VSS-Vision

Generated by Doxygen 1.8.11

Contents

1 Namespace Index **1**

1.1 Namespace List 1

2 Hierarchical Index **3**

2.1 Class Hierarchy 3

3 Class Index **5**

3.1 Class List 5

4 Namespace Documentation **7**

4.1 common Namespace Reference 7

4.1.1 Detailed Description 9

4.1.2 Function Documentation 9

4.1.2.1 angulation(Point, Point) 9

5 Class Documentation **11**

5.1 vss_state::Ball_State Class Reference 11

5.2 common::btVector3 Struct Reference 13

5.2.1 Detailed Description 13

5.3 calibration Class Reference 13

5.3.1 Detailed Description 16

5.3.2 Constructor & Destructor Documentation 16

5.3.2.1 calibration() 16

5.3.3 Member Function Documentation 16

5.3.3.1 alloc_calibration(Calibration *) 16

5.3.3.2 alloc_label_input(QCustomLabel ∗) . . . 16
5.3.3.3 applyFilters() . . . 17
5.3.3.4 paint_output() . . . 17
5.3.3.5 run() . . . 17
5.3.3.6 set_vision_reception(bool) . . . 18
5.3.3.7 zoom() . . . 18
5.4 common::Calibration Struct Reference . . . 18
5.4.1 Detailed Description . . . 19
5.5 common::ExecConfiguration Struct Reference . . . 19
5.5.1 Detailed Description . . . 20
5.6 vss_command::Global_Commands Class Reference . . . 20
5.7 vss_state::Global_State Class Reference . . . 22
5.8 Interface Class Reference . . . 24
5.8.1 Detailed Description . . . 25
5.9 MainWindow Class Reference . . . 25
5.9.1 Detailed Description . . . 29
5.9.2 Constructor & Destructor Documentation . . . 29
5.9.2.1 MainWindow(QWidget ∗parent=0) . . . 29
5.9.3 Member Function Documentation . . . 30
5.9.3.1 addMainItem() . . . 30
5.9.3.2 buildTrees() . . . 30
5.9.3.3 checkboxCamera . . . 30
5.9.3.4 checkboxImage . . . 30
5.9.3.5 checkboxVideo . . . 30
5.9.3.6 evtCalibration . . . 31
5.9.3.7 evtVision . . . 31
5.9.3.8 getAllDevices() . . . 32
5.9.3.9 mouseCurrentPos . . . 32
5.9.3.10 mouseLeftPressed . . . 32
5.9.3.11 mouseRightPressed . . . 32

5.10 common::Pixel Struct Reference . . . 32
5.10.1 Detailed Description . . . 33
5.11 QCustomLabel Class Reference . . . 33
5.11.1 Detailed Description . . . 35
5.11.2 Member Function Documentation . . . 35
5.11.2.1 wheelEvent(QWheelEvent ∗ev) . . . 35
5.12 vss_state::RGB Class Reference . . . 35
5.13 common::Robot Struct Reference . . . 37
5.13.1 Detailed Description . . . 38
5.14 vss_command::Robot_Command Class Reference . . . 38
5.15 vss_state::Robot_State Class Reference . . . 40
5.16 SQLite Class Reference . . . 41
5.16.1 Detailed Description . . . 42
5.16.2 Member Function Documentation . . . 42
5.16.2.1 callback(void ∗NotUsed, int argc, char ∗∗argv, char ∗∗azColName) . . . 42
5.16.2.2 open() . . . 42
5.17 common::State Struct Reference . . . 43
5.17.1 Detailed Description . . . 44
5.18 vss_command::StaticDescriptorInitializer_command_2eproto Struct Reference . . . 44
5.19 vss_state::StaticDescriptorInitializer_state_2eproto Struct Reference . . . 44
5.20 common::TableColor Struct Reference . . . 44
5.20.1 Detailed Description . . . 45
5.21 vision Class Reference . . . 45
5.21.1 Detailed Description . . . 47
5.22 common::VisionColor Struct Reference . . . 47
5.22.1 Detailed Description . . . 48

Index **49**

Chapter 1

Namespace Index

1.1 Namespace List

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

common
This namespace is responsible to implement all global functions and custom datatypes used in the software, like: ENUMS, btVector3, Pixel, TableColor, VisionColor, ExecConfiguration, Calibration, Robot and State 7

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

common::btVector3 . . . 13
common::Calibration . . . 18
common::ExecConfiguration . . . 19
Interface . . . 24
Message
  vss_command::Global_Commands . . . 20
  vss_command::Robot_Command . . . 38
  vss_state::Ball_State . . . 11
  vss_state::Global_State . . . 22
  vss_state::RGB . . . 35
  vss_state::Robot_State . . . 40
common::Pixel . . . 32
QLabel
  QCustomLabel . . . 33
QMainWindow
  MainWindow . . . 25
QThread
  calibration . . . 13
  vision . . . 45
common::Robot . . . 37
SQLite . . . 41
common::State . . . 43
vss_command::StaticDescriptorInitializer_command_2eproto . . . 44
vss_state::StaticDescriptorInitializer_state_2eproto . . . 44
common::TableColor . . . 44
common::VisionColor . . . 47

Chapter 3

Class Index

3.1 Class List

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

vss_state::Ball_State 11
common::btVector3
This struct represents a Vector in R^3 13
calibration
This class is responsible for calibrate all common::VisionColor, cut points and rotation 13
common::Calibration
This struct represents a calibration of colors 18
common::ExecConfiguration
This struct represents a configuration of colors of objects in workspace 19
vss_command::Global_Commands 20
vss_state::Global_State 22
Interface
This class is responsible for handle the communicatio between VSS-Vision, VSS-Simulator, V↩
SS-Viewer and VSS-SampleStrategy 24
MainWindow
This class is the main window of the software 25
common::Pixel
This struct represents a Pixel, can be: RGB and HSV. It's used float because is possible to represent color in systems, like: 0 to 1, and 0 to 255 32
QCustomLabel
This class is a modification of QLabel, QCustomLabel handle events like: mouse click, mouse move, scrool wheel move and etc 33
vss_state::RGB 35
common::Robot
This strcut represets the pose that one robot can handle. Pos and Vel 37
vss_command::Robot_Command 38
vss_state::Robot_State 40
SQLite
This class is responsible for create, read, update and delete common::Calibration 's and common::ExecConfiguration 's 41
common::State
This struct represents the state that the workspace can handle 43
vss_command::StaticDescriptorInitializer_command_2eproto 44
vss_state::StaticDescriptorInitializer_state_2eproto 44
common::TableColor
This struct represents all colors in RGB that can be used 44

vision
This class is responsible for track all objects in field 45
common::VisionColor
This struct represents a Color that can be calibrated: made by 2 Pixels that represents a range of color, Bottom Pixel Limit and Top Pixel Limit 47

Chapter 4

Namespace Documentation

4.1 common Namespace Reference

This namespace is responsible to implement all global functions and custom datatypes used in the software, like: ENUMS, btVector3, Pixel, TableColor, VisionColor, ExecConfiguration, Calibration, Robot and State.

Classes

- struct btVector3

  This struct represents a Vector in R^3.
- struct Calibration

  This struct represents a calibration of colors.
- struct ExecConfiguration

  This struct represents a configuration of colors of objects in workspace.
- struct Pixel

  This struct represents a Pixel, can be: RGB and HSV. It's used float because is possible to represent color in systems, like: 0 to 1, and 0 to 255.
- struct Robot

  This strcut represets the pose that one robot can handle. Pos and Vel.
- struct State

  This struct represents the state that the workspace can handle.
- struct TableColor

  This struct represents all colors in RGB that can be used.
- struct VisionColor

  This struct represents a Color that can be calibrated: made by 2 Pixels that represents a range of color, Bottom Pixel Limit and Top Pixel Limit.

Enumerations

- enum { **r** = 0, **g** = 1, **b** = 2 }

  This enum implements a abstraction of a RGB array.
- enum {
  rmin = 0, **gmin** = 1, **bmin** = 2, **rmax** = 3,
  gmax = 4, **bmax** = 5 }

  This enum implements a abstraction of a range of RGB array.

