Nikon Controller Wraper

15/06/2016

INDEX

1. Preamble……………………………………………………………………….4
2. Functionality of Matlab Class………………………………………….5
   2.1. Compiling c++ files in Matlab………………………………….5
   2.2. Connecting and disconecting the camera…………………5
   2.3. Capture images……………………………………………………5
   2.4. Get current camera parameters……………………………..7
   2.5. Set camera parameters……………………………………….8
   2.6. Get camera parameters range……………………………….9
   2.7. Change default parameters…………………………………..9
   2.8. Get Live View……………………………………………………10
3. Supported Camera Parameters…………………………………….11
4. Current state of the C++ wrapper…………………………………13
5. About Nikon 5200…………………………………………………………14

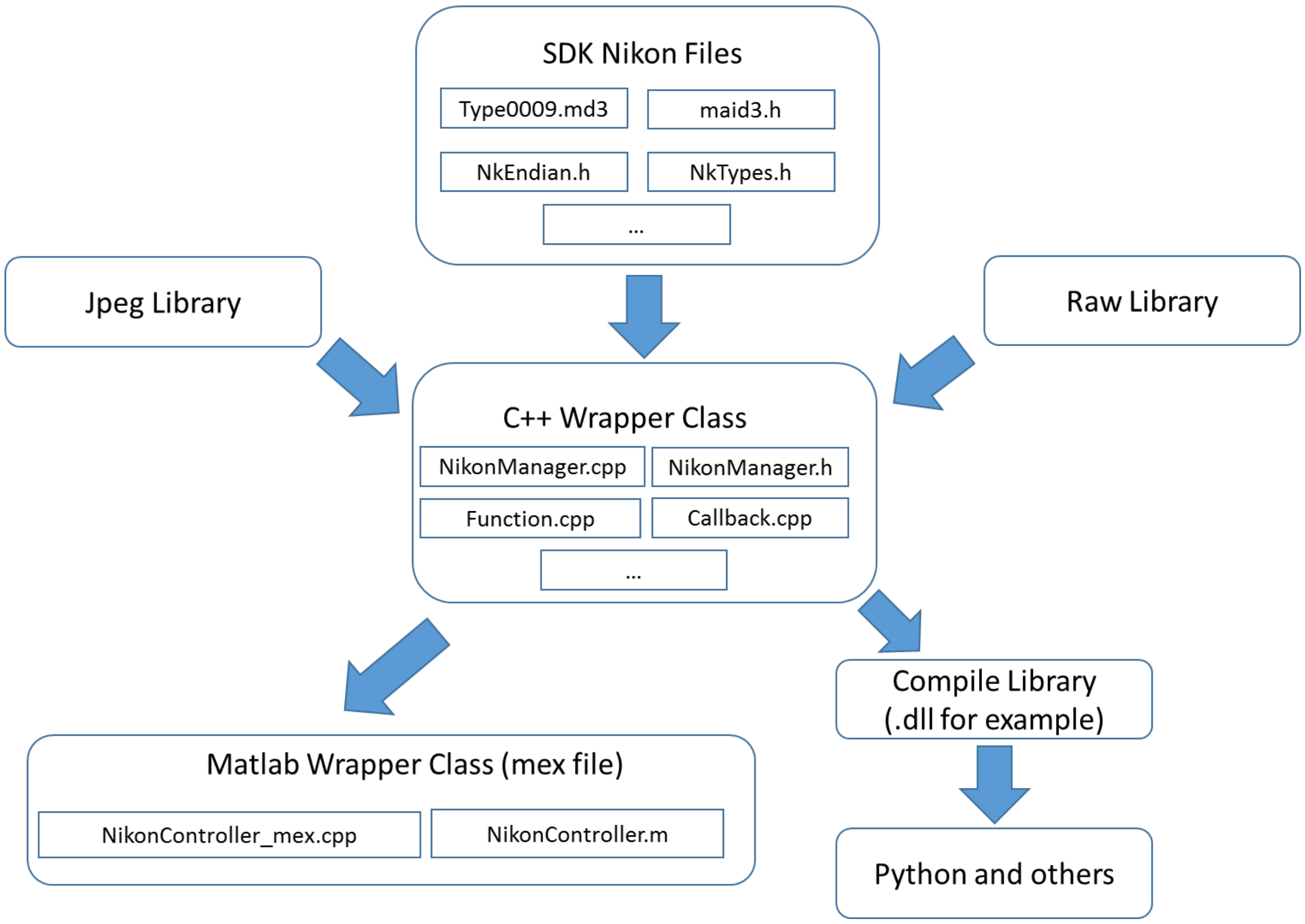
Revision History

Data	Versió	Descripció	Autor/s
30/05/2016	0.5	First version of the documentation. At this point, camera can only work at 32 bits applications. Basic functionalities are complete and explained in this document.	Victor Moyano
15/06/2016	1.0	Nikon released new 64 bits binaries, so the software is now adapted to work in 64 bit applications. A few more functionalities have been added. Such as: - Reading exif from jpeg images. - New parameter to control focus mode. - Improved the performance of the wrapper in general. - An example script of basic use of the camera have been written. - Code reorganized in general.	Victor Moyano

1. Preamble

The intention of this document is to give full knowledge on how to use the wrapper for controlling the camera Nikon D5200 using very simple Matlab instructions. In the moment of writing this document, only a C++ and Matlab wrapper are implemented, but change the Matlab wrapper to a Python wrapper it should not be a problem, if the programmer knows what he is doing.

I will try to explain which is the hierarchy of this project to ease the implementation of a Python (or any other language) wrapper in a future. The hierarchy is the following:



So, for further implementations in other languages, I recommend using the c++ class. Anyways, if someone is trying to fully understand the C++ class, I hardly recommend reading the SDK Nikon documentation.

At May 2015, Nikon released 64 bit binaries, so at this time the software is completely functional in both 32 and 64 bits operating systems. Two versions of the programs are given, one for 64, and other for 32.

