2	Function Documentation	82
5.3.2.1	createTrainingMeans	82
5.3.2.2	exportCenters	83
5.3.2.3	importCenters	83
5.3.2.4	importSTIPs	83
5.3.2.5	kmIvanAlgorithm	84
5.4	naomngt.cpp File Reference	85
5.4.1	Detailed Description	86
5.4.2	Function Documentation	86
5.4.2.1	addActivity	86
5.4.2.2	addBdd	86
5.4.2.3	addLabel	86
5.4.2.4	addVideos	87
5.4.2.5	deleteActivity	87
5.4.2.6	deleteBdd	87
5.4.2.7	emptyFolder	87
5.4.2.8	fileExist	88
5.4.2.9	inttostring	88
5.4.2.10	listActivities	88
5.4.2.11	mapActivities	88
5.4.2.12	nbOfFiles	89
5.4.2.13	trainBdd	89
5.5	naomngt.h File Reference	90
5.5.1	Detailed Description	91
5.5.2	Function Documentation	91
5.5.2.1	addActivity	91
5.5.2.2	addBdd	92
5.5.2.3	addLabel	92
5.5.2.4	addVideos	92
5.5.2.5	deleteActivity	92
5.5.2.6	deleteBdd	93
5.5.2.7	emptyFolder	93
5.5.2.8	fileExist	93
5.5.2.9	inttostring	93

---

5.5.2.10	<a href="#">listActivities</a>	94
5.5.2.11	<a href="#">mapActivities</a>	94
5.5.2.12	<a href="#">nbOfFiles</a>	94
5.5.2.13	<a href="#">trainBdd</a>	94
5.6	<a href="#">naosvm.cpp File Reference</a>	95
5.6.1	<a href="#">Detailed Description</a>	95
5.6.2	<a href="#">Function Documentation</a>	95
5.6.2.1	<a href="#">nrOfLines</a>	95
5.6.2.2	<a href="#">printProbability</a>	96
5.6.2.3	<a href="#">printProblem</a>	96
5.7	<a href="#">naosvm.h File Reference</a>	97
5.7.1	<a href="#">Detailed Description</a>	97
5.7.2	<a href="#">Function Documentation</a>	97
5.7.2.1	<a href="#">nrOfLines</a>	97
5.7.2.2	<a href="#">printProbability</a>	98
5.7.2.3	<a href="#">printProblem</a>	98



# Chapter 1

## Class Index

### 1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

activitiesMap . . . . .	9
Box< T > . . . . .	10
Cache . . . . .	20
decision_function . . . . .	21
DescInfo . . . . .	22
DescMat . . . . .	23
exec_time . . . . .	24
IplImagePyramid . . . . .	25
IplImageWrapper . . . . .	31
KCnode . . . . .	35
KCleaf . . . . .	34
KCleaf . . . . .	34
KCsplit . . . . .	37
KCsplit . . . . .	37
KCtree . . . . .	38
KMcenters . . . . .	40
KMfilterCenters . . . . .	43
KMfilterCenters . . . . .	43
KMdata . . . . .	42
KMlocal . . . . .	45
KMlocalEZ_Hybrid . . . . .	47
KMlocalEZ_Hybrid . . . . .	47
KMlocalHybrid . . . . .	48
KMlocalHybrid . . . . .	48
KMlocalLloyds . . . . .	50
KMlocalLloyds . . . . .	50
KMlocalSwap . . . . .	51
KMlocalSwap . . . . .	51
KMorthRect . . . . .	52
KMterm . . . . .	53
Point< T > . . . . .	56
PointDesc . . . . .	58

---

QMatrix . . . . .	59
Kernel . . . . .	39
ONE_CLASS_Q . . . . .	55
SVC_Q . . . . .	67
SVR_Q . . . . .	72
Size< T > . . . . .	60
Solver::SolutionInfo . . . . .	63
Solver . . . . .	64
Solver_NU . . . . .	66
svm_model . . . . .	68
svm_node . . . . .	69
svm_parameter . . . . .	70
svm_problem . . . . .	71
Tactil . . . . .	73
temps_exec . . . . .	74
Track . . . . .	75
TrackerInfo . . . . .	76

# Chapter 2

## Class Index

### 2.1 Class List

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

<a href="#">activitiesMap</a> (Correspondance between a label and an activity )	9
<a href="#">Box&lt; T &gt;</a>	10
<a href="#">Cache</a>	20
<a href="#">decision_function</a>	21
<a href="#">DescInfo</a>	22
<a href="#">DescMat</a>	23
<a href="#">exec_time</a>	24
<a href="#">IplImagePyramid</a>	25
<a href="#">IplImageWrapper</a>	31
<a href="#">KCleaf</a>	34
<a href="#">KNode</a>	35
<a href="#">KCsplit</a>	37
<a href="#">KCTree</a>	38
<a href="#">Kernel</a>	39
<a href="#">KMcenters</a>	40
<a href="#">KMdata</a>	42
<a href="#">KMfilterCenters</a>	43
<a href="#">KMlocal</a>	45
<a href="#">KMlocalEZ_Hybrid</a>	47
<a href="#">KMlocalHybrid</a>	48
<a href="#">KMlocalLloyds</a>	50
<a href="#">KMlocalSwap</a>	51
<a href="#">KMorthRect</a>	52
<a href="#">KMterm</a>	53
<a href="#">ONE_CLASS_Q</a>	55
<a href="#">Point&lt; T &gt;</a>	56
<a href="#">PointDesc</a>	58
<a href="#">QMatrix</a>	59
<a href="#">Size&lt; T &gt;</a>	60
<a href="#">Solver::SolutionInfo</a>	63
<a href="#">Solver</a>	64
<a href="#">Solver_NU</a>	66
<a href="#">SVC_Q</a>	67

<a href="#">svm_model</a> . . . . .	68
<a href="#">svm_node</a> . . . . .	69
<a href="#">svm_parameter</a> . . . . .	70
<a href="#">svm_problem</a> . . . . .	71
<a href="#">SVR_Q</a> . . . . .	72
<a href="#">Tactil</a> . . . . .	73
<a href="#">temps_exec</a> . . . . .	74
<a href="#">Track</a> . . . . .	75
<a href="#">TrackerInfo</a> . . . . .	76

# Chapter 3

## File Index

### 3.1 File List

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

<b>include/Box.h</b>	??
<b>include/densetrack/Box.h</b>	??
<b>integration/densetrack/Box.h</b>	??
<b>lib/densetrack/Box.h</b>	??
<b>modules/densetrack/Box.h</b>	??
<b>denseTrack.cpp</b>	??
<b>DenseTrack.cpp</b>	??
<b>DenseTrack.h</b>	??
<b>include/denseTrack.h</b>	??
<b>modules/densetrack/denseTrack.h</b>	??
<b>include/densetrack/functions.h</b>	??
<b>include/functions.h</b>	??
<b>integration/densetrack/functions.h</b>	??
<b>lib/densetrack/functions.h</b>	??
<b>modules/densetrack/functions.h</b>	??
<b>integration.cpp</b>	??
<b>integration.h</b>	??
<b>lib/densetrack/IplImagePyramid.cpp</b>	??
<b>modules/densetrack/IplImagePyramid.cpp</b>	??
<b>src/IplImagePyramid.cpp</b>	??
<b>include/densetrack/IplImagePyramid.h</b>	??
<b>include/IplImagePyramid.h</b>	??
<b>integration/densetrack/IplImagePyramid.h</b>	??
<b>lib/densetrack/IplImagePyramid.h</b>	??
<b>modules/densetrack/IplImagePyramid.h</b>	??
<b>lib/densetrack/IplImageWrapper.cpp</b>	??
<b>modules/densetrack/IplImageWrapper.cpp</b>	??
<b>src/IplImageWrapper.cpp</b>	??
<b>include/densetrack/IplImageWrapper.h</b>	??
<b>include/IplImageWrapper.h</b>	??
<b>integration/densetrack/IplImageWrapper.h</b>	??
<b>lib/densetrack/IplImageWrapper.h</b>	??
<b>modules/densetrack/IplImageWrapper.h</b>	??

KCtree.cpp	??
include/kmlocal/KCtree.h	??
lib/kmlocal-1.7.2/src/KCtree.h	??
KCutil.cpp	??
include/kmlocal/KCutil.h	??
lib/kmlocal-1.7.2/src/KCutil.h	??
KM_ANN.cpp	??
include/kmlocal/KM_ANN.h	??
lib/kmlocal-1.7.2/src/KM_ANN.h	??
KMcenters.cpp	??
include/kmlocal/KMcenters.h	??
lib/kmlocal-1.7.2/src/KMcenters.h	??
KMdata.cpp	??
include/kmlocal/KMdata.h	??
lib/kmlocal-1.7.2/src/KMdata.h	??
KMeans.cpp	??
include/kmlocal/KMeans.h	??
lib/kmlocal-1.7.2/src/KMeans.h	??
KMfilterCenters.cpp	??
include/kmlocal/KMfilterCenters.h	??
lib/kmlocal-1.7.2/src/KMfilterCenters.h	??
kmlminimal.cpp	??
KMlocal.cpp	??
include/kmlocal/KMlocal.h	??
lib/kmlocal-1.7.2/src/KMlocal.h	??
kmlsample.cpp	??
kmltest.cpp	??
KMrand.cpp	??
include/kmlocal/KMrand.h	??
lib/kmlocal-1.7.2/src/KMrand.h	??
KMterm.cpp	??
include/kmlocal/KMterm.h	??
lib/kmlocal-1.7.2/src/KMterm.h	??
integration/main.c	??
outils/svmTrain/main.c	??
actmngt/main.cpp	??
modules/argvToSpeech/main.cpp	??
modules/densetrack/main.cpp	??
modules/helloName/main.cpp	??
modules/integration/main.cpp	??
modules/libintegration/main.cpp	??
modules/recordVideo/main.cpp	??
modules/startEvent/main.cpp	??
outils/bow2bow0/main.cpp	??
outils/densetrack/main.cpp	??
outils/fpsConvert/main.cpp	??
outils/toGrayScale/main.cpp	??
tests/main.cpp	??
motion_stiffnessOff.py	??
motion_stiffnessOn.py	??
naodensetrack.cpp (Set of function permitting to extract dense points and their trajectories )	77
naodensetrack.h	??
naokmeans.cpp (Set of functions permitting to execute KMeans algorithms using <a href="#">KMlocal</a> classes )	79
naokmeans.h	82

<a href="#">naomngt.cpp</a> (Set of functions permitting to manage the activity recognition BDD of Bag Of Words ) . . . . .	85
<a href="#">naomngt.h</a> . . . . .	90
<a href="#">naosvm.cpp</a> (Set of functions permitting to import/ predict a svm problem, import/create a svm model ) . . . . .	95
<a href="#">naosvm.h</a> . . . . .	97
<b>numericFunctions.cpp</b> . . . . .	??
<b>include/densetrack/numericFunctions.h</b> . . . . .	??
<b>include/numericFunctions.h</b> . . . . .	??
<b>integration/densetrack/numericFunctions.h</b> . . . . .	??
<b>lib/densetrack/numericFunctions.h</b> . . . . .	??
<b>modules/densetrack/numericFunctions.h</b> . . . . .	??
<b>include/densetrack/Point.h</b> . . . . .	??
<b>include/Point.h</b> . . . . .	??
<b>integration/densetrack/Point.h</b> . . . . .	??
<b>lib/densetrack/Point.h</b> . . . . .	??
<b>modules/densetrack/Point.h</b> . . . . .	??
<b>include/densetrack/Size.h</b> . . . . .	??
<b>include/Size.h</b> . . . . .	??
<b>integration/densetrack/Size.h</b> . . . . .	??
<b>lib/densetrack/Size.h</b> . . . . .	??
<b>modules/densetrack/Size.h</b> . . . . .	??
<b>svm.cpp</b> . . . . .	??
<b>include/svm.h</b> . . . . .	??
<b>lib/libsvm-3.17/svm.h</b> . . . . .	??
<b>modules/libintegration/tactil.cpp</b> . . . . .	??
<b>src/tactil.cpp</b> . . . . .	??
<b>tactil.h</b> . . . . .	??
<b>modules/argvToSpeech/test.cpp</b> . . . . .	??
<b>modules/densetrack/test.cpp</b> . . . . .	??
<b>modules/helloName/test.cpp</b> . . . . .	??
<b>modules/recordVideo/test.cpp</b> . . . . .	??
<b>modules/startEvent/test.cpp</b> . . . . .	??
<b>src/test.cpp</b> . . . . .	??





## Chapter 4

# Class Documentation

### 4.1 activitiesMap Class Reference

Correspondance between a label and an activity.

```
#include <naomngt.h>
```

#### Public Attributes

- `int` **label**
- `std::string` **activity**

#### 4.1.1 Detailed Description

Correspondance between a label and an activity.

Definition at line 28 of file naomngt.h.

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

- [naomngt.h](#)

## 4.2 **Box< T > Class Template Reference**

### Public Types

- typedef T **ValueType**
- typedef T **ValueType**
- typedef T **ValueType**
- typedef T **ValueType**
- typedef T **ValueType**

### Public Member Functions

- **Box** (const [Point](#)< T > &topLeft, const [Size](#)< T > &size)
- **Box** (T left, T top, T width, T height)
- bool **isEmpty** () const
- bool **isValid** () const
- [Box](#) **normalized** () const
- T **getLeft** () const
- T **getTop** () const
- T **getRight** () const
- T **getBottom** () const
- [Size](#)< T > **getSize** () const
- T **getX** () const
- T **getY** () const
- void **setLeft** (T pos)
- void **setTop** (T pos)
- void **setRight** (T pos)
- void **setBottom** (T pos)
- void **setX** (T pos)
- void **setY** (T pos)
- [Point](#)< T > **getTopLeft** () const
- [Point](#)< T > **getBottomRight** () const
- [Point](#)< T > **getTopRight** () const
- [Point](#)< T > **getBottomLeft** () const
- [Point](#)< T > **getCenter** () const
- void **setTopLeft** (const [Point](#)< T > &p)
- void **setBottomRight** (const [Point](#)< T > &p)
- void **setTopRight** (const [Point](#)< T > &p)
- void **setBottomLeft** (const [Point](#)< T > &p)
- void **moveLeft** (T pos)
- void **moveTop** (T pos)
- void **moveRight** (T pos)
- void **moveBottom** (T pos)
- void **moveTopLeft** (const [Point](#)< T > &p)
- void **moveBottomRight** (const [Point](#)< T > &p)
- void **moveTopRight** (const [Point](#)< T > &p)
- void **moveBottomLeft** (const [Point](#)< T > &p)
- void **moveCenter** (const [Point](#)< T > &p)
- void **translate** (T dx, T dy)
- void **translate** (const [Point](#)< T > &p)

- **Box translated** (T dx, T dy) const
- **Box translated** (const [Point](#)< T > &p) const
- void **moveTo** (T x, T t)
- void **moveTo** (const [Point](#)< T > &p)
- void **setBox** (T x, T y, T w, T h)
- void **getBox** (T \*x=0, T \*y=0, T \*w=0, T \*h=0) const
- void **setCoords** (T x1, T y1, T x2, T y2)
- void **getCoords** (T \*x1=0, T \*y1=0, T \*x2=0, T \*y2=0) const
- template<typename T2 >  
void **scale** (T2 xScale, T2 yScale)
- T **getWidth** () const
- T **getHeight** () const
- void **setWidth** (T w)
- void **setHeight** (T h)
- void **setSize** (const [Size](#)< T > &s)
- T **getArea** () const
- [Box](#)< T > **operator|** (const [Box](#) &r) const
- [Box](#)< T > **operator&** (const [Box](#) &r) const
- [Box](#)< T > & **operator|=** (const [Box](#) &r)
- [Box](#)< T > & **operator&=** (const [Box](#) &r)
- template<typename T2 >  
[Box](#)< T > & **operator\*=** (T2 c)
- template<typename T2 >  
[Box](#)< T > & **operator/=** (T2 c)
- template<typename T2 >  
bool **contains** (const [Point](#)< T2 > &p) const
- template<typename T2 >  
bool **contains** (T2 x, T2 y) const
- bool **contains** (const [Box](#)< T > &r) const
- [Box](#)< T > **unite** (const [Box](#)< T > &r) const
- [Box](#)< T > **intersect** (const [Box](#)< T > &r) const
- bool **intersects** (const [Box](#)< T > &r) const
- template<typename T2 >  
**operator Box**< T2 > () const
- **Box** (const [Point](#)< T > &topLeft, const [Size](#)< T > &size)
- **Box** (T left, T top, T width, T height)
- bool **isEmpty** () const
- bool **isValid** () const
- [Box](#) **normalized** () const
- T **getLeft** () const
- T **getTop** () const
- T **getRight** () const
- T **getBottom** () const
- [Size](#)< T > **getSize** () const
- T **getX** () const
- T **getY** () const
- void **setLeft** (T pos)
- void **setTop** (T pos)
- void **setRight** (T pos)
- void **setBottom** (T pos)

