Camera

Functions

int	**camera_open** ()
int	**camera_open_black** ()
int	**camera_open_at_res** (enum **Resolution** res)
int	**camera_open_device** (int number, enum **Resolution** res)
int	**camera_open_device_model_at_res** (int number, enum **Model** model, enum **Resolution** res)
int	**camera_load_config** (const char *name)
void	**set_camera_width** (int width)
void	**set_camera_height** (int height)
int	**get_camera_width** (void)
int	**get_camera_height** (void)
int	**camera_update** (void)
pixel	**get_camera_pixel** (**point2** p)
int	**get_channel_count** (void)
int	**get_object_count** (int channel)
const char *	**get_object_data** (int channel, int object)
int	**get_object_data_length** (int channel, int object)
double	**get_object_confidence** (int channel, int object)
int	**get_object_area** (int channel, int object)
rectangle	**get_object_bbox** (int channel, int object)
point2	**get_object_centroid** (int channel, int object)
point2	**get_object_center** (int channel, int object)
void	**camera_close** ()
void	**set_camera_config_base_path** (const char *const path)
const unsigned char *	**get_camera_frame_row** (unsigned row)
const unsigned char *	**get_camera_frame** ()

Detailed Description

Function Documentation

void camera_close ( )

Cleanup the current camera instance.

See also

camera_open

camera_open_at_res

camera_open_device

int camera_load_config (const char * name)

Loads the config file specified by name. The system will look for the config in the base path.

Parameters

name The configuration to load. Configuration file names are case sensitive.

Note

Do NOT include the config file extension ".conf" in the name parameter.

Returns

1 on success, 0 on failure.

See also

set_camera_config_base_path

int camera_open ( )

Opens the default system camera for use at LOW_RES (160x120).

Returns

1 on success, 0 on failure

See also

camera_open_at_res

camera_open_device

camera_close

int camera_open_at_res (enum Resolution res)

Opens the default system camera for use at a given resolution.

Parameters

res The resolution the camera should operate at. This can be:

- LOW_RES (160x120)
- MED_RES (320x240)
- HIGH_RES (640x480)

Warning

Only LOW_RES is currently supported. The function will fail for other resolutions.

Returns

1 on success, 0 on failure

See also

camera_open

camera_open_device

camera_close

int camera_open_black ( )

Opens the default system camera for use at LOW_RES (160x120). This will improve frame rates for the black Logitech camera

Returns

1 on success, 0 on failure

See also

camera_open_at_res

camera_open_device

camera_close

```
int camera_open_device ( int              number,
                         enum Resolution  res
                       )
```

Opens a camera for use.

Parameters

number	The camera's id. 0 is the first camera, 1 is the second camera, etc.
res	The resolution the camera should operate at. This can be: • LOW_RES (160x120) • MED_RES (320x240) • HIGH_RES (640x480)

> **Warning**
>
> Only LOW_RES is currently supported. The function will fail for other resolutions.

Returns

1 on success, 0 on failure

See also

camera_open

camera_close

```
int camera_open_device_model_at_res ( int              number,
                                      enum Model       model,
                                      enum Resolution  res
                                    )
```

Opens a camera for use.

Parameters

number The camera's id. 0 is the first camera, 1 is the second camera, etc.

res The resolution the camera should operate at. This can be:

- LOW_RES (160x120)
- MED_RES (320x240)
- HIGH_RES (640x480)

model The camera model

- WHITE_2016 The white 2016 Botball camera
- BLACK_2017 The black 2017 Botball camera

Warning

MED_RES is supported only for the BLACK_2017 camera and HIGH_RES is not supported

Returns

1 on success, 0 on failure

See also

camera_open

camera_close

```
int camera_update ( void  )
```

Pulls a new image from the camera for processing.

Returns

1 on success, 0 on failure.

const unsigned char* get_camera_frame ( )

Retrieves the current camera frame as a BGR (BGR888) array. The returned pointer is invalid after **camera_update()** is called again.

Returns

the current BGR888 camera frame.

const unsigned char* get_camera_frame_row (unsigned row)

Retrieves the current camera frame row as a BGR (BGR888) array. The returned pointer is invalid after **camera_update()** is called again.

Returns

the current BGR888 camera frame row.

int get_camera_height (void)

Gets the camera's y resolution.

Attention

This value might be different than the previously set y resolution. Never assume the y resolution.

Returns

The camera's y resolution, in pixels.

pixel get_camera_pixel (point2 p)

Gets the color of a pixel.

Parameters

p The point at which the pixel lies.

Returns

The rgb value of the pixel located at point p.

Note

A (r, g, b) value of (-1, -1, -1) will be returned for points that are out of range.

```
int get_camera_width ( void  )
```

Gets the camera's x resolution.

Attention

This value might be different than the previously set x resolution. Never assume the x resolution.

Returns

The camera's x resolution, in pixels.

```
int get_channel_count ( void  )
```

Returns

Number of channels in the current configuration.

See also

get_object_count

```
int get_object_area ( int  channel,
                      int  object
                    )
```

Returns

The object's bounding box area. -1 is returned if the channel or object doesn't exist.

```
rectangle get_object_bbox ( int  channel,
                            int  object
                          )
```

Returns

The bounding box of the given object on the given channel.

```
point2 get_object_center ( int  channel,
                           int  object
                         )
```

Returns

The (x, y) center of the given object on the given channel.

```
point2 get_object_centroid ( int  channel,
                             int  object
                           )
```

Returns

The (x, y) centroid of the given object on the given channel.

```
double get_object_confidence ( int  channel,
                               int  object
                             )
```

Returns

The confidence, between 0.0 and 1.0, that given object on the given channel is significant. If the channel or object doesn't exist, 0.0 is returned.

```
int get_object_count ( int  channel )
```

Parameters

channel The channel to scan for objects.

Note

Objects are sorted by area, largest first.

Returns

Number of objects in the given channel, -1 if channel doesn't exist.

See also

get_channel_count

```
const char* get_object_data ( int  channel,
                              int  object
                            )
```

Returns

The string data associated with a given object on a given channel. If there is no data associated, 0 is returned.

Note

This data is not guaranteed to be null terminated.

This string pointer will be invalid after a call to camera_update()

See also

get_object_data_length

```
int get_object_data_length ( int  channel,
                             int  object
                           )
```

Returns

The length of the string data associated with a given object on a given channel. If there is no data associated, 0 is returned. If the channel or object is invalid, 0 is returned.

See also

get_object_data

```
void set_camera_config_base_path ( const char *const  path )
```

Sets the path in which to look for camera configurations.

Parameters

path the absolute directory path in which to look for camera configurations.

void set_camera_height (int height)

Sets the camera's y resolution.

Warning

Setting the camera height is not currently supported.

Parameters

width The height in pixels

void set_camera_width (int width)

Sets the camera's x resolution.

Warning

Setting the camera width is not currently supported.

Parameters

width The width in pixels