2. Functionality of Matlab Class

All the c++ class functionality merge in to a Matlab class, this way a user can use the c++ class from Matlab in a very simple way. In this section we will explain how to use the Matlab class properly. Of course, you have to make sure you have set properly the working directory in Matlab.

Important note: In order to be able to control (nearly) all the interesting parameters of the camera, you have to make sure the exposure Mode is set to manual. If you are not sure if the camera is set in Manual, please take a look at section *5. About Nikon D5200*.

2.1. Compiling c++ files in matlab:

Before using the c++ files, we have to make sure the mex file is compiled. A compiled version is already provided but, just in case, I've written a file to compile all the files. So, the only thing you need to do is call the *compile file ("compile.m*"), that you will find inside the Source folder.

2.2. Connecting and disconnecting camera:

When the Nikon is connected via USB to the computer, you can use the following functions to get an object that you will use to control the camera.

When you connect the camera, you can choose to do it setting the default parameters, or just leaving the ones the camera has at that moment. This can be done by passing a parameter to the constructor function. If you want to set the parameters to some default values, just pass 'default' to the constructor. Some example:

- ➔ Connect setting default parameters:

  ```
  >> nikon= nikonController('default')
  ```

- ➔ Connect leaving the current camera parameters (without setting parameters to default) :

  ```
  >> nikon= nikonController
  ```

To disconnect the camera, do:

```
>> nikon.delete();
```

Be sure of always disconnecting the camera when you finished your work, since only one camera is supported, and if you try to connect to a camera that is already connected, Matlab could crash pretty badly.

2.3. Capture images:

Use the capture function to capture an image, or several. We can capture two images if you are shooting in RAW+JPEG mode. You can get both images, but there is no need to if you don't want to. If you are shooting in RAW+JPEG mode and only indicate one output, the JPEG image will be erased, and you will get the RAW image. The RAW image is always placed in the first output parameter.