- enum { **h** = 0, **s** = 1, **v** = 2 }

  This enum implements a abstraction of a HSV array.
- enum {
  hmin = 0, **smin** = 1, **vmin** = 2, **hmax** = 3,
  smax = 4, **vmax** = 5 }

  This enum implements a abstraction of a range of HSV array.
- enum { **OUR_TEAM** = 0, **ADVERSARY_TEAM** = 1 }

  This enum implements a abstraction of teams.
- enum {
  ORANGE = 0, **BLUE** = 1, **YELLOW** = 2, **RED** = 3,
  PINK = 4, **PURPLE** = 5, **GREEN** = 6, **BROWN** = 7,
  ROTATION = 8, **CUT** = 9 }

  ! This enum represents all data that can be calibrated.
- enum { **SQUARES** = 0, **RECTANGLES** = 1, **CIRCLES** = 2 }

  This enum represents all types of blob that can be used.
- enum { **UNKNOWN** = -1 }

  This enum represents all unknown things.
- enum { **CAMERA** = 0, **IMAGE** = 1, **VIDEO** = 2 }

  This enum represents all types of input datas.
- enum {
  BALL = 0, **TEAM1_ROBOT1** = 1, **TEAM1_ROBOT2** = 2, **TEAM1_ROBOT3** = 3,
  TEAM2_ROBOT1 = 4, **TEAM2_ROBOT2** = 5, **TEAM2_ROBOT3** = 6 }

  This enum represents all objects in a State.

Functions

- void clearSS (stringstream &ss)

  This function clean a stringstream.
- string toString (int a)

  Get int and convert to string.
- string toString (float a)

  Get float and convert to string.
- string toString (double a)

  Get double and convert to string.
- string toString (long long int a)

  Get long long int and convert to string.
- string toString (bool a)

  Get boolean and convert to string.
- int toInt (string a)

  Get string and convert to Int.
- float toFloat (string a)

  Get string and convert to float.
- double toDouble (string a)

  Get string and convert to double.
- long long int toLongLongInt (string a)

  Get string and convert to long long int.
- bool toBool (string a)

  Get string and convert to bool.
- string cmdTerminal (string s)

  Send string to terminal and get the returned value.
- double distancePoint (btVector3, btVector3)

Estimate distance between a set of points.

- double distancePoint (Point, Point)

  Estimate distance between two points.

- Point midpoint (Point, Point)

  Estimate midpoint between two points.

- Point midpoint (Rect)

  Estimate gravity center in a quadrilateral.

- btVector3 midpoint (btVector3, btVector3)

  Estimate midpoint between a set of points.

- float angulation (Point, Point)

  Estimate angle between two straight lines.

- double radian (Point, Point)

  Estimate angle between two straight lines in radian.

4.1.1 Detailed Description

This namespace is responsible to implement all global functions and custom datatypes used in the software, like: ENUMS, btVector3, Pixel, TableColor, VisionColor, ExecConfiguration, Calibration, Robot and State.

4.1.2 Function Documentation

4.1.2.1 float common::angulation (Point *a*, Point *b*)

Estimate angle between two straight lines.

Addendum

Estimate angle between two straight lines. One line formed by the two points in the function and the other line is formed by the point in the center of the robot (estimated using midpoint() function) and for a point in the pitch where the line must form a ninety degree angle with the pitch side.

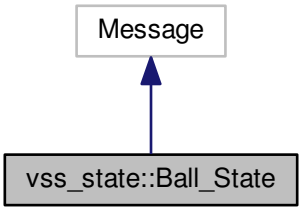
Estimate angle between two straight lines. One line formed by the two points in the function and the other line is formed by the point in the center of the robot (estimated using midpoint() function) and for a point in the pitch where the line must form a ninety degree angle with the pitch side.

Chapter 5

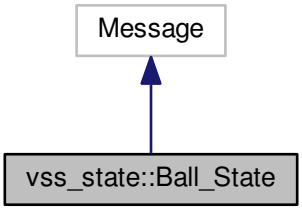
Class Documentation

5.1 vss_state::Ball_State Class Reference

Inheritance diagram for vss_state::Ball_State:



Collaboration diagram for vss_state::Ball_State:



Public Member Functions

- **Ball_State** (const Ball_State &from)
- Ball_State & **operator=** (const Ball_State &from)
- const ::google::protobuf::UnknownFieldSet & **unknown_fields** () const
- inline::google::protobuf::UnknownFieldSet ∗ **mutable_unknown_fields** ()
- void **Swap** (Ball_State ∗other)
- Ball_State ∗ **New** () const
- void **CopyFrom** (const ::google::protobuf::Message &from)
- void **MergeFrom** (const ::google::protobuf::Message &from)
- void **CopyFrom** (const Ball_State &from)
- void **MergeFrom** (const Ball_State &from)
- void **Clear** ()
- bool **IsInitialized** () const
- int **ByteSize** () const
- bool **MergePartialFromCodedStream** (::google::protobuf::io::CodedInputStream ∗input)
- void **SerializeWithCachedSizes** (::google::protobuf::io::CodedOutputStream ∗output) const
- ::google::protobuf::uint8 ∗ **SerializeWithCachedSizesToArray** (::google::protobuf::uint8 ∗output) const
- int **GetCachedSize** () const
- ::google::protobuf::Metadata **GetMetadata** () const
- bool **has_x** () const
- void **clear_x** ()
- float **x** () const
- void **set_x** (float value)
- bool **has_y** () const
- void **clear_y** ()
- float **y** () const
- void **set_y** (float value)

Static Public Member Functions

- static const ::google::protobuf::Descriptor ∗ **descriptor** ()
- static const Ball_State & **default_instance** ()

Static Public Attributes

- static const int **kXFieldNumber** = 1
- static const int **kYFieldNumber** = 2

Friends

- void **protobuf_AddDesc_state_2eproto** ()
- void **protobuf_AssignDesc_state_2eproto** ()
- void **protobuf_ShutdownFile_state_2eproto** ()

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

- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/protos/state.pb.h
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/protos/state.pb.cc

5.2 common::btVector3 Struct Reference

This struct represents a Vector in R^3.

`#include <common.h>`

Public Member Functions

- btVector3 ()

  Default constructor: btVector3 bt3;.
- btVector3 (float x, float y, float z)

  Construtor XYZ: btVector3 bt3(x, y, z);.
- btVector3 (btVector3 *b)

  Constructor copy: btVector3 bt3(btVector3(x, y, z));.
- btVector3 (Point *b)

  Constructor parse of point: btVector3 bt3(Point(x, y));.
- void show ()

  Default function: prints all variables.

Public Attributes

- float x

  Data: x, y, z.
- float **y**
- float **z**

5.2.1 Detailed Description

This struct represents a Vector in R^3.

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

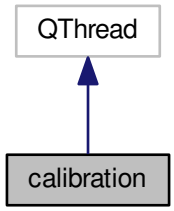
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/common.h

5.3 calibration Class Reference

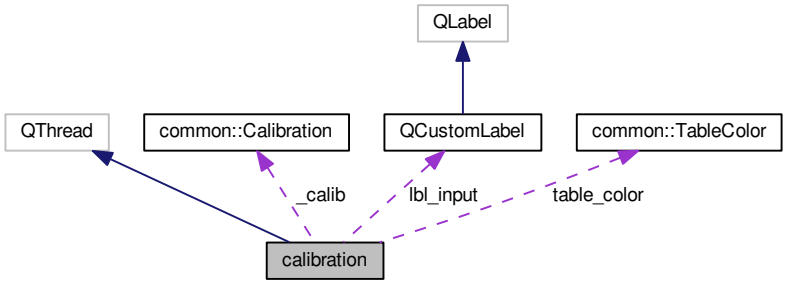
This class is responsible for calibrate all common::VisionColor, cut points and rotation.

`#include <calibration.h>`

Inheritance diagram for calibration:



Collaboration diagram for calibration:



Public Member Functions

- void calibration ()

  Default constructor: calibration c;.
- void run ()

  Virtual method from QThread, MainWindow call this method in a CONNECT.
- void finish ()

  Virtual method from QThread, MainWindow call this method in its destrcutor.
- void applyFilters ()

  Apply all filters needed to find one blob.
- void zoom ()

  Apply Zoom on image, if needed.
- void paint_output ()

  Paint the blob founded.
- void set_device (int)

  Set the device time used, common::IMAGE, common::CAMERA or common::VIDEO.
- void set_id_camera (int)

  Set the id from camera used.
- void set_path_image (string)

Set the path of image used.

- void set_path_video (string)

  Set the path of video used.