- void **setX** (T pos)
- void **setY** (T pos)
- **Point**< T > **getTopLeft** () const
- **Point**< T > **getBottomRight** () const
- **Point**< T > **getTopRight** () const
- **Point**< T > **getBottomLeft** () const
- **Point**< T > **getCenter** () const
- void **setTopLeft** (const **Point**< T > &p)
- void **setBottomRight** (const **Point**< T > &p)
- void **setTopRight** (const **Point**< T > &p)
- void **setBottomLeft** (const **Point**< T > &p)
- void **moveLeft** (T pos)
- void **moveTop** (T pos)
- void **moveRight** (T pos)
- void **moveBottom** (T pos)
- void **moveTopLeft** (const **Point**< T > &p)
- void **moveBottomRight** (const **Point**< T > &p)
- void **moveTopRight** (const **Point**< T > &p)
- void **moveBottomLeft** (const **Point**< T > &p)
- void **moveCenter** (const **Point**< T > &p)
- void **translate** (T dx, T dy)
- void **translate** (const **Point**< T > &p)
- **Box** **translated** (T dx, T dy) const
- **Box** **translated** (const **Point**< T > &p) const
- void **moveTo** (T x, T t)
- void **moveTo** (const **Point**< T > &p)
- void **setBox** (T x, T y, T w, T h)
- void **getBox** (T \*x=0, T \*y=0, T \*w=0, T \*h=0) const
- void **setCoords** (T x1, T y1, T x2, T y2)
- void **getCoords** (T \*x1=0, T \*y1=0, T \*x2=0, T \*y2=0) const
- template<typename T2 >  
void **scale** (T2 xScale, T2 yScale)
- T **getWidth** () const
- T **getHeight** () const
- void **setWidth** (T w)
- void **setHeight** (T h)
- void **setSize** (const **Size**< T > &s)
- T **getArea** () const
- **Box**< T > **operator|** (const **Box** &r) const
- **Box**< T > **operator&** (const **Box** &r) const
- **Box**< T > **& operator|=** (const **Box** &r)
- **Box**< T > **& operator&=** (const **Box** &r)
- template<typename T2 >  
**Box**< T > **& operator\*=** (T2 c)
- template<typename T2 >  
**Box**< T > **& operator/=** (T2 c)
- template<typename T2 >  
bool **contains** (const **Point**< T2 > &p) const
- template<typename T2 >  
bool **contains** (T2 x, T2 y) const

- bool **contains** (const Box< T > &r) const
- Box< T > **unite** (const Box< T > &r) const
- Box< T > **intersect** (const Box< T > &r) const
- bool **intersects** (const Box< T > &r) const
- template<typename T2 >  
  **operator Box< T2 > ()** const
- **Box** (const Point< T > &topLeft, const Size< T > &size)
- **Box** (T left, T top, T width, T height)
- bool **isEmpty** () const
- bool **isValid** () const
- **Box normalized** () const
- T **getLeft** () const
- T **getTop** () const
- T **getRight** () const
- T **getBottom** () const
- Size< T > **getSize** () const
- T **getX** () const
- T **getY** () const
- void **setLeft** (T pos)
- void **setTop** (T pos)
- void **setRight** (T pos)
- void **setBottom** (T pos)
- void **setX** (T pos)
- void **setY** (T pos)
- Point< T > **getTopLeft** () const
- Point< T > **getBottomRight** () const
- Point< T > **getTopRight** () const
- Point< T > **getBottomLeft** () const
- Point< T > **getCenter** () const
- void **setTopLeft** (const Point< T > &p)
- void **setBottomRight** (const Point< T > &p)
- void **setTopRight** (const Point< T > &p)
- void **setBottomLeft** (const Point< T > &p)
- void **moveLeft** (T pos)
- void **moveTop** (T pos)
- void **moveRight** (T pos)
- void **moveBottom** (T pos)
- void **moveTopLeft** (const Point< T > &p)
- void **moveBottomRight** (const Point< T > &p)
- void **moveTopRight** (const Point< T > &p)
- void **moveBottomLeft** (const Point< T > &p)
- void **moveCenter** (const Point< T > &p)
- void **translate** (T dx, T dy)
- void **translate** (const Point< T > &p)
- **Box translated** (T dx, T dy) const
- **Box translated** (const Point< T > &p) const
- void **moveTo** (T x, T t)
- void **moveTo** (const Point< T > &p)
- void **setBox** (T x, T y, T w, T h)
- void **getBox** (T \*x=0, T \*y=0, T \*w=0, T \*h=0) const

- void **setCoords** (T x1, T y1, T x2, T y2)
- void **getCoords** (T \*x1=0, T \*y1=0, T \*x2=0, T \*y2=0) const
- template<typename T2 >  
void **scale** (T2 xScale, T2 yScale)
- T **getWidth** () const
- T **getHeight** () const
- void **setWidth** (T w)
- void **setHeight** (T h)
- void **setSize** (const [Size](#)< T > &s)
- T **getArea** () const
- [Box](#)< T > **operator|** (const [Box](#) &r) const
- [Box](#)< T > **operator&** (const [Box](#) &r) const
- [Box](#)< T > & **operator|=** (const [Box](#) &r)
- [Box](#)< T > & **operator&=** (const [Box](#) &r)
- template<typename T2 >  
[Box](#)< T > & **operator\*=** (T2 c)
- template<typename T2 >  
[Box](#)< T > & **operator/=** (T2 c)
- template<typename T2 >  
bool **contains** (const [Point](#)< T2 > &p) const
- template<typename T2 >  
bool **contains** (T2 x, T2 y) const
- bool **contains** (const [Box](#)< T > &r) const
- [Box](#)< T > **unite** (const [Box](#)< T > &r) const
- [Box](#)< T > **intersect** (const [Box](#)< T > &r) const
- bool **intersects** (const [Box](#)< T > &r) const
- template<typename T2 >  
**operator Box**< T2 > () const
- [Box](#) (const [Point](#)< T > &topLeft, const [Size](#)< T > &size)
- [Box](#) (T left, T top, T width, T height)
- bool **isEmpty** () const
- bool **isValid** () const
- [Box](#) **normalized** () const
- T **getLeft** () const
- T **getTop** () const
- T **getRight** () const
- T **getBottom** () const
- [Size](#)< T > **getSize** () const
- T **getX** () const
- T **getY** () const
- void **setLeft** (T pos)
- void **setTop** (T pos)
- void **setRight** (T pos)
- void **setBottom** (T pos)
- void **setX** (T pos)
- void **setY** (T pos)
- [Point](#)< T > **getTopLeft** () const
- [Point](#)< T > **getBottomRight** () const
- [Point](#)< T > **getTopRight** () const
- [Point](#)< T > **getBottomLeft** () const

- **Point**< T > **getCenter** () const
- void **setTopLeft** (const **Point**< T > &p)
- void **setBottomRight** (const **Point**< T > &p)
- void **setTopRight** (const **Point**< T > &p)
- void **setBottomLeft** (const **Point**< T > &p)
- void **moveLeft** (T pos)
- void **moveTop** (T pos)
- void **moveRight** (T pos)
- void **moveBottom** (T pos)
- void **moveTopLeft** (const **Point**< T > &p)
- void **moveBottomRight** (const **Point**< T > &p)
- void **moveTopRight** (const **Point**< T > &p)
- void **moveBottomLeft** (const **Point**< T > &p)
- void **moveCenter** (const **Point**< T > &p)
- void **translate** (T dx, T dy)
- void **translate** (const **Point**< T > &p)
- **Box** **translated** (T dx, T dy) const
- **Box** **translated** (const **Point**< T > &p) const
- void **moveTo** (T x, T t)
- void **moveTo** (const **Point**< T > &p)
- void **setBox** (T x, T y, T w, T h)
- void **getBox** (T \*x=0, T \*y=0, T \*w=0, T \*h=0) const
- void **setCoords** (T x1, T y1, T x2, T y2)
- void **getCoords** (T \*x1=0, T \*y1=0, T \*x2=0, T \*y2=0) const
- template<typename T2 >  
void **scale** (T2 xScale, T2 yScale)
- T **getWidth** () const
- T **getHeight** () const
- void **setWidth** (T w)
- void **setHeight** (T h)
- void **setSize** (const **Size**< T > &s)
- T **getArea** () const
- **Box**< T > **operator|** (const **Box** &r) const
- **Box**< T > **operator&** (const **Box** &r) const
- **Box**< T > & **operator|=** (const **Box** &r)
- **Box**< T > & **operator&=** (const **Box** &r)
- template<typename T2 >  
**Box**< T > & **operator\*=** (T2 c)
- template<typename T2 >  
**Box**< T > & **operator/=** (T2 c)
- template<typename T2 >  
bool **contains** (const **Point**< T2 > &p) const
- template<typename T2 >  
bool **contains** (T2 x, T2 y) const
- bool **contains** (const **Box**< T > &r) const
- **Box**< T > **unite** (const **Box**< T > &r) const
- **Box**< T > **intersect** (const **Box**< T > &r) const
- bool **intersects** (const **Box**< T > &r) const
- template<typename T2 >  
**operator Box**< T2 > () const

- **Box** (const [Point](#)< T > &topLeft, const [Size](#)< T > &size)
- **Box** (T left, T top, T width, T height)
- bool **isEmpty** () const
- bool **isValid** () const
- [Box](#) **normalized** () const
- T **getLeft** () const
- T **getTop** () const
- T **getRight** () const
- T **getBottom** () const
- [Size](#)< T > **getSize** () const
- T **getX** () const
- T **getY** () const
- void **setLeft** (T pos)
- void **setTop** (T pos)
- void **setRight** (T pos)
- void **setBottom** (T pos)
- void **setX** (T pos)
- void **setY** (T pos)
- [Point](#)< T > **getTopLeft** () const
- [Point](#)< T > **getBottomRight** () const
- [Point](#)< T > **getTopRight** () const
- [Point](#)< T > **getBottomLeft** () const
- [Point](#)< T > **getCenter** () const
- void **setTopLeft** (const [Point](#)< T > &p)
- void **setBottomRight** (const [Point](#)< T > &p)
- void **setTopRight** (const [Point](#)< T > &p)
- void **setBottomLeft** (const [Point](#)< T > &p)
- void **moveLeft** (T pos)
- void **moveTop** (T pos)
- void **moveRight** (T pos)
- void **moveBottom** (T pos)
- void **moveTopLeft** (const [Point](#)< T > &p)
- void **moveBottomRight** (const [Point](#)< T > &p)
- void **moveTopRight** (const [Point](#)< T > &p)
- void **moveBottomLeft** (const [Point](#)< T > &p)
- void **moveCenter** (const [Point](#)< T > &p)
- void **translate** (T dx, T dy)
- void **translate** (const [Point](#)< T > &p)
- [Box](#) **translated** (T dx, T dy) const
- [Box](#) **translated** (const [Point](#)< T > &p) const
- void **moveTo** (T x, T t)
- void **moveTo** (const [Point](#)< T > &p)
- void **setBox** (T x, T y, T w, T h)
- void **getBox** (T \*x=0, T \*y=0, T \*w=0, T \*h=0) const
- void **setCoords** (T x1, T y1, T x2, T y2)
- void **getCoords** (T \*x1=0, T \*y1=0, T \*x2=0, T \*y2=0) const
- template<typename T2 >  
void **scale** (T2 xScale, T2 yScale)
- T **getWidth** () const
- T **getHeight** () const



- void **setWidth** (T w)
- void **setHeight** (T h)
- void **setSize** (const [Size](#)< T > &s)
- T **getArea** () const
- [Box](#)< T > **operator|** (const [Box](#) &r) const
- [Box](#)< T > **operator&** (const [Box](#) &r) const
- [Box](#)< T > & **operator|=** (const [Box](#) &r)
- [Box](#)< T > & **operator&=** (const [Box](#) &r)
- template<typename T2 >  
  [Box](#)< T > & **operator\*=** (T2 c)
- template<typename T2 >  
  [Box](#)< T > & **operator/=** (T2 c)
- template<typename T2 >  
  bool **contains** (const [Point](#)< T2 > &p) const
- template<typename T2 >  
  bool **contains** (T2 x, T2 y) const
- bool **contains** (const [Box](#)< T > &r) const
- [Box](#)< T > **unite** (const [Box](#)< T > &r) const
- [Box](#)< T > **intersect** (const [Box](#)< T > &r) const
- bool **intersects** (const [Box](#)< T > &r) const
- template<typename T2 >  
  **operator Box**< T2 > () const
- template<>  
  void **scale** (float xScale, float yScale)
- template<>  
  void **scale** (float xScale, float yScale)
- template<>  
  void **scale** (double xScale, double yScale)
- template<>  
  void **scale** (double xScale, double yScale)
- template<>  
  void **scale** (float xScale, float yScale)
- template<>  
  void **scale** (float xScale, float yScale)
- template<>  
  void **scale** (double xScale, double yScale)
- template<>  
  void **scale** (double xScale, double yScale)
- template<>  
  void **scale** (float xScale, float yScale)
- template<>  
  void **scale** (float xScale, float yScale)
- template<>  
  void **scale** (double xScale, double yScale)
- template<>  
  void **scale** (double xScale, double yScale)
- template<>  
  void **scale** (float xScale, float yScale)
- template<>  
  void **scale** (float xScale, float yScale)
- template<>  
  void **scale** (double xScale, double yScale)

- `template<>`  
`void scale (double xScale, double yScale)`
- `template<>`  
`void scale (float xScale, float yScale)`
- `template<>`  
`void scale (float xScale, float yScale)`
- `template<>`  
`void scale (double xScale, double yScale)`
- `template<>`  
`void scale (double xScale, double yScale)`

## Friends

- `template<typename T2 >`  
`bool operator== (const Box< T2 > &, const Box< T2 > &)`
- `template<typename T2 >`  
`bool operator!= (const Box< T2 > &, const Box< T2 > &)`
- `template<typename T2 , typename T3 >`  
`Box< T2 > operator* (const Box< T2 > &, T3 c)`
- `template<typename T2 , typename T3 >`  
`Box< T2 > operator* (T3 c, const Box< T2 > &)`
- `template<typename T2 , typename T3 >`  
`Box< T2 > operator/ (const Box< T2 > &, T3 c)`
- `template<typename T2 >`  
`bool operator== (const Box< T2 > &, const Box< T2 > &)`
- `template<typename T2 >`  
`bool operator!= (const Box< T2 > &, const Box< T2 > &)`
- `template<typename T2 , typename T3 >`  
`Box< T2 > operator* (const Box< T2 > &, T3 c)`
- `template<typename T2 , typename T3 >`  
`Box< T2 > operator* (T3 c, const Box< T2 > &)`
- `template<typename T2 , typename T3 >`  
`Box< T2 > operator/ (const Box< T2 > &, T3 c)`
- `template<typename T2 >`  
`bool operator== (const Box< T2 > &, const Box< T2 > &)`
- `template<typename T2 >`  
`bool operator!= (const Box< T2 > &, const Box< T2 > &)`
- `template<typename T2 , typename T3 >`  
`Box< T2 > operator* (const Box< T2 > &, T3 c)`
- `template<typename T2 , typename T3 >`  
`Box< T2 > operator* (T3 c, const Box< T2 > &)`
- `template<typename T2 , typename T3 >`  
`Box< T2 > operator/ (const Box< T2 > &, T3 c)`
- `template<typename T2 >`  
`bool operator== (const Box< T2 > &, const Box< T2 > &)`
- `template<typename T2 >`  
`bool operator!= (const Box< T2 > &, const Box< T2 > &)`
- `template<typename T2 , typename T3 >`  
`Box< T2 > operator* (const Box< T2 > &, T3 c)`
- `template<typename T2 , typename T3 >`  
`Box< T2 > operator* (T3 c, const Box< T2 > &)`

- template<typename T2 , typename T3 >  
Box< T2 > **operator/** (const Box< T2 > &, T3 c)
- template<typename T2 >  
bool **operator==** (const Box< T2 > &, const Box< T2 > &)
- template<typename T2 >  
bool **operator!=** (const Box< T2 > &, const Box< T2 > &)
- template<typename T2 , typename T3 >  
Box< T2 > **operator\*** (const Box< T2 > &, T3 c)
- template<typename T2 , typename T3 >  
Box< T2 > **operator\*** (T3 c, const Box< T2 > &)
- template<typename T2 , typename T3 >  
Box< T2 > **operator/** (const Box< T2 > &, T3 c)

### 4.2.1 Detailed Description

**template<typename T = double> class Box< T >**

Definition at line 12 of file include/Box.h.