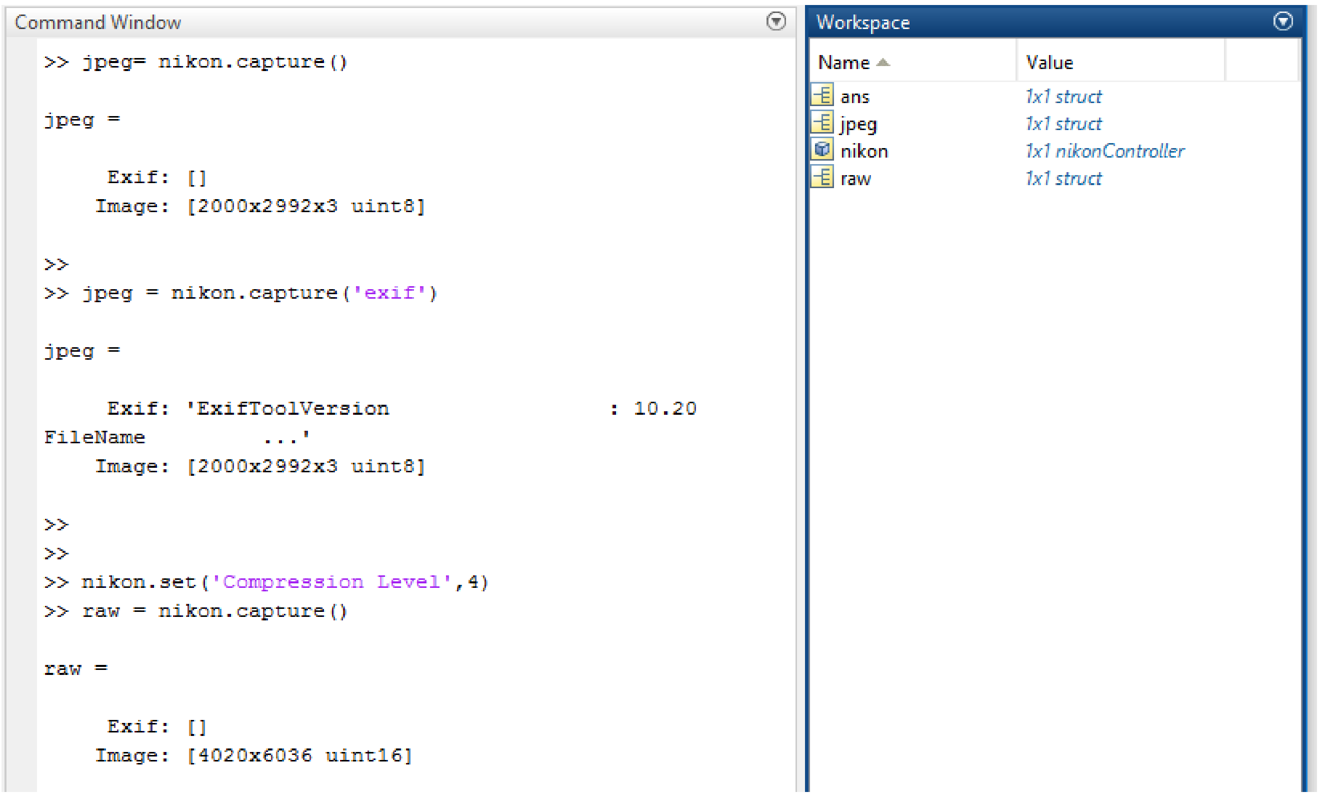
This function returns an struct containing two things: First the exif information (all the information about the parameters when taking the picture), and then the image.

If you want to get the exif information, you need to pass 'exif' as the parameters of this function. In case you don't need this information, don't pass any parameters, because this way you will get the image faster.

In the case of shooting in JPEG you will get an array of bytes (unsigned char, 8 bits per element) of dimensions height x width x 3 (due sRGB color space).

In the case of shooting in RAW you will get an array of shorts (unsigned short int, 16 bits per elements) of dimensions 4020 x 6036. In the case of Nikon 5200D, the dimensions of the RAW are always the ones shown before. Remember that the Nikon image stores 14 bits for each image, so you may need conversions to visualize the image properly.

Some examples here:



```
>> jpeg= nikon.capture()

jpeg = 

     Exif: []
    Image: [2000x2992x3 uint8]

>>
>> jpeg = nikon.capture('exif')

jpeg = 

     Exif: 'ExifToolVersion                 : 10.20
FileName         ...'
    Image: [2000x2992x3 uint8]

>>
>>
>> nikon.set('Compression Level',4)
>> raw = nikon.capture()

raw = 

     Exif: []
    Image: [4020x6036 uint16]
```

Name	Value
ans	1x1 struct
jpeg	1x1 struct
nikon	1x1 nikonController
raw	1x1 struct

As you can see, when the exif parameter is passed to the capture function, the exif cell is filled with all the given information.