- void set_vision_reception (bool)

  Turn ON and OFF the vision reception and calcs.

- void set_id_color (int)

  Set the id_color to calibrate, like: common::ORANGE, common::YELLOW ...

- void set_mouse_click_left (int)

  Set the qtd of left clicks.

- void set_mouse_click_right (int)

  Set the qtd of right clicks.

- int get_device ()

  get the device used

- int get_id_camera ()

  get the id from camera used

- string get_path_image ()

  get the path of image used

- bool get_vision_reception ()

  get the path of video used

- int get_id_color ()

  get the id color calibrate

- int get_mouse_click_left ()

  get the qtd of left clicks

- int get_mouse_click_right ()

  get the qtd of right clicks

- void alloc_label_input (QCustomLabel *)

  alloc the label of input image, image from VideoCapture

- void alloc_calibration (Calibration *)

  alloc the labels of pose from objects in workspace

Protected Attributes

- bool **run_it**
- int **device_used**
- bool **vision_reception**
- bool **start_finish**
- int **id_camera**
- string **path_image**
- string **path_video**
- int **mouse_click_left**
- int **mouse_click_right**
- int **id_calib**
- Point **zoom_blob**
- Mat **in**
- Mat **out**
- Mat **saved**
- Mat **raw_in**
- Mat **raw_out**
- Mat **storage**
- Point **size**
- VideoCapture **cap**

- vector< vector< Point > > **labels**
- vector< vector< Point > > **contours**
- vector< Vec4i > **hierarchy**
- QImage **img**
- QCustomLabel * **lbl_input**
- Calibration * **_calib**
- TableColor **table_color**

5.3.1 Detailed Description

This class is responsible for calibrate all common::VisionColor, cut points and rotation.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 calibration::calibration ( )

Default constructor: calibration c;.

Addendum

Initializes all control variables

TODO: resolt the path, we need to use relative paths

5.3.3 Member Function Documentation

5.3.3.1 void calibration::alloc_calibration (Calibration * *_calib*)

alloc the labels of pose from objects in workspace

Addendum

It must be pointer, because calibration it's unrecheable and MainWindow too.

5.3.3.2 void calibration::alloc_label_input (QCustomLabel * *lbl_input*)

alloc the label of input image, image from VideoCapture

Addendum

It must be pointer, because calibration it's unrecheable and MainWindow too.

5.3.3.3 void calibration::applyFilters ( )

Apply all filters needed to find one blob.

Addendum

convert image, RGB -> HSV

get the blobs

filter very small blobs

find the contour of all blobs

paint all blobs

Draw a with rectangle on all blobs

5.3.3.4 void calibration::paint_output ( )

Paint the blob founded.

Addendum

If the HSV values its on range, paint with color: common::TableColor.

5.3.3.5 void calibration::run ( )

Virtual method from QThread, MainWindow call this method in a CONNECT.

Addendum

loop from thread calibration

load the saved image

default imagem, blue screen

set the screen on MainWindow

If vision reception must be used

Check devices and get image from them

if IMAGE it's used, usleep(33333) its needed to limit fps in 30.

if VIDEO it's used, and the video end, we reinititialize it

if VIDEO it's used, usleep(33333) it's needed to limit fps in 30

Apply filters

Apply Zoom, if needed

update the image in MainWindow

If the vision reception was set false, we set the blue screen and release cv::VideoCapture

If the vision reception it's false, usleep(100000) to relieve the CPU

5.3.3.6 void calibration::set_vision_reception (bool)

Turn ON and OFF the vision reception and calcs.

Addendum

If device_used == common::CAMERA or common::VIDEO, start de VideoCapture

5.3.3.7 void calibration::zoom ( )

Apply Zoom on image, if needed.

Addendum

If the user click one time, the zoom it's turned on If the user click one second time, the zoom it's turned off

security, prevents that the zoom get out of image

security, prevents that the zoom get out of image

security, prevents that the zoom get out of image

security, prevents that the zoom get out of image

Apply de Zoom

Restart de count of clicks

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

- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/calibration.h
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/calibration.cpp

5.4 common::Calibration Struct Reference

This struct represents a calibration of colors.

`#include <common.h>`

Public Member Functions

- Calibration ()

  Default constructor: Calibration c; initialize the vector with range of colors in HSV to track the objects with Saved Video and Saved Image.
- Calibration (Calibration *c)

  Constructor copy: Calibration c(Calibration());.
- void show ()

  Default function: prints all variables.

Public Attributes

- string comment

  Data: comment of calibration. used to save in db.
- vector< VisionColor > colors

  Data: vector of VisionColor.
- vector< Point > cut

  Data: vector of Point.
- string data

  Data: day of calibration. used ti save in db.

5.4.1 Detailed Description

This struct represents a calibration of colors.

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

- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/common.h

5.5 common::ExecConfiguration Struct Reference

This struct represents a configuration of colors of objects in workspace.

```
#include <common.h>
```

Public Member Functions

- ExecConfiguration ()

  Default constructor: ExecConfiguration exex;.
- ExecConfiguration (ExecConfiguration *g)

  Constructor copy: ExexConfiguration exex(ExecConfiguration());.
- void show ()

  Default function: prints all variables.

Public Attributes

- string comment

  Data: comment of configuration. used to save in db.
- int ball_color

  Data: ENUM id_color.
- int team_color [2]

  Data: ENUM id_color.
- int config_labels [2]

  Data: ENUM id_color.
- int secundary_color_1 [3]

  Data: ENUM id_color.
- int secundary_color_2 [3]

  Data: ENUM id_color.

5.5.1 Detailed Description

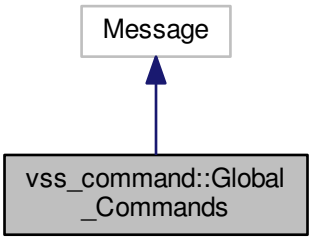
This struct represents a configuration of colors of objects in workspace.

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

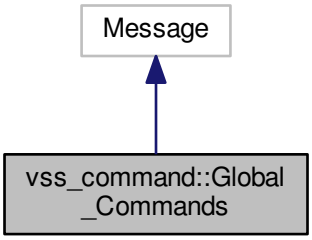
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/common.h

5.6 vss_command::Global_Commands Class Reference

Inheritance diagram for vss_command::Global_Commands:



Collaboration diagram for vss_command::Global_Commands:



Public Member Functions

- **Global_Commands** (const Global_Commands &from)
- Global_Commands & **operator=** (const Global_Commands &from)
- const ::google::protobuf::UnknownFieldSet & **unknown_fields** () const
- inline::google::protobuf::UnknownFieldSet * **mutable_unknown_fields** ()
- void **Swap** (Global_Commands *other)
- Global_Commands * **New** () const
- void **CopyFrom** (const ::google::protobuf::Message &from)
- void **MergeFrom** (const ::google::protobuf::Message &from)
- void **CopyFrom** (const Global_Commands &from)
- void **MergeFrom** (const Global_Commands &from)
- void **Clear** ()
- bool **IsInitialized** () const
- int **ByteSize** () const
- bool **MergePartialFromCodedStream** (::google::protobuf::io::CodedInputStream *input)
- void **SerializeWithCachedSizes** (::google::protobuf::io::CodedOutputStream *output) const
- ::google::protobuf::uint8 * **SerializeWithCachedSizesToArray** (::google::protobuf::uint8 *output) const
- int **GetCachedSize** () const
- ::google::protobuf::Metadata **GetMetadata** () const
- bool **has_id** () const
- void **clear_id** ()
- inline::google::protobuf::uint32 **id** () const
- void **set_id** (::google::protobuf::uint32 value)
- bool **has_is_team_yellow** () const
- void **clear_is_team_yellow** ()
- bool **is_team_yellow** () const
- void **set_is_team_yellow** (bool value)
- int **robot_commands_size** () const
- void **clear_robot_commands** ()
- const ::vss_command::Robot_Command & **robot_commands** (int index) const
- inline::vss_command::Robot_Command * **mutable_robot_commands** (int index)
- inline::vss_command::Robot_Command * **add_robot_commands** ()
- const ::google::protobuf::RepeatedPtrField< ::vss_command::Robot_Command > & **robot_commands** () const
- inline::google::protobuf::RepeatedPtrField< ::vss_command::Robot_Command > * **mutable_robot_commands** ()