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

- include/Box.h
- include/densetrack/Box.h
- integration/densetrack/Box.h
- lib/densetrack/Box.h
- modules/densetrack/Box.h

## 4.3 Cache Class Reference

### Classes

- struct **head\_t**

### Public Member Functions

- **Cache** (int l, long int size)
- int **get\_data** (const int index, Qfloat \*\*data, int len)
- void **swap\_index** (int i, int j)

#### 4.3.1 Detailed Description

Definition at line 67 of file svm.cpp.

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

- svm.cpp

## 4.4 decision\_function Struct Reference

### Public Attributes

- double \* **alpha**
- double **rho**

### 4.4.1 Detailed Description

Definition at line 1641 of file svm.cpp.

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

- svm.cpp

## 4.5 DescInfo Struct Reference

### Public Attributes

- int **nBins**
- int **fullOrientation**
- int **norm**
- float **threshold**
- int **flagThre**
- int **nxCells**
- int **nyCells**
- int **ntCells**
- int **dim**
- int **blockHeight**
- int **blockWidth**

### 4.5.1 Detailed Description

Definition at line 30 of file include/denseTrack.h.

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

- include/denseTrack.h
- naodensetrack.h
- DenseTrack.h
- modules/densetrack/denseTrack.h

## 4.6 DescMat Struct Reference

### Public Attributes

- int **height**
- int **width**
- int **nBins**
- float \* **desc**

### 4.6.1 Detailed Description

Definition at line 45 of file include/denseTrack.h.

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

- include/denseTrack.h
- naodensetrack.h
- DenseTrack.h
- modules/densetrack/denseTrack.h

## 4.7 exec\_time Struct Reference

### Public Attributes

- int **begin**
- int **end**
- sTms **sbegin**
- sTms **send**

### 4.7.1 Detailed Description

Definition at line 15 of file integration.h.

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

- integration.h



## 4.8 IplImagePyramid Class Reference

### Public Member Functions

- **IplImagePyramid** (const [IplImagePyramid](#) &pyramid)
- **IplImagePyramid** ([IplImageWrapper](#) image, double scaleFactor)
- **IplImagePyramid** (CvSize initSize, int depth, int nChannels, double scaleFactor)
- **IplImagePyramid** & **operator=** (const [IplImagePyramid](#) &pyramid)
- **operator const bool** () const
- **operator bool** ()
- std::size\_t **numOfLevels** () const
- double **getScaleFactor** () const
- std::size\_t **getIndex** (double scaleFactor, int round=0) const
- double **getScaleFactor** (std::size\_t index) const
- double **getScaleFactorInv** (std::size\_t index) const
- double **getXScaleFactor** (std::size\_t index) const
- double **getXScaleFactorInv** (std::size\_t index) const
- double **getYScaleFactor** (std::size\_t index) const
- double **getYScaleFactorInv** (std::size\_t index) const
- [IplImageWrapper](#) & **getImage** (std::size\_t index)
- const [IplImageWrapper](#) & **getImage** (std::size\_t index) const
- [IplImageWrapper](#) **getImage** (double scaleFactor, int round=0)
- const [IplImageWrapper](#) & **getImage** (double scaleFactor, int round=0) const
- void **rebuild** ([IplImageWrapper](#) image)
- **IplImagePyramid** (const [IplImagePyramid](#) &pyramid)
- **IplImagePyramid** ([IplImageWrapper](#) image, double scaleFactor)
- **IplImagePyramid** (CvSize initSize, int depth, int nChannels, double scaleFactor)
- **IplImagePyramid** & **operator=** (const [IplImagePyramid](#) &pyramid)
- **operator const bool** () const
- **operator bool** ()
- std::size\_t **numOfLevels** () const
- double **getScaleFactor** () const
- std::size\_t **getIndex** (double scaleFactor, int round=0) const
- double **getScaleFactor** (std::size\_t index) const
- double **getScaleFactorInv** (std::size\_t index) const
- double **getXScaleFactor** (std::size\_t index) const
- double **getXScaleFactorInv** (std::size\_t index) const
- double **getYScaleFactor** (std::size\_t index) const
- double **getYScaleFactorInv** (std::size\_t index) const
- [IplImageWrapper](#) & **getImage** (std::size\_t index)
- const [IplImageWrapper](#) & **getImage** (std::size\_t index) const
- [IplImageWrapper](#) **getImage** (double scaleFactor, int round=0)
- const [IplImageWrapper](#) & **getImage** (double scaleFactor, int round=0) const
- void **rebuild** ([IplImageWrapper](#) image)
- **IplImagePyramid** (const [IplImagePyramid](#) &pyramid)
- **IplImagePyramid** ([IplImageWrapper](#) image, double scaleFactor)
- **IplImagePyramid** (CvSize initSize, int depth, int nChannels, double scaleFactor)
- **IplImagePyramid** & **operator=** (const [IplImagePyramid](#) &pyramid)
- **operator const bool** () const
- **operator bool** ()

- `std::size_t numOfLevels ()` const
- `double getScaleFactor ()` const
- `std::size_t getIndex (double scaleFactor, int round=0)` const
- `double getScaleFactor (std::size_t index)` const
- `double getScaleFactorInv (std::size_t index)` const
- `double getXScaleFactor (std::size_t index)` const
- `double getXScaleFactorInv (std::size_t index)` const
- `double getYScaleFactor (std::size_t index)` const
- `double getYScaleFactorInv (std::size_t index)` const
- `IplImageWrapper & getImage (std::size_t index)`
- `const IplImageWrapper & getImage (std::size_t index)` const
- `IplImageWrapper getImage (double scaleFactor, int round=0)`
- `const IplImageWrapper & getImage (double scaleFactor, int round=0)` const
- `void rebuild (IplImageWrapper image)`
- `IplImagePyramid (const IplImagePyramid &pyramid)`
- `IplImagePyramid (IplImageWrapper image, double scaleFactor)`
- `IplImagePyramid (CvSize initSize, int depth, int nChannels, double scaleFactor)`
- `IplImagePyramid & operator= (const IplImagePyramid &pyramid)`
- `operator const bool ()` const
- `operator bool ()`
- `std::size_t numOfLevels ()` const
- `double getScaleFactor ()` const
- `std::size_t getIndex (double scaleFactor, int round=0)` const
- `double getScaleFactor (std::size_t index)` const
- `double getScaleFactorInv (std::size_t index)` const
- `double getXScaleFactor (std::size_t index)` const
- `double getXScaleFactorInv (std::size_t index)` const
- `double getYScaleFactor (std::size_t index)` const
- `double getYScaleFactorInv (std::size_t index)` const
- `IplImageWrapper & getImage (std::size_t index)`
- `const IplImageWrapper & getImage (std::size_t index)` const
- `IplImageWrapper getImage (double scaleFactor, int round=0)`
- `const IplImageWrapper & getImage (double scaleFactor, int round=0)` const
- `void rebuild (IplImageWrapper image)`
- `IplImagePyramid (const IplImagePyramid &pyramid)`
- `IplImagePyramid (IplImageWrapper image, double scaleFactor)`
- `IplImagePyramid (CvSize initSize, int depth, int nChannels, double scaleFactor)`
- `IplImagePyramid & operator= (const IplImagePyramid &pyramid)`
- `operator const bool ()` const
- `operator bool ()`
- `std::size_t numOfLevels ()` const
- `double getScaleFactor ()` const
- `std::size_t getIndex (double scaleFactor, int round=0)` const
- `double getScaleFactor (std::size_t index)` const
- `double getScaleFactorInv (std::size_t index)` const
- `double getXScaleFactor (std::size_t index)` const
- `double getXScaleFactorInv (std::size_t index)` const
- `double getYScaleFactor (std::size_t index)` const
- `double getYScaleFactorInv (std::size_t index)` const
- `IplImageWrapper & getImage (std::size_t index)`

- const [IplImageWrapper](#) & **getImage** (std::size\_t index) const
- [IplImageWrapper](#) **getImage** (double scaleFactor, int round=0)
- const [IplImageWrapper](#) & **getImage** (double scaleFactor, int round=0) const
- void **rebuild** ([IplImageWrapper](#) image)

### Protected Attributes

- std::vector< [IplImageWrapper](#) > **\_imagePyramid**
- std::vector< double > **\_scaleFactors**
- std::vector< double > **\_scaleFactorsInv**
- std::vector< double > **\_xScaleFactors**
- std::vector< double > **\_xScaleFactorsInv**
- std::vector< double > **\_yScaleFactors**
- std::vector< double > **\_yScaleFactorsInv**
- double **\_scaleFactor**
- double **\_epsilon**

#### 4.8.1 Detailed Description

Definition at line 19 of file include/densetrack/IplImagePyramid.h.

#### 4.8.2 Constructor & Destructor Documentation

##### 4.8.2.1 IplImagePyramid::IplImagePyramid (IplImageWrapper *image*, double *scaleFactor*)

NOTE: the image is referenced on the lowest pyramid level!

Definition at line 219 of file lib/densetrack/IplImagePyramid.cpp.

##### 4.8.2.2 IplImagePyramid::IplImagePyramid (CvSize *initSize*, int *depth*, int *nChannels*, double *scaleFactor*)

Build an empty pyramid (pixel values are set to zero).

Definition at line 226 of file lib/densetrack/IplImagePyramid.cpp.

##### 4.8.2.3 IplImagePyramid::IplImagePyramid (IplImageWrapper *image*, double *scaleFactor*)

NOTE: the image is referenced on the lowest pyramid level!

##### 4.8.2.4 IplImagePyramid::IplImagePyramid (CvSize *initSize*, int *depth*, int *nChannels*, double *scaleFactor*)

Build an empty pyramid (pixel values are set to zero).

##### 4.8.2.5 IplImagePyramid::IplImagePyramid (IplImageWrapper *image*, double *scaleFactor*)

NOTE: the image is referenced on the lowest pyramid level!

#### 4.8.2.6 IplImagePyramid::IplImagePyramid (CvSize *initSize*, int *depth*, int *nChannels*, double *scaleFactor*)

Build an empty pyramid (pixel values are set to zero).

#### 4.8.2.7 IplImagePyramid::IplImagePyramid (IplImageWrapper *image*, double *scaleFactor*)

NOTE: the image is referenced on the lowest pyramid level!

#### 4.8.2.8 IplImagePyramid::IplImagePyramid (CvSize *initSize*, int *depth*, int *nChannels*, double *scaleFactor*)

Build an empty pyramid (pixel values are set to zero).

#### 4.8.2.9 IplImagePyramid::IplImagePyramid (IplImageWrapper *image*, double *scaleFactor*)

NOTE: the image is referenced on the lowest pyramid level!

#### 4.8.2.10 IplImagePyramid::IplImagePyramid (CvSize *initSize*, int *depth*, int *nChannels*, double *scaleFactor*)

Build an empty pyramid (pixel values are set to zero).

### 4.8.3 Member Function Documentation

#### 4.8.3.1 IplImageWrapper IplImagePyramid::getImage (double *scaleFactor*, int *round* = 0)

##### Parameters

*round* see [getIndex\(\)](#)

#### 4.8.3.2 IplImageWrapper IplImagePyramid::getImage (double *scaleFactor*, int *round* = 0)

##### Parameters

*round* see [getIndex\(\)](#)

#### 4.8.3.3 IplImageWrapper IplImagePyramid::getImage (double *scaleFactor*, int *round* = 0)

##### Parameters

*round* see [getIndex\(\)](#)

#### 4.8.3.4 IplImageWrapper IplImagePyramid::getImage (double *scaleFactor*, int *round* = 0)

##### Parameters

*round* see [getIndex\(\)](#)

#### 4.8.3.5 IplImageWrapper IplImagePyramid::getImage (double *scaleFactor*, int *round* = 0)

##### Parameters

*round* see [getIndex\(\)](#)

Definition at line 334 of file lib/densetrack/IplImagePyramid.cpp.

#### 4.8.3.6 std::size\_t IplImagePyramid::getIndex (double *scaleFactor*, int *round* = 0) const

*round* == 0 => take the closest level (i.e., rounding) *round* < 0 => take the next level with a smaller factor (i.e., flooring) *round* > 0 => take the next level with a bigger factor (i.e., ceiling)

#### 4.8.3.7 std::size\_t IplImagePyramid::getIndex (double *scaleFactor*, int *round* = 0) const

*round* == 0 => take the closest level (i.e., rounding) *round* < 0 => take the next level with a smaller factor (i.e., flooring) *round* > 0 => take the next level with a bigger factor (i.e., ceiling)

#### 4.8.3.8 std::size\_t IplImagePyramid::getIndex (double *scaleFactor*, int *round* = 0) const

*round* == 0 => take the closest level (i.e., rounding) *round* < 0 => take the next level with a smaller factor (i.e., flooring) *round* > 0 => take the next level with a bigger factor (i.e., ceiling)

#### 4.8.3.9 std::size\_t IplImagePyramid::getIndex (double *scaleFactor*, int *round* = 0) const

*round* == 0 => take the closest level (i.e., rounding) *round* < 0 => take the next level with a smaller factor (i.e., flooring) *round* > 0 => take the next level with a bigger factor (i.e., ceiling)

#### 4.8.3.10 std::size\_t IplImagePyramid::getIndex (double *scaleFactor*, int *round* = 0) const

*round* == 0 => take the closest level (i.e., rounding) *round* < 0 => take the next level with a smaller factor (i.e., flooring) *round* > 0 => take the next level with a bigger factor (i.e., ceiling)

Definition at line 150 of file lib/densetrack/IplImagePyramid.cpp.

#### 4.8.3.11 void IplImagePyramid::rebuild (IplImageWrapper *image*)

rebuilds the pyramid (re-using the already allocated space) with the given image NOTE: this image needs to have the exact same size as the initial scale

#### 4.8.3.12 void IplImagePyramid::rebuild (IplImageWrapper *image*)

rebuilds the pyramid (re-using the already allocated space) with the given image NOTE: this image needs to have the exact same size as the initial scale

#### 4.8.3.13 void IplImagePyramid::rebuild (IplImageWrapper *image*)

rebuilds the pyramid (re-using the already allocated space) with the given image NOTE: this image needs to have the exact same size as the initial scale

**4.8.3.14 void IplImagePyramid::rebuild (IplImageWrapper *image*)**

rebuilds the pyramid (re-using the already allocated space) with the given image NOTE: this image needs to have the exact same size as the initial scale

**4.8.3.15 void IplImagePyramid::rebuild (IplImageWrapper *image*)**

rebuilds the pyramid (re-using the already allocated space) with the given image NOTE: this image needs to have the exact same size as the initial scale

Definition at line 190 of file lib/densetrack/IplImagePyramid.cpp.