2.4. Get current camera parameters:

Use the get function to get a Matlab struct containing parameters info. This functions returns a matlab struct. You can:

- Don't pass any parameter, so you get a Matlab struct containing all the parameters information. Example:

```
>> nikon.get()

ans = 

                Compression Level: 'RAW'
                       Image Size: 'L(6000*4000)'
               White Balance Mode: 'Auto'
                        Sensivity: '160'
                         Aperture: '6.3'
                    Metering Mode: 'Spot'
                    Shutter Speed: '1/60'
                  Flash Sync Mode: 'Normal'
                Active D Lighting: 'Off'
                  Auto-Distorsion: 'Off'
                Image Color Space: 'sRGB'
                         HDR Mode: 'Off'
      Continuous AF Area Priority: 'AF-C Shutter'
                     AF AreaPoint: '39 points'
    EV Steps for Exposure Control: '1/3 Step'
             Focus Preferred Area: '1'
                    Exposure Mode: 'Manual'
                Enable Bracketing: 'Not Used'
                      ISO Control: 'Off'
                     Af Sub Light: 'On'
                   Exposure Comp.: '0.00'
              Internal Flash Comp: '0.00'
```

- Pass a string referring to a specific parameter, so you get an struct containing the information about that parameter. Example:

```
>> nikon.get('Aperture')

ans = 

    Aperture: '6.3'
```

- Pass a number referring to the ID of a specific parameter (from 1 to 22), so you get a struct containing the information about that parameter. Example:

```
>> nikon.get(5)

ans = 

    Aperture: '6.3'
```

2.5. Set camera parameters:

Use the set function to set camera parameters. The inputs can be several. You can:

- pass a struct containing all the attribute names and values and set all the camera parameters to the ones given in the struct.
- pass an attribute name, and its value (positive integer or 0).
- pass an attribute id (from 1 to 22, since 22 parameters are supported), and its value (positive integer or 0).

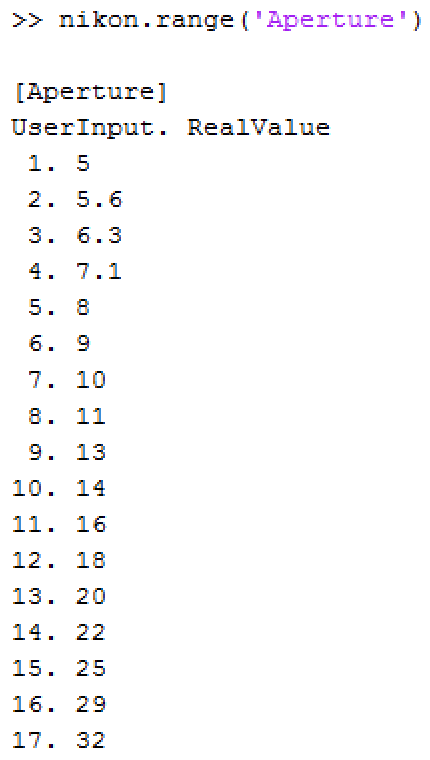
Some example:

```
>> parameters = nikon.get()

parameters = 

                Compression Level: 'RAW'
                       Image Size: 'L(6000*4000)'
               White Balance Mode: 'Auto'
                        Sensivity: '160'
                         Aperture: '6.3'
                    Metering Mode: 'Spot'
                    Shutter Speed: '1/60'
                  Flash Sync Mode: 'Normal'
                Active D Lighting: 'Off'
                  Auto-Distorsion: 'Off'
                Image Color Space: 'sRGB'
                         HDR Mode: 'Off'
     Continuous AF Area Priority: 'AF-C Shutter'
                     AF AreaPoint: '39 points'
   EV Steps for Exposure Control: '1/3 Step'
             Focus Preferred Area: '1'
                    Exposure Mode: 'Manual'
                Enable Bracketing: 'Not Used'
                      ISO Control: 'Off'
                     Af Sub Light: 'On'
                   Exposure Comp.: '0.00'
              Internal Flash Comp: '0.00'

>> nikon.set('Compression Level', 1)
>> nikon.get('Compression Level')

ans = 

    Compression Level: 'JPEG Basic'

>> nikon.set(parameters)
>> nikon.get('Compression Level')

ans = 

    Compression Level: 'RAW'
```

Important: note that the value passed to an attribute is not the real attribute value, but the user input. For example, if you want to set the aperture level to 8, you wont enter a 8 as the parameter value, but a 5, which corresponds to the aperture level= 8. You can look which inputs values you have to indicate using the range function.