Static Public Member Functions

- static const ::google::protobuf::Descriptor * **descriptor** ()
- static const Global_Commands & **default_instance** ()

Static Public Attributes

- static const int **kIdFieldNumber** = 1
- static const int **kIsTeamYellowFieldNumber** = 2
- static const int **kRobotCommandsFieldNumber** = 3

Friends

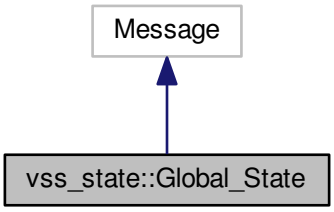
- void **protobuf_AddDesc_command_2eproto** ()
- void **protobuf_AssignDesc_command_2eproto** ()
- void **protobuf_ShutdownFile_command_2eproto** ()

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

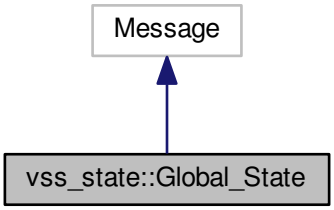
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/protos/command.pb.h
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/protos/command.pb.cc

5.7 vss_state::Global_State Class Reference

Inheritance diagram for vss_state::Global_State:



Collaboration diagram for vss_state::Global_State:



Public Member Functions

- **Global_State** (const Global_State &from)
- Global_State & **operator=** (const Global_State &from)
- const ::google::protobuf::UnknownFieldSet & **unknown_fields** () const
- inline::google::protobuf::UnknownFieldSet * **mutable_unknown_fields** ()
- void **Swap** (Global_State *other)
- Global_State * **New** () const
- void **CopyFrom** (const ::google::protobuf::Message &from)
- void **MergeFrom** (const ::google::protobuf::Message &from)
- void **CopyFrom** (const Global_State &from)
- void **MergeFrom** (const Global_State &from)
- void **Clear** ()
- bool **IsInitialized** () const
- int **ByteSize** () const
- bool **MergePartialFromCodedStream** (::google::protobuf::io::CodedInputStream *input)
- void **SerializeWithCachedSizes** (::google::protobuf::io::CodedOutputStream *output) const
- ::google::protobuf::uint8 * **SerializeWithCachedSizesToArray** (::google::protobuf::uint8 *output) const
- int **GetCachedSize** () const
- ::google::protobuf::Metadata **GetMetadata** () const
- bool **has_id** () const
- void **clear_id** ()
- inline::google::protobuf::uint32 **id** () const
- void **set_id** (::google::protobuf::uint32 value)
- bool **has_origin** () const
- void **clear_origin** ()
- bool **origin** () const
- void **set_origin** (bool value)
- int **balls_size** () const
- void **clear_balls** ()
- const ::vss_state::Ball_State & **balls** (int index) const
- inline::vss_state::Ball_State * **mutable_balls** (int index)
- inline::vss_state::Ball_State * **add_balls** ()
- const ::google::protobuf::RepeatedPtrField< ::vss_state::Ball_State > & **balls** () const
- inline::google::protobuf::RepeatedPtrField< ::vss_state::Ball_State > * **mutable_balls** ()
- int **robots_yellow_size** () const
- void **clear_robots_yellow** ()
- const ::vss_state::Robot_State & **robots_yellow** (int index) const
- inline::vss_state::Robot_State * **mutable_robots_yellow** (int index)
- inline::vss_state::Robot_State * **add_robots_yellow** ()
- const ::google::protobuf::RepeatedPtrField< ::vss_state::Robot_State > & **robots_yellow** () const
- inline::google::protobuf::RepeatedPtrField< ::vss_state::Robot_State > * **mutable_robots_yellow** ()
- int **robots_blue_size** () const
- void **clear_robots_blue** ()
- const ::vss_state::Robot_State & **robots_blue** (int index) const
- inline::vss_state::Robot_State * **mutable_robots_blue** (int index)
- inline::vss_state::Robot_State * **add_robots_blue** ()
- const ::google::protobuf::RepeatedPtrField< ::vss_state::Robot_State > & **robots_blue** () const
- inline::google::protobuf::RepeatedPtrField< ::vss_state::Robot_State > * **mutable_robots_blue** ()

Static Public Member Functions

- static const ::google::protobuf::Descriptor * **descriptor** ()
- static const Global_State & **default_instance** ()

Static Public Attributes

- static const int **kIdFieldNumber** = 1
- static const int **kOriginFieldNumber** = 2
- static const int **kBallsFieldNumber** = 3
- static const int **kRobotsYellowFieldNumber** = 4
- static const int **kRobotsBlueFieldNumber** = 5

Friends

- void **protobuf_AddDesc_state_2eproto** ()
- void **protobuf_AssignDesc_state_2eproto** ()
- void **protobuf_ShutdownFile_state_2eproto** ()

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

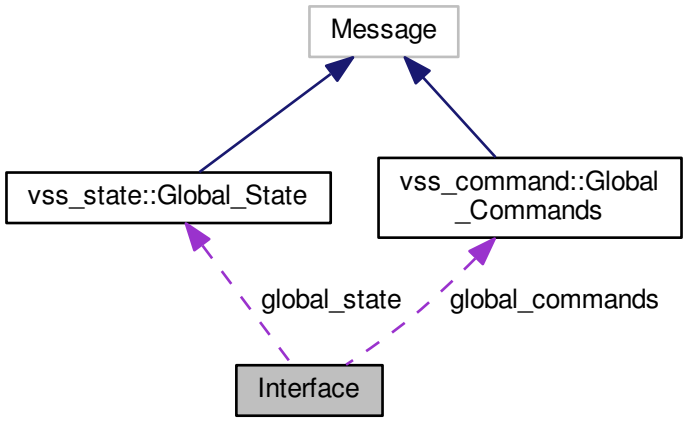
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/protos/state.pb.h
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/protos/state.pb.cc

5.8 Interface Class Reference

This class is responsible for handle the communicatio between VSS-Vision, VSS-Simulator, VSS-Viewer and VSS-SampleStrategy.

`#include <interface.h>`

Collaboration diagram for Interface:



Public Member Functions

- void **createSocketSendState** (vss_state::Global_State *)
- void **sendState** ()
- void **createSocketReceiveState** (vss_state::Global_State *)
- void **receiveState** ()
- void **createLoopSendCommandsTeam1** (vss_command::Global_Commands *)
- void **createLoopReceiveCommandsTeam1** (vss_command::Global_Commands *)
- void **createLoopSendCommandsTeam2** (vss_command::Global_Commands *)
- void **createLoopReceiveCommandsTeam2** (vss_command::Global_Commands *)
- void **printState** ()
- void **printCommand** ()

Protected Attributes

- zmq::message_t **request**
- zmq::context_t * **context**
- zmq::socket_t * **socket**
- vss_state::Global_State * **global_state**
- vss_command::Global_Commands * **global_commands**
- const char * **addr_server_multicast** = "tcp://*:5555"
- const char * **addr_client_multicast** = "tcp://localhost:5555"
- const char * **addr_server_simulator_team1** = "tcp://*:5556"
- const char * **addr_client_simulator_team1** = "tcp://localhost:5556"
- const char * **addr_server_simulator_team2** = "tcp://*:5557"
- const char * **addr_client_simulator_team2** = "tcp://localhost:5557"

5.8.1 Detailed Description

This class is responsible for handle the communicatio between VSS-Vision, VSS-Simulator, VSS-Viewer and VSS-SampleStrategy.

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

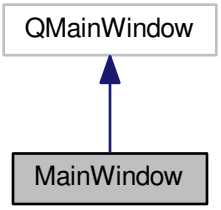
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/interface.h
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/interface.cpp

5.9 MainWindow Class Reference

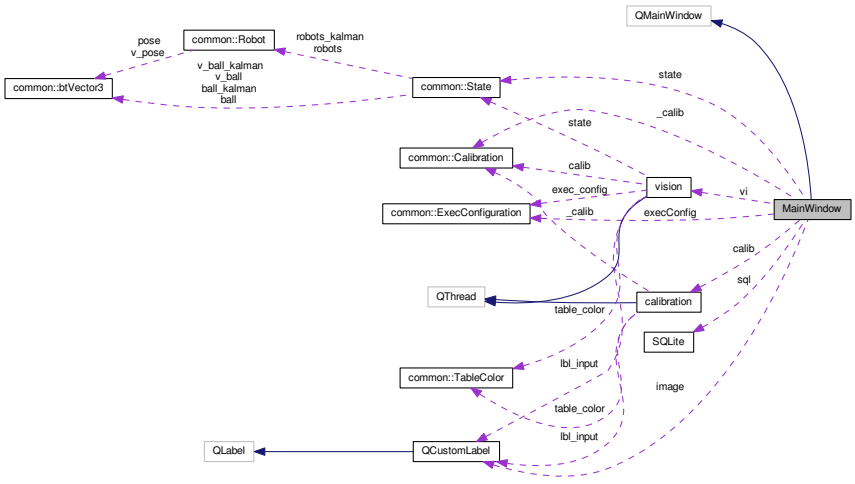
This class is the main window of the software.