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

- include/densetrack/IplImagePyramid.h
- include/IplImagePyramid.h
- integration/densetrack/IplImagePyramid.h
- lib/densetrack/IplImagePyramid.h
- modules/densetrack/IplImagePyramid.h
- lib/densetrack/IplImagePyramid.cpp
- modules/densetrack/IplImagePyramid.cpp
- src/IplImagePyramid.cpp

## 4.9 IplImageWrapper Class Reference

```
#include <IplImageWrapper.h>
```

### Public Member Functions

- **IplImageWrapper** (IplImage \*newImg=NULL, bool isOwner=true)
- **IplImageWrapper** (CvSize size, int depth, int channels)
- **IplImageWrapper** (std::string fileName)
- **IplImageWrapper** (const [IplImageWrapper](#) &newImg)
- [IplImageWrapper](#) **clone** () const
- [IplImageWrapper](#) & **operator=** (const [IplImageWrapper](#) &img)
- **operator IplImage \*** ()
- **operator const IplImage \*** () const
- **operator const bool** () const
- **operator bool** ()
- IplImage \* **operator->** ()
- const IplImage \* **operator->** () const
- IplImage \* **getReference** ()
- const IplImage \* **getReference** () const
- std::size\_t **numOfReferences** () const
- bool **hasMask** () const
- [Box](#)< int > **getMask** () const
- void **setMask** (const [Box](#)< int > &mask)
- void **clearMask** ()
- **IplImageWrapper** (IplImage \*newImg=NULL, bool isOwner=true)
- **IplImageWrapper** (CvSize size, int depth, int channels)
- **IplImageWrapper** (std::string fileName)
- **IplImageWrapper** (const [IplImageWrapper](#) &newImg)
- [IplImageWrapper](#) **clone** () const
- [IplImageWrapper](#) & **operator=** (const [IplImageWrapper](#) &img)
- **operator IplImage \*** ()
- **operator const IplImage \*** () const
- **operator const bool** () const
- **operator bool** ()
- IplImage \* **operator->** ()
- const IplImage \* **operator->** () const
- IplImage \* **getReference** ()
- const IplImage \* **getReference** () const
- std::size\_t **numOfReferences** () const
- bool **hasMask** () const
- [Box](#)< int > **getMask** () const
- void **setMask** (const [Box](#)< int > &mask)
- void **clearMask** ()
- **IplImageWrapper** (IplImage \*newImg=NULL, bool isOwner=true)
- **IplImageWrapper** (CvSize size, int depth, int channels)
- **IplImageWrapper** (std::string fileName)
- **IplImageWrapper** (const [IplImageWrapper](#) &newImg)
- [IplImageWrapper](#) **clone** () const

- [IplImageWrapper](#) & **operator=** (const [IplImageWrapper](#) &img)
- **operator IplImage \*** ()
- **operator const IplImage \*** () const
- **operator const bool** () const
- **operator bool** ()
- IplImage \* **operator->** ()
- const IplImage \* **operator->** () const
- IplImage \* **getReference** ()
- const IplImage \* **getReference** () const
- std::size\_t **numOfReferences** () const
- bool **hasMask** () const
- [Box](#)< int > **getMask** () const
- void **setMask** (const [Box](#)< int > &mask)
- void **clearMask** ()
- **IplImageWrapper** (IplImage \*newImg=NULL, bool isOwner=true)
- **IplImageWrapper** (CvSize size, int depth, int channels)
- **IplImageWrapper** (std::string fileName)
- **IplImageWrapper** (const [IplImageWrapper](#) &newImg)
- [IplImageWrapper](#) **clone** () const
- [IplImageWrapper](#) & **operator=** (const [IplImageWrapper](#) &img)
- **operator IplImage \*** ()
- **operator const IplImage \*** () const
- **operator const bool** () const
- **operator bool** ()
- IplImage \* **operator->** ()
- const IplImage \* **operator->** () const
- IplImage \* **getReference** ()
- const IplImage \* **getReference** () const
- std::size\_t **numOfReferences** () const
- bool **hasMask** () const
- [Box](#)< int > **getMask** () const
- void **setMask** (const [Box](#)< int > &mask)
- void **clearMask** ()
- **IplImageWrapper** (IplImage \*newImg=NULL, bool isOwner=true)
- **IplImageWrapper** (CvSize size, int depth, int channels)
- **IplImageWrapper** (std::string fileName)
- **IplImageWrapper** (const [IplImageWrapper](#) &newImg)
- [IplImageWrapper](#) **clone** () const
- [IplImageWrapper](#) & **operator=** (const [IplImageWrapper](#) &img)
- **operator IplImage \*** ()
- **operator const IplImage \*** () const
- **operator const bool** () const
- **operator bool** ()
- IplImage \* **operator->** ()
- const IplImage \* **operator->** () const
- IplImage \* **getReference** ()
- const IplImage \* **getReference** () const
- std::size\_t **numOfReferences** () const
- bool **hasMask** () const
- [Box](#)< int > **getMask** () const
- void **setMask** (const [Box](#)< int > &mask)
- void **clearMask** ()



## Protected Member Functions

- void **decrementAndFree** ()
- void **decrementAndFree** ()
- void **decrementAndFree** ()
- void **decrementAndFree** ()
- void **decrementAndFree** ()

## Protected Attributes

- IplImage \* **\_img**
- std::size\_t \* **\_nRefs**
- boost::optional< [Box](#)< int > > **\_mask**

### 4.9.1 Detailed Description

Wrapper in order to use IplImages more easily in STL containers .. for that we need a destructor that releases an image correctly using cvReleaseImage(). We also want to have smart pointers with a reference count in order to copy containers.

NOTE: Do not use generic algorithms with this class! (See discussions on auto\_ptr, similar thoughts/problems apply to this smart pointer!)

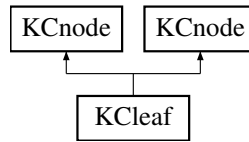
Definition at line 25 of file include/densetrack/IplImageWrapper.h.

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

- include/densetrack/IplImageWrapper.h
- include/IplImageWrapper.h
- integration/densetrack/IplImageWrapper.h
- lib/densetrack/IplImageWrapper.h
- modules/densetrack/IplImageWrapper.h
- lib/densetrack/IplImageWrapper.cpp
- modules/densetrack/IplImageWrapper.cpp
- src/IplImageWrapper.cpp

## 4.10 KCleaf Class Reference

Inheritance diagram for KCleaf:



### Public Member Functions

- **KCleaf** (int dim, [KMorthRect](#) &bb, int n, KMdatIdxArray b)
- KMpoint **getPoint** ()
- virtual void **makeSums** (int &n, KMpoint &theSum, double &theSumSq)
- virtual void **getNeighbors** (KMctrIdxArray cands, int kCands)
- virtual void **getAssignments** (KMctrIdxArray cands, int kCands, KMctrIdxArray closeCtr, double \*sqDist)
- virtual void **sampleCtr** (KMpoint c, [KMorthRect](#) &bb)
- virtual void **print** (int level)
- **KCleaf** (int dim, [KMorthRect](#) &bb, int n, KMdatIdxArray b)
- KMpoint **getPoint** ()
- virtual void **makeSums** (int &n, KMpoint &theSum, double &theSumSq)
- virtual void **getNeighbors** (KMctrIdxArray cands, int kCands)
- virtual void **getAssignments** (KMctrIdxArray cands, int kCands, KMctrIdxArray closeCtr, double \*sqDist)
- virtual void **sampleCtr** (KMpoint c, [KMorthRect](#) &bb)
- virtual void **print** (int level)

### Protected Attributes

- KMidxArray **bkt**

#### 4.10.1 Detailed Description

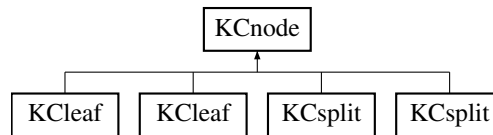
Definition at line 192 of file include/kmlocal/KCtree.h.

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

- include/kmlocal/KCtree.h
- lib/kmlocal-1.7.2/src/KCtree.h
- KCtree.cpp

## 4.11 KNode Class Reference

Inheritance diagram for KNode:



### Public Member Functions

- **KNode** (int dim, [KMorthRect](#) &bb)
- void **cellMidpt** (KMpoint pt)
- [KMorthRect](#) & **bndBox** ()
- virtual void **makeSums** (int &n, KMpoint &theSum, double &theSumSq)=0
- virtual void **getNeighbors** (KMctrIdxArray cand, int kCands)=0
- virtual void **getAssignments** (KMctrIdxArray cand, int kCands, KMctrIdxArray closeCtr, double \*sqDist)=0
- virtual void **sampleCtr** (KMpoint c, [KMorthRect](#) &bb)=0
- virtual void **print** (int level)=0
- int **n\_nodes** ()
- **KNode** (int dim, [KMorthRect](#) &bb)
- void **cellMidpt** (KMpoint pt)
- [KMorthRect](#) & **bndBox** ()
- virtual void **makeSums** (int &n, KMpoint &theSum, double &theSumSq)=0
- virtual void **getNeighbors** (KMctrIdxArray cand, int kCands)=0
- virtual void **getAssignments** (KMctrIdxArray cand, int kCands, KMctrIdxArray closeCtr, double \*sqDist)=0
- virtual void **sampleCtr** (KMpoint c, [KMorthRect](#) &bb)=0
- virtual void **print** (int level)=0
- int **n\_nodes** ()

### Protected Attributes

- const int **multCand**
- int **n\_data**
- KMpoint **sum**
- double **sumSq**
- [KMorthRect](#) **bnd\_box**

### Friends

- class [KCTree](#)

### 4.11.1 Detailed Description

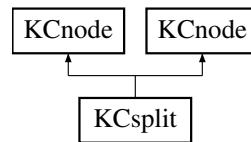
Definition at line 136 of file include/kmlocal/KCtree.h.

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

- include/kmlocal/KCtree.h
- lib/kmlocal-1.7.2/src/KCtree.h
- KCtree.cpp

## 4.12 KCsplit Class Reference

Inheritance diagram for KCsplit:



### Public Member Functions

- **KCsplit** (int dim, [KMorthRect](#) &bb, int cd, KMcoord cv, KMcoord lv, KMcoord hv, [KCptr](#) lc=NULL, [KCptr](#) hc=NULL)
- virtual void **makeSums** (int &n, KMpoint &theSum, double &theSumSq)
- virtual void **getNeighbors** (KMctrIdxArray cands, int kCands)
- virtual void **getAssignments** (KMctrIdxArray cands, int kCands, KMctrIdxArray closeCtr, double \*sqDist)
- virtual void **sampleCtr** (KMpoint c, [KMorthRect](#) &bb)
- virtual void **print** (int level)
- **KCsplit** (int dim, [KMorthRect](#) &bb, int cd, KMcoord cv, KMcoord lv, KMcoord hv, [KCptr](#) lc=NULL, [KCptr](#) hc=NULL)
- virtual void **makeSums** (int &n, KMpoint &theSum, double &theSumSq)
- virtual void **getNeighbors** (KMctrIdxArray cands, int kCands)
- virtual void **getAssignments** (KMctrIdxArray cands, int kCands, KMctrIdxArray closeCtr, double \*sqDist)
- virtual void **sampleCtr** (KMpoint c, [KMorthRect](#) &bb)
- virtual void **print** (int level)

### Protected Attributes

- int **cut\_dim**
- KMcoord **cut\_val**
- KMcoord **cd\_bnds** [2]
- [KCptr](#) **child** [2]

#### 4.12.1 Detailed Description

Definition at line 242 of file include/kmlocal/KCtree.h.

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

- include/kmlocal/KCtree.h
- lib/kmlocal-1.7.2/src/KCtree.h
- KCtree.cpp

## 4.13 KCtree Class Reference

### Public Member Functions

- **KCtree** (KMdataArray pa, int n, int dd, int n\_max=0, KMpoint bb\_lo=NULL, KMpoint bb\_hi=NULL)
- void **getNeighbors** ([KMfilterCenters](#) &ctrs)
- void **getAssignments** ([KMfilterCenters](#) &ctrs, KMctrIdxArray closeCtr, double \*sqDist)
- void **sampleCtr** (KMpoint c)
- void **print** (bool with\_pts)
- **KCtree** (KMdataArray pa, int n, int dd, int n\_max=0, KMpoint bb\_lo=NULL, KMpoint bb\_hi=NULL)
- void **getNeighbors** ([KMfilterCenters](#) &ctrs)
- void **getAssignments** ([KMfilterCenters](#) &ctrs, KMctrIdxArray closeCtr, double \*sqDist)
- void **sampleCtr** (KMpoint c)
- void **print** (bool with\_pts)

### Protected Member Functions

- void **skeletonTree** (KMdataArray pa, int n, int dd, int n\_max, KMpoint bb\_lo, KMpoint bb\_hi, KMdatIdxArray pi)
- [KCptr](#) **buildKcTree** (KMdataArray pa, KMdatIdxArray pidx, int n, int dim, [KMorthRect](#) &bnd\_box)
- void **skeletonTree** (KMdataArray pa, int n, int dd, int n\_max, KMpoint bb\_lo, KMpoint bb\_hi, KMdatIdxArray pi)
- [KCptr](#) **buildKcTree** (KMdataArray pa, KMdatIdxArray pidx, int n, int dim, [KMorthRect](#) &bnd\_box)

### Protected Attributes

- int **dim**
- int **n\_pts**
- int **max\_pts**
- KMdataArray **pts**
- KMdatIdxArray **pidx**
- [KCptr](#) **root**
- [KMorthRect](#) **bnd\_box**

#### 4.13.1 Detailed Description

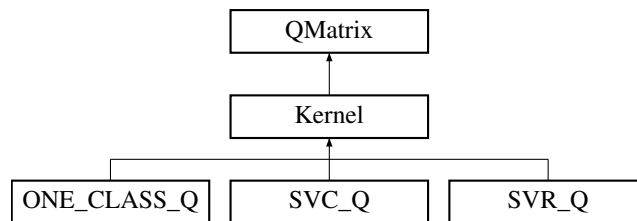
Definition at line 56 of file include/kmlocal/KCtree.h.

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

- include/kmlocal/KCtree.h
- lib/kmlocal-1.7.2/src/KCtree.h
- KCtree.cpp

## 4.14 Kernel Class Reference

Inheritance diagram for Kernel:



### Public Member Functions

- **Kernel** (int l, [svm\\_node](#) \*const \*x, const [svm\\_parameter](#) &param)
- virtual [Qfloat](#) \* **get\_Q** (int column, int len) const =0
- virtual [double](#) \* **get\_QD** () const =0
- virtual void **swap\_index** (int i, int j) const

### Static Public Member Functions

- static [double](#) **k\_function** (const [svm\\_node](#) \*x, const [svm\\_node](#) \*y, const [svm\\_parameter](#) &param)

### Protected Attributes

- [double](#)(**Kernel::\* kernel\_function** )(int i, int j) const

#### 4.14.1 Detailed Description

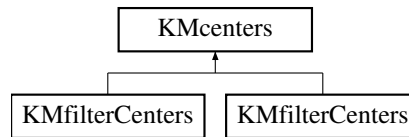
Definition at line 202 of file `svm.cpp`.

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

- `svm.cpp`

## 4.15 KMcenters Class Reference

Inheritance diagram for KMcenters:



### Public Member Functions

- **KMcenters** (int k, [KMdata](#) &p)
- **KMcenters** (const [KMcenters](#) &s)
- [KMcenters](#) & **operator=** (const [KMcenters](#) &s)
- int **getDim** () const
- int **getNpts** () const
- int **getK** () const
- [KMdata](#) & **getData** ()
- KMpointArray **getDataPts** () const
- KMcenterArray **getCtrPts** () const
- KMcenter & **operator[]** (int i)
- const KMcenter & **operator[]** (int i) const
- void **resize** (int k)
- virtual void **print** (bool fancy=true)
- **KMcenters** (int k, [KMdata](#) &p)
- **KMcenters** (const [KMcenters](#) &s)
- [KMcenters](#) & **operator=** (const [KMcenters](#) &s)
- int **getDim** () const
- int **getNpts** () const
- int **getK** () const
- [KMdata](#) & **getData** ()
- KMpointArray **getDataPts** () const
- KMcenterArray **getCtrPts** () const
- KMcenter & **operator[]** (int i)
- const KMcenter & **operator[]** (int i) const
- void **resize** (int k)
- virtual void **print** (bool fancy=true)

### Protected Attributes

- int **kCtrs**
- [KMdata](#) \* **pts**
- KMcenterArray **ctrs**



### 4.15.1 Detailed Description

Definition at line 36 of file include/kmlocal/KMcenters.h.

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

- include/kmlocal/KMcenters.h
- lib/kmlocal-1.7.2/src/KMcenters.h
- KMcenters.cpp

## 4.16 KMdata Class Reference

### Public Member Functions

- **KMdata** (int d, int n)
- int **getDim** () const
- int **getNPts** () const
- KMdataArray **getPts** () const
- **KCtree** \* **getKcTree** () const
- KMdataPoint & **operator**[] (int i)
- const KMdataPoint & **operator**[] (int i) const
- void **setNPts** (int n)
- void **buildKcTree** ()
- virtual void **sampleCtr** (KMpoint sample)
- virtual void **sampleCtrs** (KMpointArray sample, int k, bool allowDuplicate)
- void **resize** (int d, int n)
- void **print** (bool fancy=true)
- **KMdata** (int d, int n)
- int **getDim** () const
- int **getNPts** () const
- KMdataArray **getPts** () const
- **KCtree** \* **getKcTree** () const
- KMdataPoint & **operator**[] (int i)
- const KMdataPoint & **operator**[] (int i) const
- void **setNPts** (int n)
- void **buildKcTree** ()
- virtual void **sampleCtr** (KMpoint sample)
- virtual void **sampleCtrs** (KMpointArray sample, int k, bool allowDuplicate)
- void **resize** (int d, int n)
- void **print** (bool fancy=true)

### 4.16.1 Detailed Description

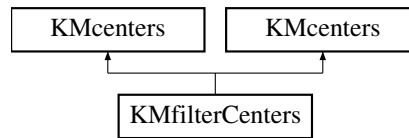
Definition at line 49 of file include/kmlocal/KMdata.h.