2.6. Get camera parameters range:

To see all the values a given parameter can have, use the range function. You can get the range of all the parameters, or a specific one. An example here:

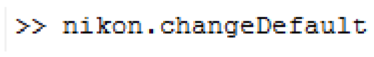


```
>> nikon.range('Aperture')

[Aperture]
UserInput. RealValue
 1. 5
 2. 5.6
 3. 6.3
 4. 7.1
 5. 8
 6. 9
 7. 10
 8. 11
 9. 13
10. 14
11. 16
12. 18
13. 20
14. 22
15. 25
16. 29
17. 32
```

2.7. Change default parameters:

The parameters set when calling the constructor with the default options, are the ones stored in the defaultParameters.mat file. You can find this file inside the Nikon folder. If you want to change the default Parameters the camera will be set when connecting, just call the changeDefault from the NikonController class. This function will change the default parameters stored in defaultParameters by the ones your camera has at the time when calling the function.

An example here:



```
>> nikon.changeDefault
```

2.8. Get Live Images:

There is a functionality in the wrapper that allows the user to see what the camera is seeing in a video-like mode. It just takes low quality frames and displays them in a figure in matlab. You can do it this way:

```
>> nikon.liveView
```

When you use this command, you should see a figure in Matlab showing live video from the camera.

3. Supported camera Parameters

The current camera parameters supported in this wrapper are the shown in the following table:

Parameter ID	**Name of Parameter**	**Short Description**
1	Compression Level	Determines in which format images will be captured (NEF, JPEG or both)
2	Image Size	Determines the size of the jpeg image. Nef images are always the maximum size.
3	White Balance Mode	To choose the white balance mode.
4	Sensitivity	Digital equivalent of film speed. The higher the ISO sensitivity, the less light needed to make an exposure. This way we can have higher shutter speeds or smaller apertures, but the more likely is the image to be affected by noise.
5	Aperture	Determines the area of exposure. The higher the f-number (number shown at the user) the less the area exposed. If the area decreases, the intensity of the image should decrease as well.
6	Metering Mode	Refers to the way in which a camera determines the exposure.
7	Shutter Speed	Time of exposition when capturing an image. The higher the Shutter Speed, the higher the intensity of the image.
8	Flash Sync Mode	Refer to flash mode. If the flash is closed, this has no effect at all.
9	Active D Lighting	When shooting in RAW, you would not have to worry about this. Is a postprocessing method applied to RAW images to do some modifications when obtaining a JPEG image.
10	Auto-Distorsion	It corrects barrel or pin cushion distortion that occurs in images due to the characteristics of specific lenses.
11	Image Color Space	Color space of the image taken.
12	HDR Mode	This is for getting better ranges of what is dark and what is light in the image taken.
13	Continuous AF Area Priority	Focus area when doing autofocus
14	AF AreaPoint	Select focus points when doing autofocus
15	EV Steps for Exposure Control	Set steps for exposure control
16	Focus Preferred Area	Select a preferred area to focus

17	Exposure Mode	Current camera mode. (Manual, Auto, Aperture.. etc)
18	Enable Bracketing	For shooting the same subject several times, to ensure at least one of the photos is properly exposed.
19	ISO Control	-
20	Af Sub Light	-
21	Exposure Comp.	Compensation of exposure
22	Internal Flash Comp	Compensation when using the built-in flash of the camera
23	Focus Mode	Algorithm that will be used for doing focus when capturing an image.

Some of the parameters shown before do not have any influence at all when shooting in RAW.

4. Current State of the C++ Wrapper

Note: this section is a work in progress.

In the moment of writing this document, the C++ NikonManager class has the following capabilities:

- Capture any picture (RAW or JPEG) and get the buffer image without saving it to disk. (Saving the image in to disk is a task the user must do).

- Get exif headers from the JPEG or RAW files and return it to user.

- Transform the buffer image in to data recognized by any programming language (i.e. array of bytes). There are several things to keep in mind when doing this. The buffer image that this wrapper provides contains bytes corresponding to an image file. This means, if we are shooting in jpeg, the buffer image will contain bytes that refer to the jpeg file. A jpeg file contains a lot of information besides the image information, for example a header (exif file). Same thing happens with .nef (RAW format for Nikon). The c++ wrapper contains functions for getting only the image date from jpeg and raw files and put it in to an array.

- Get the value of any of the parameters supported.

- Set the value of any of the parameters supported, although it can be exceptions, due the exposure Mode in which the camera is set.

- Get the range of any of the parameters supported.