`#include <mainwindow.h>`

Inheritance diagram for MainWindow:



Collaboration diagram for MainWindow:



Public Slots

- void mouseCurrentPos ()

  Method that get the signal from move the mouse inside QCustomLabel input image.
- void mouseLeftPressed ()

  Method that get the signal from left click inside QCustomLabel input image.
- void mouseRightPressed ()

  Method that get the signal from right click inside QCustomLabel input image.
- void mouseReleased ()

  Method that get the signal from release left click inside QCustomLabel input image.
- void mouseLeave ()

  Method that get the signal from release the mouse from QCustomLabel input image.
- void evtCalibration ()

  Method that get the signal from click on QPushButton Do Calibration.
- void evtVision ()

Method that get the signal from click on QPushButton Run Vision.

- void checkboxCamera (int)

  Method that get the signal from click on QCheckbox of use camera.
- void checkboxImage (int)

  Method that get the signal from click on QCheckbox of use image.
- void checkboxVideo (int)

  Method that get the signal from click on QCheckbox of use video.
- void updateHmin (int)

  Method that get the signal from slide on HSV-Hmin.
- void updateSmin (int)

  Method that get the signal from slide on HSV-SMin.
- void updateVmin (int)

  Method that get the signal from slide on HSV-Vmin.
- void updateHmax (int)

  Method that get the signal from slide on HSV-Hmax.
- void updateSmax (int)

  Method that get the signal from slide on HSV-Smax.
- void updateVmax (int)

  Method that get the signal from slide on HSV-Vmax.

Public Member Functions

- MainWindow (QWidget ∗parent=0)

Protected Member Functions

- void buildTrees ()

  Method responsible for build all the left tab.
- void addMainItem ()

  Method responsible for add the main item "Yellow box".
- void addInputDataItem ()

  Method responsible for add the item of choose input data.
- void addCameraCalibrationItem ()

  Method reponsible for add the item of calibrate camera, cut points and rotation.
- void addBlobFindingItem ()

  Method responsible for add the item of set parameters of vision, like for blobs: area-min, area-max, proportion-max, proportion-min.
- void addVisualizationItem ()

  Method responsible for add the item of set visualization (DEPRECATED)
- void addDefinePatternsItem ()

  Method responsible for add the item that defines the type of pattern of colors (TODO)
- void addExecutionConfig ()

  Method responsible for add the item of define the configuration of colors of each team.
- void addCalibrateColors ()

  Method responsible for add the item of calibration of colors.
- void **addVisionOptions** ()
- void addNetOptions ()

  Method responsible for add the item of define IP and PORT for interface.
- void addExecutionOptions ()

  Method reponsible form add the item of initializes the vision tracking.

- void initCalibrationColors ()

  Method responsible for update the layout for turn on the calibration.
- void finishCalibrationColors ()

  Method responsible for update the layout for turn off the calibration.
- void initPlotValues ()

  Method responsible for update the layout for turn on the vision.
- void finishPlotValues ()

  Method responsible for update the layout for turn off the vision.
- void updateValuesHSV ()

  Method responsible for update de commom::VisionColor of calibration, depends of id_color in common::TableColor.
- void update_hsv_s ()

  Method responsible for update the slide HSV-Smin and HSV-Smax, with the id_color in common::TableColor.
- void getAllDevices ()

  Method responsible for get all cameras on PC.
- void defineColors ()

  Method responsible for define the colors that gonna be used on vision.
- int translateColor (QString)

  Method responsible for translate the colors on QCheckbox to vector.

Protected Attributes

- calibration **calib**
- vision **vi**
- SQLite * **sql**
- Ui::MainWindow * **ui**
- QIcon **blockdevice**
- QIcon **ksame**
- QIcon **kdf**
- QIcon **package**
- QTreeWidgetItem * **mainItem**
- QList< QTreeWidgetItem * > **inputData**
- QList< QTreeWidgetItem * > **cameraCalibration**
- QList< QTreeWidgetItem * > **blobFinding**
- QList< QTreeWidgetItem * > **visualization**
- QList< QTreeWidgetItem * > **definePatterns**
- QList< QTreeWidgetItem * > **defineExecutionConfig**
- QList< QTreeWidgetItem * > **calibrateColors**
- QList< QTreeWidgetItem * > **visionOptions**
- QList< QTreeWidgetItem * > **netOptions**
- QList< QTreeWidgetItem * > **executionOptions**
- QComboBox * **cmbCameraIds**
- QComboBox * **cmbSavedImages**
- QComboBox * **cmbSavedVideos**
- QComboBox * **cmbColors**
- QComboBox * **cmbMainColors_1**
- QComboBox * **cmbMainColors_2**
- QComboBox * **cmbSecColors_1**
- QComboBox * **cmbSecColors_2**
- QComboBox * **cmbSecColors_3**
- QComboBox * **cmbSecColors_4**
- QComboBox * **cmbSecColors_5**
- QComboBox * **cmbSecColors_6**

- QComboBox * **cmbBallColors**
- QCheckBox * **checkUseCamera**
- QCheckBox * **checkUseImage**
- QCheckBox * **checkUseVideo**
- QPushButton * **btnDoColorCalib**
- QPushButton * **btnRunVision**
- QWidget * **contLayoutH3**
- vector< QLabel * > **lblHeadersHSV**
- vector< QSlider * > **slidersHSV**
- vector< QLabel * > **lblHeadersPlot**
- vector< QLabel * > **lblPlots**
- QLabel * **lbl_val**
- QCustomLabel * **image**
- QLabel * **coordinate_mouse**
- QLabel * **zoom_image**
- QImage * **in**
- vector< string > **devices**
- vector< int > **colors**
- Calibration **_calib**
- State **state**
- ExecConfiguration **execConfig**
- stringstream **ss**
- string **hsv_s**
- bool **calib_vs_vision**
- bool **init**

5.9.1 Detailed Description

This class is the main window of the software.

5.9.2 Constructor & Destructor Documentation

5.9.2.1 MainWindow::MainWindow (QWidget * *parent* = 0) [explicit]

Addendum

Define the connection to SQLite

Create all QThings

Define icons used

Get all cameras connected to PC

Build the left tab

Disable camera option, if any camera it's connected

Begin Define styles

End Define styles

Initializes the calibration thread

Initializes the vision thread

5.9.3 Member Function Documentation

5.9.3.1 void MainWindow::addMainItem ( ) [protected]

Method responsible for add the main item "Yellow box".

Addendum

Create the entire tab

5.9.3.2 void MainWindow::buildTrees ( ) [protected]

Method responsible for build all the left tab.

Addendum

Set the number os columns and the size of them

5.9.3.3 void MainWindow::checkboxCamera (int *a*) [slot]

Method that get the signal from click on QCheckbox of use camera.

Addendum

5.9.3.4 void MainWindow::checkboxImage (int *a*) [slot]

Method that get the signal from click on QCheckbox of use image.

Addendum

If image it's true, video and camera it's false

5.9.3.5 void MainWindow::checkboxVideo (int *a*) [slot]

Method that get the signal from click on QCheckbox of use video.

Addendum

If video it's true, iamge and camera it's false

5.9.3.6 void MainWindow::evtCalibration ( ) [slot]

Method that get the signal from click on QPushButton Do Calibration.

Addendum

Toggle between ON/OFF calibration

If camera it's used, set device common::CAMERA and its id

If image it's used, set device common::IMAGE

If video it's used, set device common::VIDEO

Send which color will be calibrate

Turn ON the calibration thread

Disable options of input data

Update the values of HSV that will be plot on sliders

Enable options of input data

Turn OFF the calibration thread

5.9.3.7 void MainWindow::evtVision ( ) [slot]

Method that get the signal from click on QPushButton Run Vision.