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

- include/kmlocal/KMdata.h
- lib/kmlocal-1.7.2/src/KMdata.h
- KMdata.cpp

## 4.17 KMfilterCenters Class Reference

Inheritance diagram for KMfilterCenters:



### Public Member Functions

- **KMfilterCenters** (int k, [KMdata](#) &p, double df=1)
- **KMfilterCenters** (const [KMfilterCenters](#) &s)
- [KMfilterCenters](#) & **operator=** (const [KMfilterCenters](#) &s)
- [KMpointArray](#) **getSums** (bool autoUpdate=true)
- double \* **getSumSqs** (bool autoUpdate=true)
- int \* **getWeights** (bool autoUpdate=true)
- double **getDist** (bool autoUpdate=true)
- double **getAvgDist** (bool autoUpdate=true)
- double \* **getDists** (bool autoUpdate=true)
- void **getAssignments** ([KMctrIdxArray](#) closeCtr, double \*sqDist)
- void **genRandom** ()
- void **lloyd1Stage** ()
- void **swap1Stage** ()
- virtual void **print** (bool fancy=true)
- **KMfilterCenters** (int k, [KMdata](#) &p, double df=1)
- **KMfilterCenters** (const [KMfilterCenters](#) &s)
- [KMfilterCenters](#) & **operator=** (const [KMfilterCenters](#) &s)
- [KMpointArray](#) **getSums** (bool autoUpdate=true)
- double \* **getSumSqs** (bool autoUpdate=true)
- int \* **getWeights** (bool autoUpdate=true)
- double **getDist** (bool autoUpdate=true)
- double **getAvgDist** (bool autoUpdate=true)
- double \* **getDists** (bool autoUpdate=true)
- void **getAssignments** ([KMctrIdxArray](#) closeCtr, double \*sqDist)
- void **genRandom** ()
- void **lloyd1Stage** ()
- void **swap1Stage** ()
- virtual void **print** (bool fancy=true)

### Protected Member Functions

- void **computeDistortion** ()
- void **moveToCentroid** ()
- void **swapOneCenter** (bool allowDuplicate=true)
- void **validate** ()
- void **invalidate** ()
- void **computeDistortion** ()

- void **moveToCentroid** ()
- void **swapOneCenter** (bool allowDuplicate=true)
- void **validate** ()
- void **invalidate** ()

### Protected Attributes

- KMpointArray **sums**
- double \* **sumSqs**
- int \* **weights**
- double \* **dists**
- double **currDist**
- bool **valid**
- double **dampFactor**

#### 4.17.1 Detailed Description

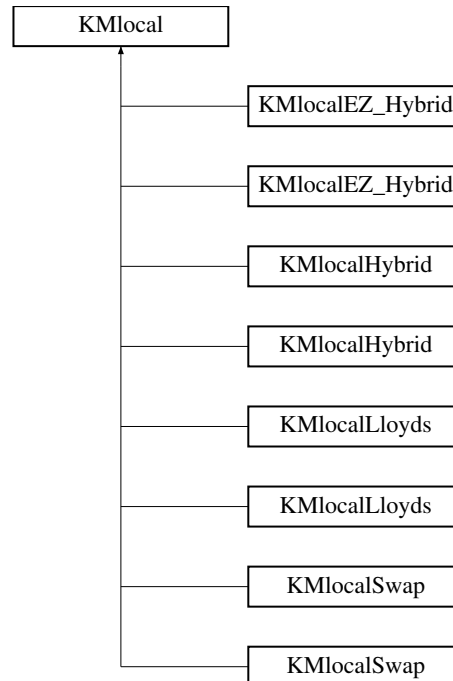
Definition at line 105 of file include/kmlocal/KMfilterCenters.h.

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

- include/kmlocal/KMfilterCenters.h
- lib/kmlocal-1.7.2/src/KMfilterCenters.h
- KMfilterCenters.cpp

## 4.18 KMlocal Class Reference

Inheritance diagram for KMlocal:



### Public Member Functions

- **KMlocal** (const [KMfilterCenters](#) &sol, const [KMterm](#) &t)
- virtual [KMfilterCenters](#) **execute** ()
- int **getTotalStages** () const
- **KMlocal** (const [KMfilterCenters](#) &sol, const [KMterm](#) &t)
- virtual [KMfilterCenters](#) **execute** ()
- int **getTotalStages** () const

### Protected Member Functions

- virtual void **printStageStats** ()
- virtual void **reset** ()
- virtual bool **isDone** () const
- virtual void **beginRun** ()
- virtual void **beginStage** ()
- virtual [KMalg](#) **selectMethod** ()=0
- virtual void **endStage** ()
- virtual bool **isRunDone** ()
- virtual void **endRun** ()
- virtual void **tryAcceptance** ()
- virtual void **printStageStats** ()
- virtual void **reset** ()

- virtual bool **isDone** () const
- virtual void **beginRun** ()
- virtual void **beginStage** ()
- virtual KMalg **selectMethod** ()=0
- virtual void **endStage** ()
- virtual bool **isRunDone** ()
- virtual void **endRun** ()
- virtual void **tryAcceptance** ()

### Protected Attributes

- int **nPts**
- int **kCtrs**
- int **dim**
- [KMterm](#) **term**
- int **maxTotStage**
- int **stageNo**
- int **runInitStage**
- [KMfilterCenters](#) **curr**
- [KMfilterCenters](#) **best**

#### 4.18.1 Detailed Description

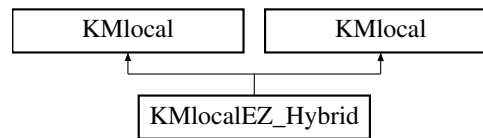
Definition at line 104 of file include/kmlocal/KMlocal.h.

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

- include/kmlocal/KMlocal.h
- lib/kmlocal-1.7.2/src/KMlocal.h
- KMlocal.cpp

## 4.19 KMlocalEZ\_Hybrid Class Reference

Inheritance diagram for KMlocalEZ\_Hybrid:



### Public Member Functions

- **KMlocalEZ\_Hybrid** (const [KMfilterCenters](#) &sol, const [KMterm](#) &t)
- **KMlocalEZ\_Hybrid** (const [KMfilterCenters](#) &sol, const [KMterm](#) &t)

### Protected Member Functions

- double **consecRDL** ()
- virtual void **printStageStats** ()
- virtual void **printRunStats** ()
- virtual void **reset** ()
- virtual void **beginStage** ()
- virtual [KMalg](#) **selectMethod** ()
- virtual void **endStage** ()
- virtual bool **isRunDone** ()
- virtual void **endRun** ()
- virtual void **tryAcceptance** ()
- double **consecRDL** ()
- virtual void **printStageStats** ()
- virtual void **printRunStats** ()
- virtual void **reset** ()
- virtual void **beginStage** ()
- virtual [KMalg](#) **selectMethod** ()
- virtual void **endStage** ()
- virtual bool **isRunDone** ()
- virtual void **endRun** ()
- virtual void **tryAcceptance** ()

#### 4.19.1 Detailed Description

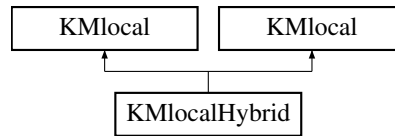
Definition at line 783 of file include/kmlocal/KMlocal.h.

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

- include/kmlocal/KMlocal.h
- lib/kmlocal-1.7.2/src/KMlocal.h

## 4.20 KMlocalHybrid Class Reference

Inheritance diagram for KMlocalHybrid:



### Public Member Functions

- **KMlocalHybrid** (const [KMfilterCenters](#) &sol, const [KMterm](#) &t)
- **KMlocalHybrid** (const [KMfilterCenters](#) &sol, const [KMterm](#) &t)

### Protected Member Functions

- double **accumRDL** ()
- double **consecRDL** ()
- virtual void **printStageStats** ()
- virtual void **printRunStats** ()
- int **nTrials** ()
- bool **simAnnealAccept** (double rdl)
- void **initTempRuns** ()
- bool **isTempRunDone** ()
- void **endTempRun** ()
- virtual void **reset** ()
- virtual void **beginStage** ()
- virtual [KMalg](#) **selectMethod** ()
- virtual void **endStage** ()
- virtual bool **isRunDone** ()
- virtual void **endRun** ()
- virtual void **tryAcceptance** ()
- double **accumRDL** ()
- double **consecRDL** ()
- virtual void **printStageStats** ()
- virtual void **printRunStats** ()
- int **nTrials** ()
- bool **simAnnealAccept** (double rdl)
- void **initTempRuns** ()
- bool **isTempRunDone** ()
- void **endTempRun** ()
- virtual void **reset** ()
- virtual void **beginStage** ()
- virtual [KMalg](#) **selectMethod** ()
- virtual void **endStage** ()
- virtual bool **isRunDone** ()
- virtual void **endRun** ()
- virtual void **tryAcceptance** ()



### 4.20.1 Detailed Description

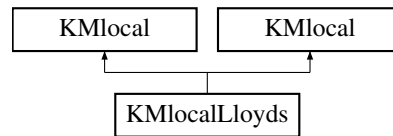
Definition at line 594 of file include/kmlocal/KMlocal.h.

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

- include/kmlocal/KMlocal.h
- lib/kmlocal-1.7.2/src/KMlocal.h

## 4.21 KMlocalLloyds Class Reference

Inheritance diagram for KMlocalLloyds:



### Public Member Functions

- **KMlocalLloyds** (const [KMfilterCenters](#) &sol, const [KMterm](#) &t)
- **KMlocalLloyds** (const [KMfilterCenters](#) &sol, const [KMterm](#) &t)

### Protected Member Functions

- double **accumRDL** ()
- virtual void **printStageStats** ()
- virtual void **printRunStats** ()
- virtual void **reset** ()
- virtual [KMalg](#) **selectMethod** ()
- virtual void **endStage** ()
- virtual bool **isRunDone** ()
- virtual void **endRun** ()
- double **accumRDL** ()
- virtual void **printStageStats** ()
- virtual void **printRunStats** ()
- virtual void **reset** ()
- virtual [KMalg](#) **selectMethod** ()
- virtual void **endStage** ()
- virtual bool **isRunDone** ()
- virtual void **endRun** ()

### 4.21.1 Detailed Description

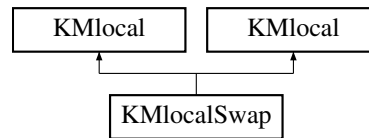
Definition at line 263 of file include/kmlocal/KMlocal.h.

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

- include/kmlocal/KMlocal.h
- lib/kmlocal-1.7.2/src/KMlocal.h

## 4.22 KMlocalSwap Class Reference

Inheritance diagram for KMlocalSwap:



### Public Member Functions

- **KMlocalSwap** (const [KMfilterCenters](#) &sol, const [KMterm](#) &t, int p=1)
- **KMlocalSwap** (const [KMfilterCenters](#) &sol, const [KMterm](#) &t, int p=1)

### Protected Member Functions

- virtual void **reset** ()
- virtual void **beginRun** ()
- virtual KMalg **selectMethod** ()
- virtual void **endStage** ()
- virtual bool **isRunDone** ()
- virtual void **endRun** ()
- virtual void **tryAcceptance** ()
- virtual void **reset** ()
- virtual void **beginRun** ()
- virtual KMalg **selectMethod** ()
- virtual void **endStage** ()
- virtual bool **isRunDone** ()
- virtual void **endRun** ()
- virtual void **tryAcceptance** ()

### 4.22.1 Detailed Description

Definition at line 363 of file include/kmlocal/KMlocal.h.

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

- include/kmlocal/KMlocal.h
- lib/kmlocal-1.7.2/src/KMlocal.h

## 4.23 KMorthRect Class Reference

### Public Member Functions

- **KMorthRect** (int dd, KMcoord l=0, KMcoord h=0)
- **KMorthRect** (int dd, const [KMorthRect](#) &r)
- **KMorthRect** (int dd, KMpoint l, KMpoint h)
- bool **inside** (int dim, KMpoint p)
- void **expand** (int dim, double x, [KMorthRect](#) r)
- void **sample** (int dim, KMpoint p)
- **KMorthRect** (int dd, KMcoord l=0, KMcoord h=0)
- **KMorthRect** (int dd, const [KMorthRect](#) &r)
- **KMorthRect** (int dd, KMpoint l, KMpoint h)
- bool **inside** (int dim, KMpoint p)
- void **expand** (int dim, double x, [KMorthRect](#) r)
- void **sample** (int dim, KMpoint p)

### Public Attributes

- KMpoint **lo**
- KMpoint **hi**

#### 4.23.1 Detailed Description

Definition at line 380 of file include/kmlocal/KM\_ANN.h.

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

- include/kmlocal/KM\_ANN.h
- lib/kmlocal-1.7.2/src/KM\_ANN.h
- KM\_ANN.cpp

## 4.24 KMterm Class Reference

### Public Member Functions

- **KMterm** (double a, double b, double c, double d, double mcr, double mar, int mrs, double ipa, int trl, double trf)
- void **setMaxTotStage** (int i, double val)
- void **setAbsMaxTotStage** (int s)
- int **getMaxTotStage** (int k, int n) const
- double **getMinConsecRDL** () const
- double **getMinAccumRDL** () const
- int **getMaxRunStage** () const
- void **setMinConsecRDL** (double rdl)
- void **setMinAccumRDL** (double rdl)
- void **setMaxRunStage** (int ms)
- double **getInitProbAccept** () const
- void **setInitProbAccept** (double ipa)
- int **getTempRunLength** () const
- void **setTempRunLength** (int trl)
- double **getTempReducFact** () const
- void **setTempReducFact** (double trf)
- **KMterm** (double a, double b, double c, double d, double mcr, double mar, int mrs, double ipa, int trl, double trf)
- void **setMaxTotStage** (int i, double val)
- void **setAbsMaxTotStage** (int s)
- int **getMaxTotStage** (int k, int n) const
- double **getMinConsecRDL** () const
- double **getMinAccumRDL** () const
- int **getMaxRunStage** () const
- void **setMinConsecRDL** (double rdl)
- void **setMinAccumRDL** (double rdl)
- void **setMaxRunStage** (int ms)
- double **getInitProbAccept** () const
- void **setInitProbAccept** (double ipa)
- int **getTempRunLength** () const
- void **setTempRunLength** (int trl)
- double **getTempReducFact** () const
- void **setTempReducFact** (double trf)

### Protected Member Functions

- int **maxStage** (const double param[KM\_TERM\_VEC\_LEN], int k, int n) const
- int **maxStage** (const double param[KM\_TERM\_VEC\_LEN], int k, int n) const

### 4.24.1 Detailed Description

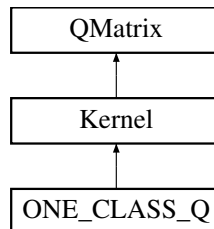
Definition at line 92 of file include/kmlocal/KMterm.h.

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

- include/kmlocal/KMterm.h
- lib/kmlocal-1.7.2/src/KMterm.h
- KMterm.cpp

## 4.25 ONE\_CLASS\_Q Class Reference

Inheritance diagram for ONE\_CLASS\_Q:



### Public Member Functions

- **ONE\_CLASS\_Q** (const [svm\\_problem](#) &prob, const [svm\\_parameter](#) &param)
- Qfloat \* **get\_Q** (int i, int len) const
- double \* **get\_QD** () const
- void **swap\_index** (int i, int j) const

### 4.25.1 Detailed Description

Definition at line 1316 of file svm.cpp.