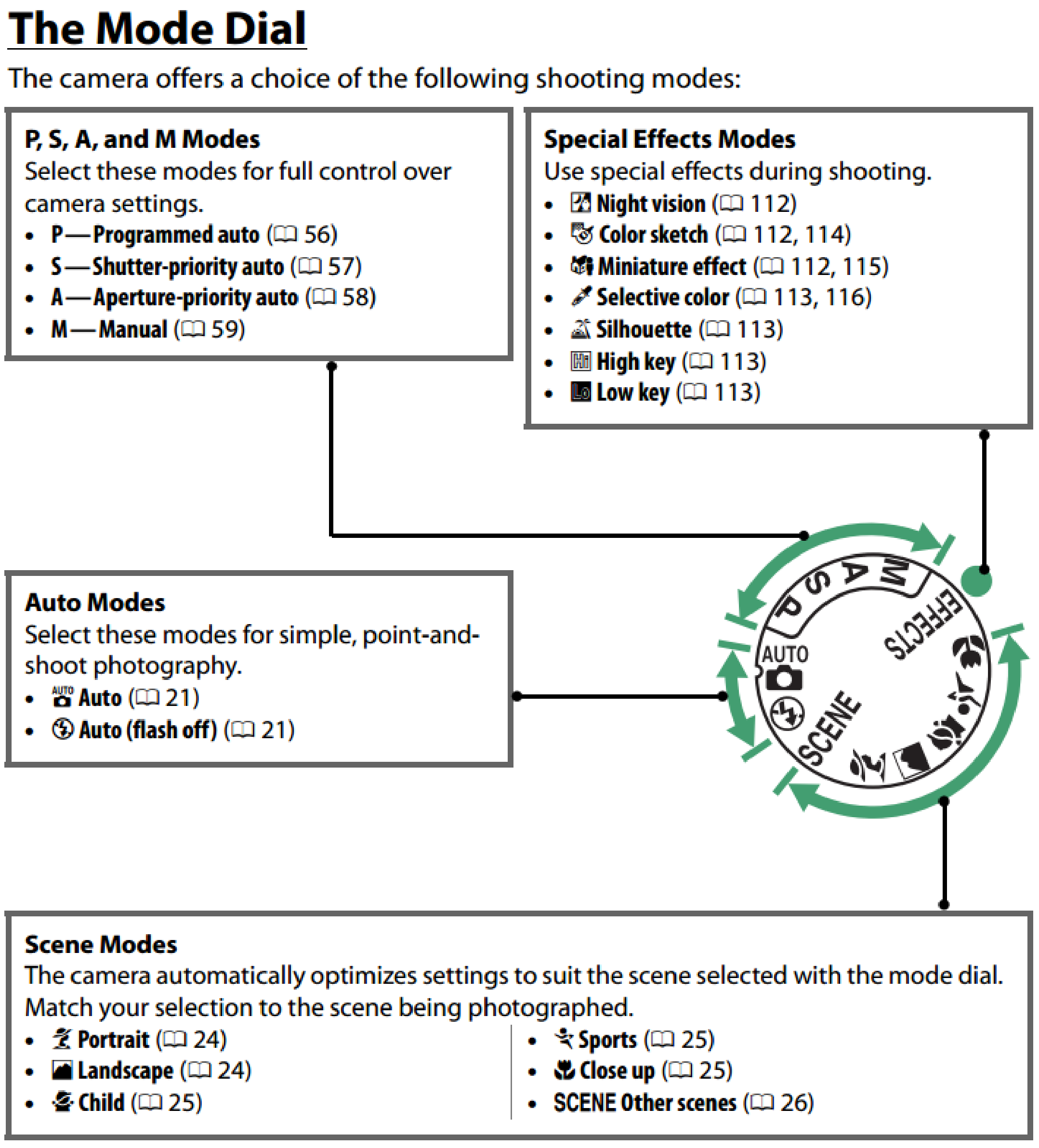
- Get LiveVideo from the camera, using the liveView function.

5. About Nikon D5200

Note: this section is a work in progress.

In this section I will put some information that may be useful for anyone using the Nikon D5200 model.