Addendum

Toggle between ON/OFF calibration

If camera it's used, set device common::CAMERA and its id

If image it's used, set device common::IMAGE

If video it's used, set device common::VIDEO

Disable options of input data

Turn ON the vision thread

Enable options of input data

Turn OFF the vision thread

5.9.3.8 void MainWindow::getAllDevices ( ) [protected]

Method responsible for get all cameras on PC.

Addendum

Trait the exit or common::cmdTerminal();

5.9.3.9 void MainWindow::mouseCurrentPos ( ) [slot]

Method that get the signal from move the mouse inside QCustomLabel input image.

Addendum

Update the plot values on layout

5.9.3.10 void MainWindow::mouseLeftPressed ( ) [slot]

Method that get the signal from left click inside QCustomLabel input image.

Addendum

Update the qtd of left clicks on vision

5.9.3.11 void MainWindow::mouseRightPressed ( ) [slot]

Method that get the signal from right click inside QCustomLabel input image.

Addendum

Update the qtd of right clicks on vision

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

- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/mainwindow.h
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/mainwindow.cpp

5.10 common::Pixel Struct Reference

This struct represents a Pixel, can be: RGB and HSV. It's used float because is possible to represent color in systems, like: 0 to 1, and 0 to 255.

`#include <common.h>`

Public Member Functions

- Pixel ()

  Constructor default: Pixel p;.
- Pixel (float r, float g, float b)

  Constructor RGB1: Pixel p(r, g, b);.
- Pixel (float rgb[3])

  Constructor RGB2: Pixel p(rgb[3]);.
- Pixel (Pixel *p)

  Constructor copy: Pixel p(Pixel(r, g, b));.
- Pixel (btVector3 *p)

  Constructor btVector3: Pixel p(btVector(r, g, b));.
- void show ()

  Default function: prints all variables.

Public Attributes

- float rgb [3]

  Data: array[3] of RGB.

5.10.1 Detailed Description

This struct represents a Pixel, can be: RGB and HSV. It's used float because is possible to represent color in systems, like: 0 to 1, and 0 to 255.

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

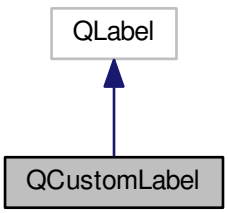
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/common.h

5.11 QCustomLabel Class Reference

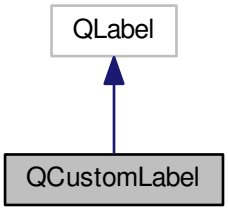
This class is a modification of QLabel, QCustomLabel handle events like: mouse click, mouse move, scrool wheel move and etc.

```
#include <qcustomlabel.h>
```

Inheritance diagram for QCustomLabel:



Collaboration diagram for QCustomLabel:



Signals

- void Mouse_Pos ()

  Send signal of the mouse movement.
- void Mouse_Left_Pressed ()

  Send signal of the mouse left key pressed.
- void Mouse_Right_Pressed ()

  Send signal of the mouse right key pressed.
- void Mouse_Release ()

  Send signal of the mouse in/out of the label area.
- void Mouse_Left ()

  Send signal asking if the mouse left button is pressed.
- void Mouse_Scroll ()

  Send signal of the mouse scroll rolled.
- void Key_Pressed ()

  Send signal of the mouse scroll pressed.

Public Member Functions

- **QCustomLabel** (QLabel ∗parent=0)
- void mouseMoveEvent (QMouseEvent ∗ev)

  This method make some adds modificationsin at the Qt method Mouse_Pos().
- void mousePressEvent (QMouseEvent ∗ev)

  This method make some adds modificationsin at the Qt method Mouse_Right_Pressed() and Mouse_Left_Pressed().
- void mouseReleaseEvent (QMouseEvent ∗ev)

  This method make some adds modificationsin at the Qt method Mouse_Release().
- void leaveEvent (QEvent ∗ev)

  This method make some adds modificationsin at the Qt method Mouse_Left().
- void wheelEvent (QWheelEvent ∗ev)

  This method make some adds modificationsin at the Qt method Mouse_Scroll().
- void keyPressEvent (QKeyEvent ∗ev)

  This method make some adds modificationsin at the Qt method Key_Pressed().

Public Attributes

- int **x**
- int **y**
- int **delta**
- int **volatil**
- int **last_left_click_x**
- int **last_left_click_y**
- int **last_right_click_x**
- int **last_right_click_y**

5.11.1 Detailed Description

This class is a modification of QLabel, QCustomLabel handle events like: mouse click, mouse move, scrool wheel move and etc.

5.11.2 Member Function Documentation

5.11.2.1 void QCustomLabel::wheelEvent (QWheelEvent * *ev*)

This method make some adds modificationsin at the Qt method Mouse_Scroll().

Addendum

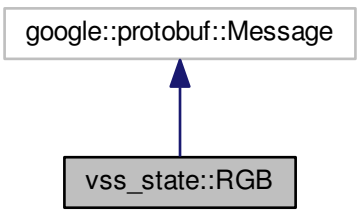
Get the variation of the mouse scroll and set a gain to the scroll velocity.

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

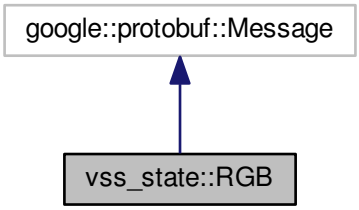
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/qcustomlabel.h
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/qcustomlabel.cpp

5.12 vss_state::RGB Class Reference

Inheritance diagram for vss_state::RGB:



Collaboration diagram for vss_state::RGB:



Public Member Functions

- **RGB** (const RGB &from)
- RGB & **operator=** (const RGB &from)
- const ::google::protobuf::UnknownFieldSet & **unknown_fields** () const
- inline::google::protobuf::UnknownFieldSet * **mutable_unknown_fields** ()
- void **Swap** (RGB *other)
- RGB * **New** () const
- void **CopyFrom** (const ::google::protobuf::Message &from)
- void **MergeFrom** (const ::google::protobuf::Message &from)
- void **CopyFrom** (const RGB &from)
- void **MergeFrom** (const RGB &from)
- void **Clear** ()
- bool **IsInitialized** () const
- int **ByteSize** () const
- bool **MergePartialFromCodedStream** (::google::protobuf::io::CodedInputStream *input)
- void **SerializeWithCachedSizes** (::google::protobuf::io::CodedOutputStream *output) const
- ::google::protobuf::uint8 * **SerializeWithCachedSizesToArray** (::google::protobuf::uint8 *output) const
- int **GetCachedSize** () const
- ::google::protobuf::Metadata **GetMetadata** () const
- bool **has_r** () const
- void **clear_r** ()
- inline::google::protobuf::uint32 **r** () const
- void **set_r** (::google::protobuf::uint32 value)
- bool **has_g** () const
- void **clear_g** ()
- inline::google::protobuf::uint32 **g** () const
- void **set_g** (::google::protobuf::uint32 value)
- bool **has_b** () const
- void **clear_b** ()
- inline::google::protobuf::uint32 **b** () const
- void **set_b** (::google::protobuf::uint32 value)

Static Public Member Functions

- static const ::google::protobuf::Descriptor * **descriptor** ()
- static const RGB & **default_instance** ()

Static Public Attributes

- static const int **kRFieldNumber** = 1
- static const int **kGFieldNumber** = 2
- static const int **kBFieldNumber** = 3

Friends

- void **protobuf_AddDesc_state_2eproto** ()
- void **protobuf_AssignDesc_state_2eproto** ()
- void **protobuf_ShutdownFile_state_2eproto** ()

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

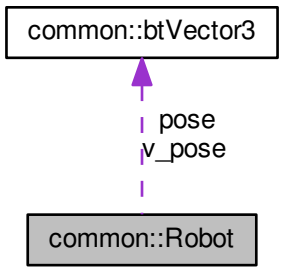
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/protos/state.pb.h
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/protos/state.pb.cc

5.13 common::Robot Struct Reference

This strcut represets the pose that one robot can handle. Pos and Vel.

`#include <common.h>`

Collaboration diagram for common::Robot:



Public Member Functions

- Robot ()

  Default constructor: Robot t;.
- Robot (btVector3 pose, btVector3 v_pose)

  Constructor 2: Robot t(btVector3(x, y, yaw), btVector3(x, y, yaw))
- Robot (Robot *r)

  Constructor copy: Robot t(Robot());.
- void show ()

  Default function: prints all variables.