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

- svm.cpp

## 4.26 Point< T > Class Template Reference

### Public Types

- typedef T **ValueType**
- typedef T **ValueType**
- typedef T **ValueType**
- typedef T **ValueType**
- typedef T **ValueType**

### Public Member Functions

- **Point** (T xpos, T ypos)
- bool **isNull** () const
- T **getX** () const
- T **getY** () const
- void **setX** (T x)
- void **setY** (T y)
- **Point**< T > & **operator+=** (const **Point**< T > &p)
- **Point**< T > & **operator-=** (const **Point**< T > &p)
- template<typename T2 >  
  **Point**< T > & **operator\*=** (T2 c)
- template<typename T2 >  
  **Point**< T > & **operator/=** (T2 c)
- template<typename T2 >  
  **operator Point**< T2 > () const
- **Point** (T xpos, T ypos)
- bool **isNull** () const
- T **getX** () const
- T **getY** () const
- void **setX** (T x)
- void **setY** (T y)
- **Point**< T > & **operator+=** (const **Point**< T > &p)
- **Point**< T > & **operator-=** (const **Point**< T > &p)
- template<typename T2 >  
  **Point**< T > & **operator\*=** (T2 c)
- template<typename T2 >  
  **Point**< T > & **operator/=** (T2 c)
- template<typename T2 >  
  **operator Point**< T2 > () const
- **Point** (T xpos, T ypos)
- bool **isNull** () const
- T **getX** () const
- T **getY** () const
- void **setX** (T x)
- void **setY** (T y)
- **Point**< T > & **operator+=** (const **Point**< T > &p)
- **Point**< T > & **operator-=** (const **Point**< T > &p)
- template<typename T2 >  
  **Point**< T > & **operator\*=** (T2 c)



- `template<typename T2 >`  
`Point< T > & operator/= (T2 c)`
- `template<typename T2 >`  
`operator Point< T2 > () const`
- `Point (T xpos, T ypos)`
- `bool isNull () const`
- `T getX () const`
- `T getY () const`
- `void setX (T x)`
- `void setY (T y)`
- `Point< T > & operator+= (const Point< T > &p)`
- `Point< T > & operator-= (const Point< T > &p)`
- `template<typename T2 >`  
`Point< T > & operator*= (T2 c)`
- `template<typename T2 >`  
`Point< T > & operator/= (T2 c)`
- `template<typename T2 >`  
`operator Point< T2 > () const`
- `Point (T xpos, T ypos)`
- `bool isNull () const`
- `T getX () const`
- `T getY () const`
- `void setX (T x)`
- `void setY (T y)`
- `Point< T > & operator+= (const Point< T > &p)`
- `Point< T > & operator-= (const Point< T > &p)`
- `template<typename T2 >`  
`Point< T > & operator*= (T2 c)`
- `template<typename T2 >`  
`Point< T > & operator/= (T2 c)`
- `template<typename T2 >`  
`operator Point< T2 > () const`

### 4.26.1 Detailed Description

`template<typename T> class Point< T >`

Definition at line 10 of file `include/densetrack/Point.h`.

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

- `include/densetrack/Point.h`
- `include/Point.h`
- `integration/densetrack/Point.h`
- `lib/densetrack/Point.h`
- `modules/densetrack/Point.h`

## 4.27 PointDesc Class Reference

### Public Member Functions

- **PointDesc** (const [DescInfo](#) &hogInfo, const [DescInfo](#) &hofInfo, const [DescInfo](#) &mbhInfo, const CvPoint2D32f &point\_)
- **PointDesc** (const [DescInfo](#) &hogInfo, const [DescInfo](#) &hofInfo, const [DescInfo](#) &mbhInfo, const CvPoint2D32f &point\_)
- **PointDesc** (const [DescInfo](#) &hogInfo, const [DescInfo](#) &hofInfo, const [DescInfo](#) &mbhInfo, const CvPoint2D32f &point\_)
- **PointDesc** (const [DescInfo](#) &hogInfo, const [DescInfo](#) &hofInfo, const [DescInfo](#) &mbhInfo, const CvPoint2D32f &point\_)

### Public Attributes

- std::vector< float > **hog**
- std::vector< float > **hof**
- std::vector< float > **mbhX**
- std::vector< float > **mbhY**
- CvPoint2D32f **point**

#### 4.27.1 Detailed Description

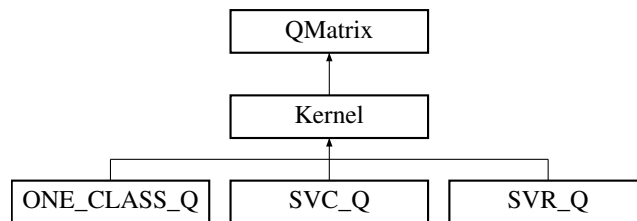
Definition at line 53 of file include/denseTrack.h.

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

- include/denseTrack.h
- naodensetrack.h
- DenseTrack.h
- modules/densetrack/denseTrack.h

## 4.28 QMatrix Class Reference

Inheritance diagram for QMatrix:



### Public Member Functions

- virtual Qfloat \* **get\_Q** (int column, int len) const =0
- virtual double \* **get\_QD** () const =0
- virtual void **swap\_index** (int i, int j) const =0

### 4.28.1 Detailed Description

Definition at line 194 of file svm.cpp.

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

- svm.cpp

## 4.29 Size< T > Class Template Reference

### Public Types

- typedef T **ValueType**
- typedef T **ValueType**
- typedef T **ValueType**
- typedef T **ValueType**
- typedef T **ValueType**

### Public Member Functions

- **Size** (T w, T h)
- bool **isEmpty** () const
- bool **isValid** () const
- T **getWidth** () const
- T **getHeight** () const
- void **setWidth** (T w)
- void **setHeight** (T h)
- void **transpose** ()
- T **getArea** () const
- **Size**< T > & **operator+=** (const **Size**< T > &)
- **Size**< T > & **operator-=** (const **Size**< T > &)
- template<typename T2 >  
  **Size**< T > & **operator\*=** (T2 c)
- template<typename T2 >  
  **Size**< T > & **operator/=** (T2 c)
- template<typename T2 >  
  **operator Size**< T2 > () const
- **Size** (T w, T h)
- bool **isEmpty** () const
- bool **isValid** () const
- T **getWidth** () const
- T **getHeight** () const
- void **setWidth** (T w)
- void **setHeight** (T h)
- void **transpose** ()
- T **getArea** () const
- **Size**< T > & **operator+=** (const **Size**< T > &)
- **Size**< T > & **operator-=** (const **Size**< T > &)
- template<typename T2 >  
  **Size**< T > & **operator\*=** (T2 c)
- template<typename T2 >  
  **Size**< T > & **operator/=** (T2 c)
- template<typename T2 >  
  **operator Size**< T2 > () const
- **Size** (T w, T h)
- bool **isEmpty** () const
- bool **isValid** () const
- T **getWidth** () const

- T **getHeight** () const
- void **setWidth** (T w)
- void **setHeight** (T h)
- void **transpose** ()
- T **getArea** () const
- **Size**< T > & **operator+=** (const **Size**< T > &)
- **Size**< T > & **operator-=** (const **Size**< T > &)
- template<typename T2 >  
  **Size**< T > & **operator\*=** (T2 c)
- template<typename T2 >  
  **Size**< T > & **operator/=** (T2 c)
- template<typename T2 >  
  **operator Size**< T2 > () const
- **Size** (T w, T h)
- bool **isEmpty** () const
- bool **isValid** () const
- T **getWidth** () const
- T **getHeight** () const
- void **setWidth** (T w)
- void **setHeight** (T h)
- void **transpose** ()
- T **getArea** () const
- **Size**< T > & **operator+=** (const **Size**< T > &)
- **Size**< T > & **operator-=** (const **Size**< T > &)
- template<typename T2 >  
  **Size**< T > & **operator\*=** (T2 c)
- template<typename T2 >  
  **Size**< T > & **operator/=** (T2 c)
- template<typename T2 >  
  **operator Size**< T2 > () const
- **Size** (T w, T h)
- bool **isEmpty** () const
- bool **isValid** () const
- T **getWidth** () const
- T **getHeight** () const
- void **setWidth** (T w)
- void **setHeight** (T h)
- void **transpose** ()
- T **getArea** () const
- **Size**< T > & **operator+=** (const **Size**< T > &)
- **Size**< T > & **operator-=** (const **Size**< T > &)
- template<typename T2 >  
  **Size**< T > & **operator\*=** (T2 c)
- template<typename T2 >  
  **Size**< T > & **operator/=** (T2 c)
- template<typename T2 >  
  **operator Size**< T2 > () const

### 4.29.1 Detailed Description

**template<typename T = double> class Size< T >**

Definition at line 11 of file include/densetrack/Size.h.

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

- include/densetrack/Size.h
- include/Size.h
- integration/densetrack/Size.h
- lib/densetrack/Size.h
- modules/densetrack/Size.h

## 4.30 Solver::SolutionInfo Struct Reference

### Public Attributes

- double **obj**
- double **rho**
- double **upper\_bound\_p**
- double **upper\_bound\_n**
- double **r**

### 4.30.1 Detailed Description

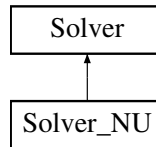
Definition at line 398 of file svm.cpp.

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

- svm.cpp

## 4.31 Solver Class Reference

Inheritance diagram for Solver:



### Classes

- struct [SolutionInfo](#)

### Public Member Functions

- void **Solve** (int l, const [QMatrix](#) &Q, const double \*p\_, const schar \*y\_, double \*alpha\_, double Cp, double Cn, double eps, [SolutionInfo](#) \*si, int shrinking)

### Protected Types

- enum { **LOWER\_BOUND**, **UPPER\_BOUND**, **FREE** }

### Protected Member Functions

- double **get\_C** (int i)
- void **update\_alpha\_status** (int i)
- bool **is\_upper\_bound** (int i)
- bool **is\_lower\_bound** (int i)
- bool **is\_free** (int i)
- void **swap\_index** (int i, int j)
- void **reconstruct\_gradient** ()
- virtual int **select\_working\_set** (int &i, int &j)
- virtual double **calculate\_rho** ()
- virtual void **do\_shrinking** ()

### Protected Attributes

- int **active\_size**
- schar \* **y**
- double \* **G**
- char \* **alpha\_status**
- double \* **alpha**
- const [QMatrix](#) \* **Q**
- const double \* **QD**
- double **eps**
- double **Cp**



- double **Cn**
- double \* **p**
- int \* **active\_set**
- double \* **G\_bar**
- int **l**
- bool **unshrink**

#### 4.31.1 Detailed Description

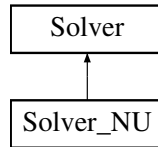
Definition at line 393 of file svm.cpp.

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

- svm.cpp

## 4.32 Solver\_NU Class Reference

Inheritance diagram for Solver\_NU:



### Public Member Functions

- void **Solve** (int l, const [QMatrix](#) &Q, const double \*p, const schar \*y, double \*alpha, double Cp, double Cn, double eps, [SolutionInfo](#) \*si, int shrinking)

#### 4.32.1 Detailed Description

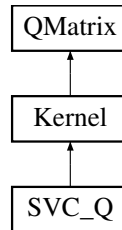
Definition at line 1009 of file svm.cpp.

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

- svm.cpp

## 4.33 SVC\_Q Class Reference

Inheritance diagram for SVC\_Q:



### Public Member Functions

- **SVC\_Q** (const [svm\\_problem](#) &prob, const [svm\\_parameter](#) &param, const `schar *`y\_)
- `Qfloat *`**get\_Q** (int i, int len) const
- `double *`**get\_QD** () const
- `void` **swap\_index** (int i, int j) const

#### 4.33.1 Detailed Description

Definition at line 1266 of file `svm.cpp`.

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

- `svm.cpp`

## 4.34 svm\_model Struct Reference

### Public Attributes

- struct [svm\\_parameter](#) **param**
- int **nr\_class**
- int **l**
- struct [svm\\_node](#) \*\* **SV**
- double \*\* **sv\_coef**
- double \* **rho**
- double \* **probA**
- double \* **probB**
- int \* **sv\_indices**
- int \* **label**
- int \* **nSV**
- int **free\_sv**

### 4.34.1 Detailed Description

Definition at line 52 of file include/svm.h.

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

- include/svm.h
- lib/libsvm-3.17/svm.h

## 4.35 svm\_node Struct Reference

### Public Attributes

- int **index**
- double **value**

### 4.35.1 Detailed Description

Definition at line 12 of file include/svm.h.

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

- include/svm.h
- lib/libsvm-3.17/svm.h

## 4.36 svm\_parameter Struct Reference

### Public Attributes

- int **svm\_type**
- int **kernel\_type**
- int **degree**
- double **gamma**
- double **coef0**
- double **cache\_size**
- double **eps**
- double **C**
- int **nr\_weight**
- int \* **weight\_label**
- double \* **weight**
- double **nu**
- double **p**
- int **shrinking**
- int **probability**

### 4.36.1 Detailed Description

Definition at line 28 of file include/svm.h.

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

- include/svm.h
- lib/libsvm-3.17/svm.h

## 4.37 svm\_problem Struct Reference

### Public Attributes

- int l
- double \* y
- struct [svm\\_node](#) \*\* x

#### 4.37.1 Detailed Description

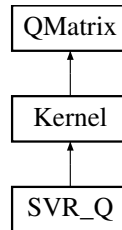
Definition at line 18 of file include/svm.h.

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

- include/svm.h
- lib/libsvm-3.17/svm.h

## 4.38 SVR\_Q Class Reference

Inheritance diagram for SVR\_Q:



### Public Member Functions

- **SVR\_Q** (const [svm\\_problem](#) &prob, const [svm\\_parameter](#) &param)
- void **swap\_index** (int i, int j) const
- Qfloat \* **get\_Q** (int i, int len) const
- double \* **get\_QD** () const

### 4.38.1 Detailed Description

Definition at line 1362 of file svm.cpp.

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

- svm.cpp



## 4.39 Tactil Class Reference

### Public Member Functions

- **Tactil** (boost::shared\_ptr< AL::ALBroker > broker, const std::string &name)
- virtual void **init** ()
- void **onFrontHeadTouched** ()
- void **helloAnimation** ()

#### 4.39.1 Detailed Description

Definition at line 32 of file tactil.h.

#### 4.39.2 Member Function Documentation

##### 4.39.2.1 void Tactil::init () [virtual]

Overloading ALModule::init(). This is called right after the module has been loaded

Definition at line 25 of file modules/libintegration/tactil.cpp.

##### 4.39.2.2 void Tactil::onFrontHeadTouched ()

This method will be called every time the event FrontTactilTouched is raised.

Definition at line 50 of file modules/libintegration/tactil.cpp.

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

- tactil.h
- modules/libintegration/tactil.cpp
- src/tactil.cpp

## 4.40 temps\_exec Struct Reference

### Public Attributes

- int **debut**
- int **fin**
- sTms **sdebut**
- sTms **sfin**

### 4.40.1 Detailed Description

Definition at line 18 of file integration/main.c.

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

- integration/main.c

## 4.41 Track Class Reference

### Public Member Functions

- **Track** (int maxNPoints\_)
- void **addPointDesc** (const [PointDesc](#) &point)
- **Track** (int maxNPoints\_)
- void **addPointDesc** (const [PointDesc](#) &point)
- **Track** (int maxNPoints\_)
- void **addPointDesc** (const [PointDesc](#) &point)
- **Track** (int maxNPoints\_)
- void **addPointDesc** (const [PointDesc](#) &point)

### Public Attributes

- std::list< [PointDesc](#) > **pointDescs**
- int **maxNPoints**

#### 4.41.1 Detailed Description

Definition at line 71 of file include/denseTrack.h.

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

- include/denseTrack.h
- naodensetrack.h
- DenseTrack.h
- modules/densetrack/denseTrack.h

## 4.42 TrackerInfo Struct Reference

### Public Attributes

- int **trackLength**
- int **initGap**

### 4.42.1 Detailed Description

Definition at line 24 of file include/denseTrack.h.