As said previously, the camera mode has to be set in manual in order to use the wrapper properly. This has to be done manual, since there is no way to do it by coding. Take a look at the next figure, for knowing if the camera is set in Manual:



The Mode Dial

The camera offers a choice of the following shooting modes:

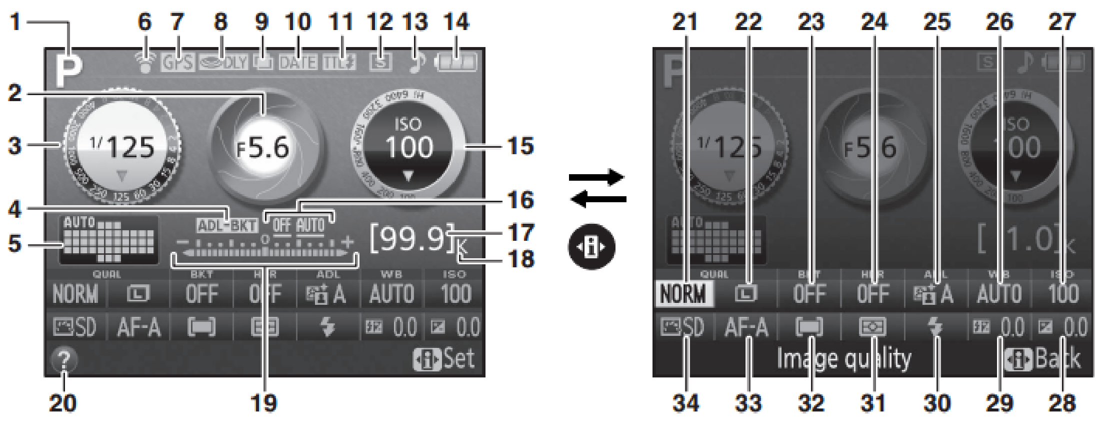
P, S, A, and M Modes
Select these modes for full control over camera settings.
- **P—Programmed auto** (🕮 56)
- **S—Shutter-priority auto** (🕮 57)
- **A—Aperture-priority auto** (🕮 58)
- **M—Manual** (🕮 59)

Special Effects Modes
Use special effects during shooting.
- **Night vision** (🕮 112)
- **Color sketch** (🕮 112, 114)
- **Miniature effect** (🕮 112, 115)
- **Selective color** (🕮 113, 116)
- **Silhouette** (🕮 113)
- **High key** (🕮 113)
- **Low key** (🕮 113)

Auto Modes
Select these modes for simple, point-and-shoot photography.
- **Auto** (🕮 21)
- **Auto (flash off)** (🕮 21)

Scene Modes
The camera automatically optimizes settings to suit the scene selected with the mode dial. Match your selection to the scene being photographed.
- **Portrait** (🕮 24)
- **Landscape** (🕮 24)
- **Child** (🕮 25)
- **Sports** (🕮 25)
- **Close up** (🕮 25)
- **SCENE Other scenes** (🕮 26)

Also, you can see if the current camera mode in the display screen, with some other interesting information about the camera. Take a look at the next scheme to learn about the information shown in the display screen:



- **1** Shooting mode
  - auto/
  - auto (flash off)
  - Scene modes
  - **P**, **S**, **A**, and **M** modes
  - Special effects mode
- **2** Aperture (f-number)
  Aperture display
- **3** Shutter speed
  Shutter-speed display
- **4** Bracketing indicator
- **5** Auto-area AF indicator
  3D-tracking indicator
  Focus point
- **6** Eye-Fi connection indicator
- **7** GPS connection indicator
- **8** Exposure delay mode
- **9** Multiple exposure indicator.
- **10** Print date indicator
- **11** Flash control indicator
  Flash compensation indicator for optional flash units
- **12** Release mode
- **13** “Beep” indicator
- **14** Battery indicator
- **15** ISO sensitivity
  ISO sensitivity display
  Auto ISO sensitivity indicator
- **16** ADL bracketing amount
- **17** Number of exposures remaining
  White balance recording indicator
  Capture mode indicator
- **18** “K” (appears when memory remains for over 1000 exposures)
- **19** Exposure indicator
  Exposure compensation indicator
  Bracketing progress indicator
- **20** Help icon
- **21** Image quality
- **22** Image size
- **23** Bracketing increment
- **24** HDR (High Dynamic Range)
- **25** Active D-Lighting
- **26** White balance
- **27** ISO sensitivity
- **28** Exposure compensation
- **29** Flash compensation
- **30** Flash mode
- **31** Metering
- **32** AF-area mode
- **33** Focus mode
- **34** Picture Control

Note: Display shown with all indicators lit for illustrative purposes.

In the shooting Mode (1, in the previous figure) you should see an M, indicating mode is set to manual.