Public Attributes

- btVector3 pose

  Data: Pose.
- btVector3 v_pose

  Data: V_Pose.

5.13.1 Detailed Description

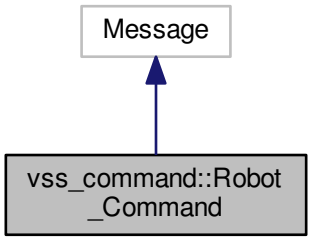
This strcut represets the pose that one robot can handle. Pos and Vel.

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

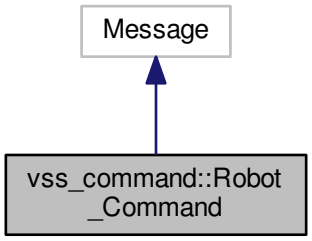
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/common.h

5.14 vss_command::Robot_Command Class Reference

Inheritance diagram for vss_command::Robot_Command:



Collaboration diagram for vss_command::Robot_Command:



Public Member Functions

- **Robot_Command** (const Robot_Command &from)
- Robot_Command & **operator=** (const Robot_Command &from)
- const ::google::protobuf::UnknownFieldSet & **unknown_fields** () const
- inline::google::protobuf::UnknownFieldSet ∗ **mutable_unknown_fields** ()
- void **Swap** (Robot_Command ∗other)
- Robot_Command ∗ **New** () const
- void **CopyFrom** (const ::google::protobuf::Message &from)
- void **MergeFrom** (const ::google::protobuf::Message &from)
- void **CopyFrom** (const Robot_Command &from)
- void **MergeFrom** (const Robot_Command &from)
- void **Clear** ()
- bool **IsInitialized** () const
- int **ByteSize** () const
- bool **MergePartialFromCodedStream** (::google::protobuf::io::CodedInputStream ∗input)
- void **SerializeWithCachedSizes** (::google::protobuf::io::CodedOutputStream ∗output) const
- ::google::protobuf::uint8 ∗ **SerializeWithCachedSizesToArray** (::google::protobuf::uint8 ∗output) const
- int **GetCachedSize** () const
- ::google::protobuf::Metadata **GetMetadata** () const
- bool **has_id** () const
- void **clear_id** ()
- inline::google::protobuf::uint32 **id** () const
- void **set_id** (::google::protobuf::uint32 value)
- bool **has_left_vel** () const
- void **clear_left_vel** ()
- float **left_vel** () const
- void **set_left_vel** (float value)
- bool **has_right_vel** () const
- void **clear_right_vel** ()
- float **right_vel** () const
- void **set_right_vel** (float value)

Static Public Member Functions

- static const ::google::protobuf::Descriptor ∗ **descriptor** ()
- static const Robot_Command & **default_instance** ()

Static Public Attributes

- static const int **kIdFieldNumber** = 1
- static const int **kLeftVelFieldNumber** = 2
- static const int **kRightVelFieldNumber** = 3

Friends

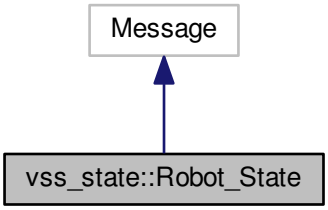
- void **protobuf_AddDesc_command_2eproto** ()
- void **protobuf_AssignDesc_command_2eproto** ()
- void **protobuf_ShutdownFile_command_2eproto** ()

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

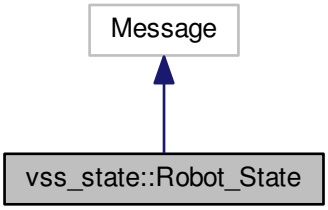
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/protos/command.pb.h
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/protos/command.pb.cc

5.15 vss_state::Robot_State Class Reference

Inheritance diagram for vss_state::Robot_State:



Collaboration diagram for vss_state::Robot_State:



Public Member Functions

- **Robot_State** (const Robot_State &from)
- Robot_State & **operator=** (const Robot_State &from)
- const ::google::protobuf::UnknownFieldSet & **unknown_fields** () const
- inline::google::protobuf::UnknownFieldSet ∗ **mutable_unknown_fields** ()
- void **Swap** (Robot_State ∗other)
- Robot_State ∗ **New** () const
- void **CopyFrom** (const ::google::protobuf::Message &from)
- void **MergeFrom** (const ::google::protobuf::Message &from)
- void **CopyFrom** (const Robot_State &from)
- void **MergeFrom** (const Robot_State &from)
- void **Clear** ()
- bool **IsInitialized** () const
- int **ByteSize** () const
- bool **MergePartialFromCodedStream** (::google::protobuf::io::CodedInputStream ∗input)
- void **SerializeWithCachedSizes** (::google::protobuf::io::CodedOutputStream ∗output) const
- ::google::protobuf::uint8 ∗ **SerializeWithCachedSizesToArray** (::google::protobuf::uint8 ∗output) const

- int **GetCachedSize** () const
- ::google::protobuf::Metadata **GetMetadata** () const
- bool **has_x** () const
- void **clear_x** ()
- float **x** () const
- void **set_x** (float value)
- bool **has_y** () const
- void **clear_y** ()
- float **y** () const
- void **set_y** (float value)
- bool **has_color** () const
- void **clear_color** ()
- const ::vss_state::RGB & **color** () const
- inline::vss_state::RGB * **mutable_color** ()
- inline::vss_state::RGB * **release_color** ()
- void **set_allocated_color** (::vss_state::RGB *color)
- bool **has_yaw** () const
- void **clear_yaw** ()
- float **yaw** () const
- void **set_yaw** (float value)

Static Public Member Functions

- static const ::google::protobuf::Descriptor * **descriptor** ()
- static const Robot_State & **default_instance** ()

Static Public Attributes

- static const int **kXFieldNumber** = 1
- static const int **kYFieldNumber** = 2
- static const int **kColorFieldNumber** = 3
- static const int **kYawFieldNumber** = 4

Friends

- void **protobuf_AddDesc_state_2eproto** ()
- void **protobuf_AssignDesc_state_2eproto** ()
- void **protobuf_ShutdownFile_state_2eproto** ()

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

- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/protos/state.pb.h
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/protos/state.pb.cc

5.16 SQLite Class Reference

This class is responsible for create, read, update and delete common::Calibration 's and common::Exec↩ Configuration 's.

```
#include <sqlite.h>
```

Public Member Functions

- SQLite (string, string)

  Default constructor: SQLite sql("name db", "pswd")
- void open ()

  This method open the connection with the database.
- void close ()

  This metohd close the connection with the database.
- void **insert_calibration** ()
- void **select_calibration** (string s="")

Static Protected Member Functions

- static int callback (void *NotUsed, int argc, char **argv, char **azColName)

  Print the data result from query in database.

Protected Attributes

- sqlite3 * **db**
- string **path_database**
- string **password**
- stringstream **query**
- int **status_db**
- char * **error_query**
- const char * **data**

5.16.1 Detailed Description

This class is responsible for create, read, update and delete common::Calibration 's and common::Exec↩ Configuration 's.

5.16.2 Member Function Documentation

5.16.2.1 int SQLite::callback (void * *NotUsed*, int *argc*, char ** *argv*, char ** *azColName*) [static], [protected]

Print the data result from query in database.

Addendum

Print the info like on the table in database

5.16.2.2 void SQLite::open ( )

This method open the connection with the database.

Addendum

Return any error, if something happen when try to open the database

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

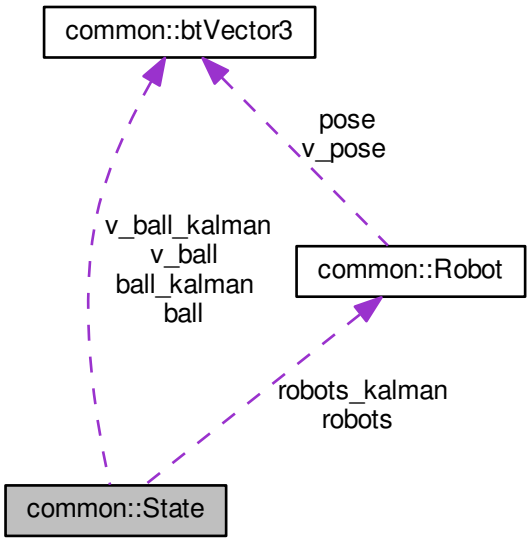
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/sqlite.h
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/sqlite.cpp

5.17 common::State Struct Reference

This struct represents the state that the workspace can handle.