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

- include/denseTrack.h
- naodensetrack.h
- DenseTrack.h
- modules/densetrack/denseTrack.h

# Chapter 5

## File Documentation

### 5.1 naodensetrack.cpp File Reference

Set of function permitting to extract dense points and their trajectories.

```
#include "naodensetrack.h"
```

#### Functions

- CvScalar **getRect** (const CvPoint2D32f point, const CvSize size, const [DescInfo](#) descInfo)
- void **BuildDescMat** (const IplImage \*xComp, const IplImage \*yComp, [DescMat](#) \*descMat, const [DescInfo](#) descInfo)
- std::vector< float > **getDesc** (const [DescMat](#) \*descMat, CvScalar rect, [DescInfo](#) descInfo, float epsilon)
- void **HogComp** (IplImage \*img, [DescMat](#) \*descMat, [DescInfo](#) descInfo)
- void **HofComp** (IplImage \*flow, [DescMat](#) \*descMat, [DescInfo](#) descInfo)
- void **MbhComp** (IplImage \*flow, [DescMat](#) \*descMatX, [DescMat](#) \*descMatY, [DescInfo](#) descInfo)
- void **OpticalFlowTracker** (IplImage \*flow, std::vector< CvPoint2D32f > &points\_in, std::vector< CvPoint2D32f > &points\_out, std::vector< int > &status)
- int **isValid** (std::vector< CvPoint2D32f > &track, float &mean\_x, float &mean\_y, float &var\_x, float &var\_y, float &length, float min\_var, float max\_var, float max\_dis)
- void **cvDenseSample** (IplImage \*grey, IplImage \*eig, std::vector< CvPoint2D32f > &points, const double quality, const double min\_distance)
- void **cvDenseSample** (IplImage \*grey, IplImage \*eig, std::vector< CvPoint2D32f > &points\_in, std::vector< CvPoint2D32f > &points\_out, const double quality, const double min\_distance)
- void **InitTrackerInfo** ([TrackerInfo](#) \*tracker, int track\_length, int init\_gap)
- [DescMat](#) \* **InitDescMat** (int height, int width, int nBins)
- void **ReleDescMat** ([DescMat](#) \*descMat)
- void **InitDescInfo** ([DescInfo](#) \*descInfo, int nBins, int flag, int orientation, int size, int nxy\_cell, int nt\_cell, float min\_flow)
- void **usage** ()
- int **extractSTIPs** (std::string video, int dim, int maxPts, [KMdata](#) \*dataPts)

*Permits to extract STIPs from a video .avi. It save the HOG and HOG of the trajectories in the object [KMdata](#).*

### 5.1.1 Detailed Description

Set of function permitting to extract dense points and their trajectories.

#### Author

LEAR

#### Date

05/07/2013

Definition in file [naodensetrack.cpp](#).

### 5.1.2 Function Documentation

#### 5.1.2.1 `int extractSTIPs (std::string video, int dim, int maxPts, KMdata * dataPts)`

Permits to extract STIPs from a video .avi. It save the HOG and HOG of the trajectories in the object [KMdata](#).

#### Parameters

- ← *stip* Name of the video.
- ← *dim* STIPs dimension.
- ← *maxPts* Maximum number of points we want to use.
- *dataPts* The object in which we save the STIPs.

#### Returns

Number of points extracted.

Definition at line 491 of file [naodensetrack.cpp](#).

## 5.2 naokmeans.cpp File Reference

Set of functions permitting to execute KMeans algorithms using [KMlocal](#) classes.

```
#include "naokmeans.h"
#include <time.h>
```

### Functions

- int [importSTIPs](#) (std::string stip, int dim, int maxPts, [KMdata](#) \*dataPts)  
*STIPs importation function in the format 1 point = 1 line. Each dimension are separated from one space (" ").*
- void [exportSTIPs](#) (std::string stip, int dim, const [KMdata](#) &dataPts)
- void [exportCenters](#) (std::string centers, int dim, int k, [KMfilterCenters](#) ctrs)  
*Export function to save [KMfilterCenters](#) in a file. One line corresponds to one point with dim value (separated from one space " ").*
- void [importCenters](#) (std::string centers, int dim, int k, [KMfilterCenters](#) \*ctrs)  
*Importation function saving external centers in the [KMfilterCenters](#) object. One line corresponds to one centers with its values (separated from one space " ").*
- void [kmIvanAlgorithm](#) (int ic, int dim, const [KMdata](#) &dataPts, int k, [KMfilterCenters](#) &ctrs)  
*This is an optimized KMeans algorithm. Ivan's algorithm uses basic KMeans algorithm (here the Lloyd's one) and the idea was to initialize centers intelligently.*
- void [createTrainingMeans](#) (std::string stipFile, int dim, int maxPts, int k, std::string meansFile)  
*Import HOG and HOF from a file and compute KMeans algorithm to create the file training.means.*

### 5.2.1 Detailed Description

Set of functions permitting to execute KMeans algorithms using [KMlocal](#) classes.

#### Author

Fabien ROUALDES (institut Mines-Télécom)

#### Date

02/07/2013

Definition in file [naokmeans.cpp](#).

### 5.2.2 Function Documentation

#### 5.2.2.1 void [createTrainingMeans](#) (std::string *stipFile*, int *dim*, int *maxPts*, int *k*, std::string *meansFile*)

Import HOG and HOF from a file and compute KMeans algorithm to create the file training.means.

**Parameters**

- ← *stipFile* The file containing the STIPs.
- ← *dim* Points and centers's dimension.
- ← *maxPts* The maximum number of data we can compute
- ← *k* The number of centers
- *meansFile* The file in wich we will save the KMeans centers.

Definition at line 262 of file naokmeans.cpp.

**5.2.2.2 void exportCenters (std::string centers, int dim, int k, KMfilterCenters ctrs)**

Export function to save [KMfilterCenters](#) in a file. One line corresponds to one point with dim value (separated from one space " ").

**Parameters**

- ← *centers* Name of the file which will be containing dimensions of each centers.
- ← *dim* Center's dimension.
- ← *k* Number of centers.
- ← *ctrs* The centers.

Definition at line 83 of file naokmeans.cpp.

**5.2.2.3 void importCenters (std::string centers, int dim, int k, KMfilterCenters \* ctrs)**

Importation function saving external centers in the [KMfilterCenters](#) object. One line corresponds to one centers with its values (separated from one space " ").

**Parameters**

- ← *centers* Name of the file which will be containing dimensions of each centers.
- ← *dim* Center's dimension.
- ← *k* Number of centers.
- *ctrs* The centers.

Definition at line 109 of file naokmeans.cpp.

**5.2.2.4 int importSTIPs (std::string stip, int dim, int maxPts, KMdata \* dataPts)**

STIPs importation function in the format 1 point = 1 line. Each dimension are separated from one space (" ").

**Parameters**

- ← *stip* Name of the file containing the STIPs.
- ← *dim* The STIPs dimension.
- ← *maxPts* The maximum number of points you want to import.
- *dataPts* The [KMlocal](#) object which will be containing STIPs.



**Returns**

Number of points imported.

Definition at line 23 of file naokmeans.cpp.

**5.2.2.5 void kmIvanAlgorithm (int *ic*, int *dim*, const KMdata & *dataPts*, int *k*, KMfilterCenters & *ctrs*)**

This is an optimized KMeans algorithm. Ivan's algorithm uses basic KMeans algorithm (here the Lloyd's one) and the idea was to initialize centers intelligently.

**Parameters**

← *ic* The iteration coefficient will determine the number of iterations in each phases.

← *dim* Points and centers's dimension.

← *dataPts* The data we want to compute the centers.

← *k* The number of centers.

→ *ctrs* The centers.

The Ivan's algorithm is divided into 3 phases. The first phase is executed on 25 per cent of the data (randomly sampled). To begin, the centers are randomly generated. Then  $ic * 4$  iterations of a KMeans algorithm are executed. During the second part we cluster 50 per cent of the data using the older centroids. This step is computed  $ic * 2$  times. Finally, we make  $ic * 1$  iteration on all the data.

Definition at line 149 of file naokmeans.cpp.

## 5.3 naokmeans.h File Reference

```
#include <cstdlib>
#include <iostream>
#include <string.h>
#include <fstream>
#include "KMlocal.h"
```

### Functions

- int [importSTIPs](#) (std::string stip, int dim, int maxPts, [KMdata](#) \*dataPts)  
*STIPs importation function in the format 1 point = 1 line. Each dimension are separated from one space (" ").*
- void [exportSTIPs](#) (std::string stip, int dim, const [KMdata](#) &dataPts)
- void [importCenters](#) (std::string centers, int dim, int k, [KMfilterCenters](#) \*ctrs)  
*Importation function saving external centers in the [KMfilterCenters](#) object. One line corresponds to one centers with its values (separated from one space " ").*
- void [exportCenters](#) (std::string centers, int dim, int k, [KMfilterCenters](#) ctrs)  
*Export function to save [KMfilterCenters](#) in a file. One line corresponds to one point with dim value (separated from one space " ").*
- void [kmIvanAlgorithm](#) (int ic, int dim, const [KMdata](#) &dataPts, int k, [KMfilterCenters](#) &ctrs)  
*This is an optimized KMeans algorithm. Ivan's algorithm uses basic KMeans algorithm (here the Lloyd's one) and the idea was to initialize centers intelligently.*
- void [createTrainingMeans](#) (std::string stipFile, int dim, int maxPts, int k, std::string meansFile)  
*Import HOG and HOF from a file and compute KMeans algorithm to create the file training.means.*

### 5.3.1 Detailed Description

#### Author

Fabien ROUALDES (institut Mines-Télécom)

#### Date

02/07/2013 Set of function permitting to execute KMeans algorithms

Definition in file [naokmeans.h](#).

### 5.3.2 Function Documentation

#### 5.3.2.1 void [createTrainingMeans](#) (std::string *stipFile*, int *dim*, int *maxPts*, int *k*, std::string *meansFile*)

Import HOG and HOF from a file and compute KMeans algorithm to create the file training.means.

**Parameters**

- ← *stipFile* The file containing the STIPs.
- ← *dim* Points and centers's dimension.
- ← *maxPts* The maximum number of data we can compute
- ← *k* The number of centers
- *meansFile* The file in wich we will save the KMeans centers.

Definition at line 262 of file naokmeans.cpp.

**5.3.2.2 void exportCenters (std::string centers, int dim, int k, KMfilterCenters ctrs)**

Export function to save [KMfilterCenters](#) in a file. One line corresponds to one point with dim value (separated from one space " ").

**Parameters**

- ← *centers* Name of the file which will be containing dimensions of each centers.
- ← *dim* Center's dimension.
- ← *k* Number of centers.
- ← *ctrs* The centers.

Definition at line 83 of file naokmeans.cpp.

**5.3.2.3 void importCenters (std::string centers, int dim, int k, KMfilterCenters \* ctrs)**

Importation function saving external centers in the [KMfilterCenters](#) object. One line corresponds to one centers with its values (separated from one space " ").

**Parameters**

- ← *centers* Name of the file which will be containing dimensions of each centers.
- ← *dim* Center's dimension.
- ← *k* Number of centers.
- *ctrs* The centers.

Definition at line 109 of file naokmeans.cpp.

**5.3.2.4 int importSTIPs (std::string stip, int dim, int maxPts, KMdata \* dataPts)**

STIPs importation function in the format 1 point = 1 line. Each dimension are separated from one space (" ").

**Parameters**

- ← *stip* Name of the file containing the STIPs.
- ← *dim* The STIPs dimension.
- ← *maxPts* The maximum number of points you want to import.
- *dataPts* The [KMlocal](#) object which will be containing STIPs.

**Returns**

Number of points imported.

Definition at line 23 of file naokmeans.cpp.

**5.3.2.5 void kmIvanAlgorithm (int *ic*, int *dim*, const KMdata & *dataPts*, int *k*, KMfilterCenters & *ctrs*)**

This is an optimized KMeans algorithm. Ivan's algorithm uses basic KMeans algorithm (here the Lloyd's one) and the idea was to initialize centers intelligently.

**Parameters**

- ← *ic* The iteration coefficient will determine the number of iterations in each phases.
- ← *dim* Points and centers's dimension.
- ← *dataPts* The data we want to compute the centers.
- ← *k* The number of centers.
- *ctrs* The centers.

The Ivan's algorithm is divided into 3 phases. The first phase is executed on 25 per cent of the data (randomly sampled). To begin, the centers are randomly generated. Then  $ic * 4$  iterations of a KMeans algorithm are executed. During the second part we cluster 50 per cent of the data using the older centroids. This step is computed  $ic * 2$  times. Finally, we make  $ic * 1$  iteration on all the data.

Definition at line 149 of file naokmeans.cpp.

## 5.4 naomngt.cpp File Reference

Set of functions permitting to manage the activity recognition BDD of Bag Of Words.

```
#include "naomngt.h"
```

### Functions

- void [listBdds](#) ()  
*List BDDs present in the global database.*
- void [listActivities](#) (std::string bdd)  
*List activities present in the specified database.*
- int [mapActivities](#) (std::string path2bdd, [activitiesMap](#) \*\*am)  
*Fills the object [activitiesMap](#) which contain the equivalence Label-Activity.*
- int [nbOfFiles](#) (std::string path)  
*Counts the number of files in a folder.*
- bool [fileExist](#) (std::string file, std::string folder)  
*Checks if the file name does not exist.*
- void [addVideos](#) (std::string bddName, std::string activity, int nbVideos, std::string \*videoPaths, int dim, int maxPts)  
*Adds a new video in the choosen activity of the specified BDD.*
- std::string [inttostring](#) (int int2str)  
*Converts an int into a string.*
- void [trainBdd](#) (std::string bddName, int dim, int maxPts, int k)  
*Trains the specified BDD.*
- void [addLabel](#) (int label, std::string file, int k)  
*Changes the label of the Bag Of Words.*
- void [addActivity](#) (std::string activityName, std::string bddName)  
*Creates a new activity in the specified BDD.*
- void [deleteActivity](#) (std::string activityName, std::string bddName)  
*Deletes an existant activity in the specified BDD.*
- void [addBdd](#) (std::string bddName)  
*Creates a new BDD.*
- void [deleteBdd](#) (std::string bddName)  
*Deletes a BDD.*
- void [emptyFolder](#) (std::string folder)  
*Deletes all files present in the folder.*
- void [refreshBdd](#) (std::string bddName, int dim, int maxPts)

### 5.4.1 Detailed Description

Set of functions permitting to manage the activity recognition BDD of Bag Of Words.

#### Author

Fabien ROUALDES (institut Mines-Télécom)

#### Date

17/07/2013

Definition in file [naomngt.cpp](#).

### 5.4.2 Function Documentation

#### 5.4.2.1 void addActivity (std::string *activityName*, std::string *bddName*)

Creates a new activity in the specified BDD.

##### Parameters

← *activityName* The name of the new activity.

← *bddName* The name of the BDD.

Definition at line 437 of file naomngt.cpp.

#### 5.4.2.2 void addBdd (std::string *bddName*)

Creates a new BDD.

##### Parameters

← *bddName* The name of the BDD we want to create.

Definition at line 546 of file naomngt.cpp.

#### 5.4.2.3 void addLabel (int *label*, std::string *file*, int *k*)

Changes the label of the Bag Of Words.

##### Parameters

← *label* The label.

← *file* The file containing the Bag Of Words.

← *k* The dimension of the Bag Of Words.

Definition at line 381 of file naomngt.cpp.

**5.4.2.4 void addVideos (std::string *bddName*, std::string *activity*, int *nbVideos*, std::string \* *videoPaths*, int *dim*, int *maxPts*)**

Adds a new video in the choosen activity of the specified BDD.

**Parameters**

- ← *bddName* The name of the BDD.
- ← *activity* The name of the activity.
- ← *nbVideos* The number of videos we want to add.
- ← *videoPaths* The different paths to the videos.
- ← *dim* The dimension of the HOG and HOF.
- ← *maxPts* The maximum vectors we want to compute.

Definition at line 156 of file naomngt.cpp.

**5.4.2.5 void deleteActivity (std::string *activityName*, std::string *bddName*)**

Deletes an existant activity in the specified BDD.

**Parameters**

- ← *activityName* The name of the activity to delete.
- ← *bddName* The name of the BDD.

Definition at line 485 of file naomngt.cpp.

**5.4.2.6 void deleteBdd (std::string *bddName*)**

Deletes a BDD.

**Parameters**

- ← *bddName* The name of the bdd.

Definition at line 583 of file naomngt.cpp.

**5.4.2.7 void emptyFolder (std::string *folder*)**

Deletes all files present in the folder.

**Parameters**

- ← *folder* The path to the folder.

Definition at line 605 of file naomngt.cpp.

#### 5.4.2.8 bool fileExist (std::string *file*, std::string *folder*)

Checks if the file name does not exist.

##### Parameters

← *file* The path to the file.

← *folder* The path to the folder.

##### Returns

True or false.

Definition at line 126 of file naomngt.cpp.

#### 5.4.2.9 std::string inttostring (int *int2str*)

Converts an int into a string.

##### Parameters

← *int2str* The int to convert.

##### Returns

The string converted.

Definition at line 206 of file naomngt.cpp.

#### 5.4.2.10 void listActivities (std::string *bdd*)

List activities present in the specified database.

##### Parameters

← *bdd* The name of the bdd.

Definition at line 34 of file naomngt.cpp.

#### 5.4.2.11 int mapActivities (std::string *path2bdd*, activitiesMap \*\* *am*)

Fills the object [activitiesMap](#) which contain the equivalence Label-Activity.