`#include <common.h>`

Collaboration diagram for common::State:



Public Member Functions

- State ()

  Default constructor: State s;.
- void show ()

  Default function: prints all variables.

Public Attributes

- Robot robots [6]

  All robots by vision.
- Robot robots_kalman [6]

  All robots by kalman.
- btVector3 ball

  Pos ball by vision.
- btVector3 v_ball

  Vel ball by vision.
- btVector3 ball_kalman

  Pos ball by kalman.
- btVector3 v_ball_kalman

  Vel ball by kalman.

5.17.1 Detailed Description

This struct represents the state that the workspace can handle.

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

- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/common.h

5.18 vss_command::StaticDescriptorInitializer_command_2eproto Struct Reference

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

- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/protos/command.pb.cc

5.19 vss_state::StaticDescriptorInitializer_state_2eproto Struct Reference

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

- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/protos/state.pb.cc

5.20 common::TableColor Struct Reference

This struct represents all colors in RGB that can be used.

`#include <common.h>`

Public Member Functions

- TableColor ()

  Default constructor: TableColor tb;.

Public Attributes

- vector< Pixel > colors
  Data: vector of Pixel.

5.20.1 Detailed Description

This struct represents all colors in RGB that can be used.

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

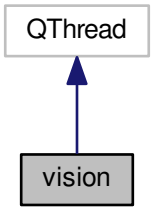
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/common.h

5.21 vision Class Reference

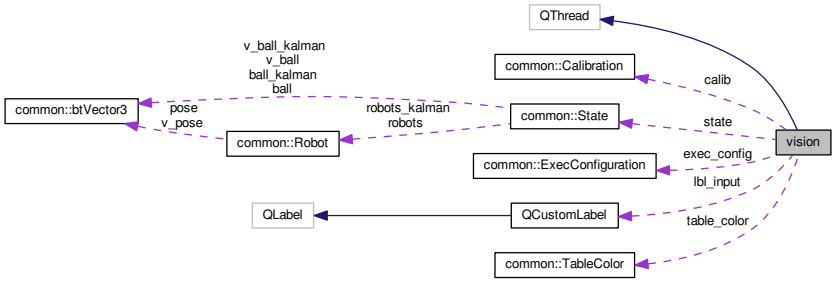
This class is responsible for track all objects in field.

`#include <vision.h>`

Inheritance diagram for vision:



Collaboration diagram for vision:



Public Member Functions

- void **run** ()
- void **search_color** (int)
- void **alloc_calibration** (Calibration ∗)
- void **alloc_state** (State ∗)
- void **alloc_label_input** (QCustomLabel ∗)
- void **alloc_colors** (vector< int > ∗)
- void **alloc_execution_config** (ExecConfiguration ∗)
- void **alloc_label_plots** (vector< QLabel ∗ > ∗)
- void **set_device** (int)
- void **set_id_camera** (int)
- void **set_path_image** (string)
- void **set_vision_reception** (bool)
- void **set_path_video** (string)
- int **get_device** ()
- int **get_id_camera** ()
- string **get_path_image** ()
- bool **get_vision_reception** ()
- string **get_path_video** ()
- void **finish** ()

Protected Member Functions

- vector< Rect > **orderByArea** (vector< Rect >)
- Rect **expandRect** (Point)
- bool **itsALabel** (Rect)
- bool **itsAwayFromLimits** (Point)
- void **recognizeObjects** ()
- bool **isValidPoint** (Point)
- void **updatePlot** ()

Protected Attributes

- bool **run_it**
- bool **start_finish**
- int **device_used**
- int **id_camera**
- bool **vision_reception**
- string **path_image**
- string **path_video**
- int **side_cut**
- float **area_min**
- float **area_max**
- float **distc_min**
- float **distc_max**
- float **propc_min**
- float **propc_max**
- Calibration ∗ **calib**
- State ∗ **state**
- ExecConfiguration ∗ **exec_config**
- TableColor **table_color**
- Mat **in**

- Mat **out**
- Mat **saved**
- Mat **raw_out**
- Mat **raw_in**
- Mat **storage**
- VideoCapture **cap**
- vector< vector< Point > > **coordinate_old**
- vector< vector< Point > > **labels**
- vector< vector< Point > > **contours**
- vector< Vec4i > **hierarchy**
- vector< Point > **vol_coordinate**
- vector< int > * **colors**
- bool **find_old_labels** [13]
- KalmanFilter **kfs** [13]
- QCustomLabel * **lbl_input**
- vector< QLabel * > * **lbl_plots**

5.21.1 Detailed Description

This class is responsible for track all objects in field.

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

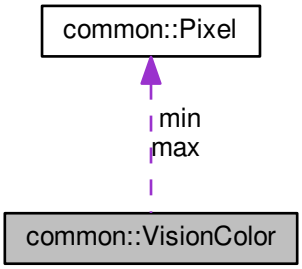
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/vision.h
- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/vision.cpp

5.22 common::VisionColor Struct Reference

This struct represents a Color that can be calibrated: made by 2 Pixels that represents a range of color, Bottom Pixel Limit and Top Pixel Limit.

`#include <common.h>`

Collaboration diagram for common::VisionColor:



Public Member Functions

- VisionColor ()

  Constructor default: VisionColor c;.
- VisionColor (Pixel min, Pixel max)

  Constructor Pixels: VisionColor c(Pixel (r, g, b), Pixel (r, g, b));.
- VisionColor (VisionColor ∗color)

  Constructor copy: VisionColor c(VisionColor);.
- void show ()

  Default function: prints all variables.

Public Attributes

- Pixel min

  Data: Bottom Pixel Limit.
- Pixel max

  Data: Top Pixel Limit.

5.22.1 Detailed Description

This struct represents a Color that can be calibrated: made by 2 Pixels that represents a range of color, Bottom Pixel Limit and Top Pixel Limit.

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

- /home/johnathan/Repositories/SIRLab/VSS-Vision/src/common.h

Index

addMainItem
    MainWindow, 30
alloc_calibration
    calibration, 16
alloc_label_input
    calibration, 16
angulation
    common, 9
applyFilters
    calibration, 16

buildTrees
    MainWindow, 30

calibration, 13
    alloc_calibration, 16
    alloc_label_input, 16
    applyFilters, 16
    calibration, 16
    paint_output, 17
    run, 17
    set_vision_reception, 17
    zoom, 18
callback
    SQLite, 42
checkboxCamera
    MainWindow, 30
checkboxImage
    MainWindow, 30
checkboxVideo
    MainWindow, 30
common, 7
    angulation, 9
common::Calibration, 18
common::ExecConfiguration, 19
common::Pixel, 32
common::Robot, 37
common::State, 43
common::TableColor, 44
common::VisionColor, 47
common::btVector3, 13

evtCalibration
    MainWindow, 30
evtVision
    MainWindow, 31

getAllDevices
    MainWindow, 31

Interface, 24

MainWindow, 25
    addMainItem, 30
    buildTrees, 30
    checkboxCamera, 30
    checkboxImage, 30
    checkboxVideo, 30
    evtCalibration, 30
    evtVision, 31
    getAllDevices, 31
    MainWindow, 29
    mouseCurrentPos, 32
    mouseLeftPressed, 32
    mouseRightPressed, 32
mouseCurrentPos
    MainWindow, 32
mouseLeftPressed
    MainWindow, 32
mouseRightPressed
    MainWindow, 32

open
    SQLite, 42

paint_output
    calibration, 17

QCustomLabel, 33
    wheelEvent, 35

run
    calibration, 17

SQLite, 41
    callback, 42
    open, 42
set_vision_reception
    calibration, 17

vision, 45
vss_command::Global_Commands, 20
vss_command::Robot_Command, 38
vss_command::StaticDescriptorInitializer_command_↩
        2eproto, 44
vss_state::Ball_State, 11
vss_state::Global_State, 22
vss_state::RGB, 35
vss_state::Robot_State, 40
vss_state::StaticDescriptorInitializer_state_2eproto, 44

wheelEvent
    QCustomLabel, 35

zoom
calibration, 18