##### Parameters

← *path2bdd* The path to the BDD

↔ *am* A pointer to an object [activitiesMap](#).

##### Returns

The number of activities.

Definition at line 53 of file naomngt.cpp.



#### 5.4.2.12 int nbOffFiles (std::string *path*)

Counts the number of files in a folder.

##### Parameters

← *path* The path to the folder.

##### Returns

The number of files.

Definition at line 101 of file naomngt.cpp.

#### 5.4.2.13 void trainBdd (std::string *bddName*, int *dim*, int *maxPts*, int *k*)

Trains the specified BDD.

##### Parameters

← *bddName* The name of the BDD.

← *dim* The dimension of the STIPs.

← *maxPts* The maximum number of points we want to compute.

← *k* The number of cluster (means).

Definition at line 225 of file naomngt.cpp.

## 5.5 naomngt.h File Reference

```
#include <stdlib.h>
#include <dirent.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <iostream>
#include <fstream>
#include <sstream>
#include <vector>
#include <string>
#include "naokmeans.h"
#include "naosvm.h"
#include "naodensetrack.h"
```

### Classes

- class [activitiesMap](#)  
*Correspondance between a label and an activity.*

### Functions

- void [listBdds](#) ()  
*List BDDs present in the global database.*
- void [listActivities](#) (std::string bdd)  
*List activities present in the specified database.*
- int [mapActivities](#) (std::string path2bdd, [activitiesMap](#) \*\*am)  
*Fills the object [activitiesMap](#) which contain the equivalence Label-Activity.*
- int [nbOfFiles](#) (std::string path)  
*Counts the number of files in a folder.*
- bool [fileExist](#) (std::string file, std::string folder)  
*Checks if the file name does not exist.*
- void [addVideos](#) (std::string bddName, std::string activity, int nbVideos, std::string \*videoPaths, int dim, int maxPts)  
*Adds a new video in the choosen activity of the specified BDD.*
- std::string [inttostring](#) (int int2str)  
*Converts an int into a string.*

- void [trainBdd](#) (std::string bddName, int dim, int maxPts, int k)  
*Trains the specified BDD.*
- void [addLabel](#) (int label, std::string file, int k)  
*Changes the label of the Bag Of Words.*
- void [addBdd](#) (std::string bddName)  
*Creates a new BDD.*
- void [addActivity](#) (std::string activityName, std::string bddName)  
*Creates a new activity in the specified BDD.*
- void [deleteBdd](#) (std::string bddName)  
*Deletes a BDD.*
- void [deleteActivity](#) (std::string activityName, std::string bddName)  
*Deletes an existant activity in the specified BDD.*
- void [emptyFolder](#) (std::string folder)  
*Deletes all files present in the folder.*
- void [refreshBdd](#) (std::string bddName, int dim, int maxPts)

### 5.5.1 Detailed Description

#### Author

Fabien ROUALDES (Institut Mines-Télécom)

#### Date

17/07/2013

Definition in file [naomngt.h](#).

### 5.5.2 Function Documentation

#### 5.5.2.1 void addActivity (std::string activityName, std::string bddName)

Creates a new activity in the specified BDD.

#### Parameters

- ← **activityName** The name of the new activity.
- ← **bddName** The name of the BDD.

Definition at line 437 of file naomngt.cpp.

### 5.5.2.2 void addBdd (std::string *bddName*)

Creates a new BDD.

#### Parameters

← *bddName* The name of the BDD we want to create.

Definition at line 546 of file naomngt.cpp.

### 5.5.2.3 void addLabel (int *label*, std::string *file*, int *k*)

Changes the label of the Bag Of Words.

#### Parameters

← *label* The label.

← *file* The file containing the Bag Of Words.

← *k* The dimension of the Bag Of Words.

Definition at line 381 of file naomngt.cpp.

### 5.5.2.4 void addVideos (std::string *bddName*, std::string *activity*, int *nbVideos*, std::string \* *videoPaths*, int *dim*, int *maxPts*)

Adds a new video in the choosen activity of the specified BDD.

#### Parameters

← *bddName* The name of the BDD.

← *activity* The name of the activity.

← *nbVideos* The number of videos we want to add.

← *videoPaths* The different paths to the videos.

← *dim* The dimension of the HOG and HOF.

← *maxPts* The maximum vectors we want to compute.

Definition at line 156 of file naomngt.cpp.

### 5.5.2.5 void deleteActivity (std::string *activityName*, std::string *bddName*)

Deletes an existant activity in the specified BDD.

#### Parameters

← *activityName* The name of the activity to delete.

← *bddName* The name of the BDD.

Definition at line 485 of file naomngt.cpp.

#### 5.5.2.6 void deleteBdd (std::string *bddName*)

Deletes a BDD.

##### Parameters

← *bddName* The name of the bdd.

Definition at line 583 of file naomngt.cpp.

#### 5.5.2.7 void emptyFolder (std::string *folder*)

Deletes all files present in the folder.

##### Parameters

← *folder* The path to the folder.

Definition at line 605 of file naomngt.cpp.

#### 5.5.2.8 bool fileExist (std::string *file*, std::string *folder*)

Checks if the file name does not exist.

##### Parameters

← *file* The path to the file.

← *folder* The path to the folder.

##### Returns

True or false.

Definition at line 126 of file naomngt.cpp.

#### 5.5.2.9 std::string inttostring (int *int2str*)

Converts an int into a string.

##### Parameters

← *int2str* The int to convert.

##### Returns

The string converted.

Definition at line 206 of file naomngt.cpp.

**5.5.2.10 void listActivities (std::string *bdd*)**

List activities present in the specified database.

**Parameters**

← *bdd* The name of the bdd.

Definition at line 34 of file naomngt.cpp.

**5.5.2.11 int mapActivities (std::string *path2bdd*, activitiesMap \*\* *am*)**

Fills the object [activitiesMap](#) which contain the equivalence Label-Activity.

**Parameters**

← *path2bdd* The path to the BDD

↔ *am* A pointer to an object [activitiesMap](#).

**Returns**

The number of activities.

Definition at line 53 of file naomngt.cpp.

**5.5.2.12 int nbOffFiles (std::string *path*)**

Counts the number of files in a folder.

**Parameters**

← *path* The path to the folder.

**Returns**

The number of files.

Definition at line 101 of file naomngt.cpp.

**5.5.2.13 void trainBdd (std::string *bddName*, int *dim*, int *maxPts*, int *k*)**

Trains the specified BDD.

**Parameters**

← *bddName* The name of the BDD.

← *dim* The dimension of the STIPs.

← *maxPts* The maximum number of points we want to compute.

← *k* The number of cluster (means).

Definition at line 225 of file naomngt.cpp.

## 5.6 naosvm.cpp File Reference

Set of functions permitting to import/ predict a svm problem, import/create a svm model.

```
#include "naosvm.h"
```

### Functions

- struct [svm\\_problem](#) **importProblem** (std::string file, int k)
- void **exportProblem** (struct [svm\\_problem](#) svmProblem, std::string file)
- void **exportProblemZero** (struct [svm\\_problem](#) svmProblem, std::string file, int k)
- struct [svm\\_problem](#) **computeBOW** (int label, const [KMdata](#) &dataPts, [KMfilterCenters](#) &ctrs)
- void [printProblem](#) (struct [svm\\_problem](#) svmProblem)  
*It permits to print the SVM problem in the standard output.*
- int [nrOfLines](#) (std::string filename)  
*A function returning the number of lines (which correspond to the number of activities).*
- void [printProbability](#) (struct [svm\\_model](#) \*pModel, struct [svm\\_node](#) \*nodes)  
*Print for each labels the probability of the activity (stored in the SVM node structure).*
- struct [svm\\_model](#) \* **createSvmModel** (std::string bowFile, int k)

### 5.6.1 Detailed Description

Set of functions permitting to import/ predict a svm problem, import/create a svm model.

#### Author

Fabien ROUALDES (institut Mines-Télécom)

#### Date

17/07/2013

Definition in file [naosvm.cpp](#).

### 5.6.2 Function Documentation

#### 5.6.2.1 int nrOfLines (std::string filename)

A function returning the number of lines (which correspond to the number of activities).

#### Parameters

← *fileName* The file we want to count the number of lines.

#### Returns

The number of lines of the file.

Definition at line 282 of file naosvm.cpp.

**5.6.2.2 void printProbability (struct svm\_model \* *pModel*, struct svm\_node \* *nodes*)**

Print for each labels the probability of the activity (stored in the SVM node structure).

**Parameters**

- ← *pModel* A pointer to the SVM model.
- ← *nodes* The activity stored in SVM nodes.

Definition at line 304 of file naosvm.cpp.

**5.6.2.3 void printProblem (struct svm\_problem *svmProblem*)**

It permits to print the SVM problem in the standard output.

**Parameters**

- ← *svmProblem* It is the structure containing the SVM problem.

Definition at line 245 of file naosvm.cpp.



## 5.7 naosvm.h File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <iostream>
#include <fstream>
#include <sstream>
#include <vector>
#include <string>
#include "svm.h"
#include "KMlocal.h"
```

### Functions

- struct [svm\\_model](#) \* **createSvmModel** (std::string bowFile, int k)
- void **printProbability** (struct [svm\\_model](#) \*pModel, struct [svm\\_node](#) \*nodes)  
*Print for each labels the probability of the activity (stored in the SVM node structure).*
- struct [svm\\_problem](#) **importProblem** (std::string file, int k)
- struct [svm\\_problem](#) **computeBOW** (int label, const [KMdata](#) &dataPts, [KMfilterCenters](#) &ctrs)
- void **exportProblem** (struct [svm\\_problem](#) svmProblem, std::string file)
- void **exportProblemZero** (struct [svm\\_problem](#) svmProblem, std::string file, int k)
- void **printProblem** (struct [svm\\_problem](#) svmProblem)  
*It permits to print the SVM problem in the standard output.*
- int **nrOfLines** (std::string filename)  
*A function returning the number of lines (which correspond to the number of activities).*

### 5.7.1 Detailed Description

#### Author

Fabien ROUALDES (institut Mines-Télécom)

#### Date

09/07/2013 Set of function permitting to import/ predict a svm problem, import/create a svm model

Definition in file [naosvm.h](#).

### 5.7.2 Function Documentation

#### 5.7.2.1 int nrOfLines (std::string filename)

A function returning the number of lines (which correspond to the number of activities).

**Parameters**

← *fileName* The file we want to count the number of lines.

**Returns**

The number of lines of the file.

Definition at line 282 of file naosvm.cpp.

**5.7.2.2 void printProbability (struct svm\_model \* *pModel*, struct svm\_node \* *nodes*)**

Print for each labels the probability of the activity (stored in the SVM node structure).

**Parameters**

← *pModel* A pointer to the SVM model.

← *nodes* The activity stored in SVM nodes.

Definition at line 304 of file naosvm.cpp.

**5.7.2.3 void printProblem (struct svm\_problem *svmProblem*)**

It permits to print the SVM problem in the standard output.

**Parameters**

← *svmProblem* It is the structure containing the SVM problem.

Definition at line 245 of file naosvm.cpp.

# Index

- activitiesMap, [9](#)
- addActivity
  - [naomngt.cpp](#), [86](#)
  - [naomngt.h](#), [91](#)
- addBdd
  - [naomngt.cpp](#), [86](#)
  - [naomngt.h](#), [91](#)
- addLabel
  - [naomngt.cpp](#), [86](#)
  - [naomngt.h](#), [92](#)
- addVideos
  - [naomngt.cpp](#), [86](#)
  - [naomngt.h](#), [92](#)
- Box, [10](#)
- Cache, [20](#)
- createTrainingMeans
  - [naokmeans.cpp](#), [79](#)
  - [naokmeans.h](#), [82](#)
- decision\_function, [21](#)
- deleteActivity
  - [naomngt.cpp](#), [87](#)
  - [naomngt.h](#), [92](#)
- deleteBdd
  - [naomngt.cpp](#), [87](#)
  - [naomngt.h](#), [92](#)
- DescInfo, [22](#)
- DescMat, [23](#)
- emptyFolder
  - [naomngt.cpp](#), [87](#)
  - [naomngt.h](#), [93](#)
- exec\_time, [24](#)
- exportCenters
  - [naokmeans.cpp](#), [80](#)
  - [naokmeans.h](#), [83](#)
- extractSTIPs
  - [naodensetrack.cpp](#), [78](#)
- fileExist
  - [naomngt.cpp](#), [87](#)
  - [naomngt.h](#), [93](#)
- getImage
  - [IplImagePyramid](#), [28](#)
- getIndex
  - [IplImagePyramid](#), [29](#)
- importCenters
  - [naokmeans.cpp](#), [80](#)
  - [naokmeans.h](#), [83](#)
- importSTIPs
  - [naokmeans.cpp](#), [80](#)
  - [naokmeans.h](#), [83](#)
- init
  - [Tactil](#), [73](#)
- inttostring
  - [naomngt.cpp](#), [88](#)
  - [naomngt.h](#), [93](#)
- IplImagePyramid, [25](#)
  - [getImage](#), [28](#)
  - [getIndex](#), [29](#)
  - [IplImagePyramid](#), [27](#), [28](#)
  - [rebuild](#), [29](#), [30](#)
- IplImageWrapper, [31](#)
- KCleaf, [34](#)
- KCnode, [35](#)
- KCsplrit, [37](#)
- KCtree, [38](#)
- Kernel, [39](#)
- KMcenters, [40](#)
- KMdata, [42](#)
- KMfilterCenters, [43](#)
- kmIvanAlgorithm
  - [naokmeans.cpp](#), [81](#)
  - [naokmeans.h](#), [84](#)
- KMlocal, [45](#)
- KMlocalEZ\_Hybrid, [47](#)
- KMlocalHybrid, [48](#)
- KMlocalLloyds, [50](#)
- KMlocalSwap, [51](#)
- KMorthRect, [52](#)
- KMterm, [53](#)
- listActivities
  - [naomngt.cpp](#), [88](#)
  - [naomngt.h](#), [93](#)
- mapActivities

- naomngt.cpp, 88
- naomngt.h, 94
- naodensetrack.cpp, 77
  - extractSTIPs, 78
- naokmeans.cpp, 79
  - createTrainingMeans, 79
  - exportCenters, 80
  - importCenters, 80
  - importSTIPs, 80
  - kmIvanAlgorithm, 81
- naokmeans.h, 82
  - createTrainingMeans, 82
  - exportCenters, 83
  - importCenters, 83
  - importSTIPs, 83
  - kmIvanAlgorithm, 84
- naomngt.cpp, 85
  - addActivity, 86
  - addBdd, 86
  - addLabel, 86
  - addVideos, 86
  - deleteActivity, 87
  - deleteBdd, 87
  - emptyFolder, 87
  - fileExist, 87
  - inttostring, 88
  - listActivities, 88
  - mapActivities, 88
  - nbOfFiles, 88
  - trainBdd, 89
- naomngt.h, 90
  - addActivity, 91
  - addBdd, 91
  - addLabel, 92
  - addVideos, 92
  - deleteActivity, 92
  - deleteBdd, 92
  - emptyFolder, 93
  - fileExist, 93
  - inttostring, 93
  - listActivities, 93
  - mapActivities, 94
  - nbOfFiles, 94
  - trainBdd, 94
- naosvm.cpp, 95
  - nrOfLines, 95
  - printProbability, 95
  - printProblem, 96
- naosvm.h, 97
  - nrOfLines, 97
  - printProbability, 98
  - printProblem, 98
- nbOfFiles
  - naomngt.cpp, 88
  - naomngt.h, 94
- nrOfLines
  - naosvm.cpp, 95
  - naosvm.h, 97
- ONE\_CLASS\_Q, 55
- onFrontHeadTouched
  - Tactil, 73
- Point, 56
- PointDesc, 58
- printProbability
  - naosvm.cpp, 95
  - naosvm.h, 98
- printProblem
  - naosvm.cpp, 96
  - naosvm.h, 98
- QMatrix, 59
- rebuild
  - IplImagePyramid, 29, 30
- Size, 60
- Solver, 64
- Solver::SolutionInfo, 63
- Solver\_NU, 66
- SVC\_Q, 67
- svm\_model, 68
- svm\_node, 69
- svm\_parameter, 70
- svm\_problem, 71
- SVR\_Q, 72
- Tactil, 73
  - init, 73
  - onFrontHeadTouched, 73
- temps\_exec, 74
- Track, 75
- TrackerInfo, 76
- trainBdd
  - naomngt.cpp, 89
  - naomngt.h, 94