pythonDMDapp quick documentation

1. Hardware

1.a) Class DMD:

Basically a sequence of "Try" --> "Except" to check if the DMD is already running (and, if not, to start it). Some of the Class' functionalities are:

- "start_DMD": starts the DMD (...). Doesn't need any input.

- "stop_DMD": stops the DMD (...). Doesn't need any input.

- "seq_alloc": allocates the sequence of images to the RAM memory of the DMD. Inputs are:
  * imgSeq: sequence of images, in the form "np.array([img_1, img_2, ... img_n])";
  * num_img: number of images in the sequence. The default is "1" (takes just the 1st img);
  * picture_time: how long each pattern stays in the DMD before being changed. Default is "1e6". Careful: picture_time is currently in microsseconds!

- "free_seq": frees the DMD from the previous sequence. Next sequence can be uploaded to it without having to use "start_DMD" again;

I would not change anything here.

1.b) Class Camera:

If a UEye CCD is used, this basically estabilishes communication with it, allowing to obtain pictures with the desired exposure time. Some of the Class' functionalities are:

- "start_ccd": starts the CCD (...). Doesn't need any input.

- "get_image": gets a picture with the CCD (results in a 2D np.array). If zoom=False, then the function results in the whole picture from the CCD (1280 x 1024). If zoom=True, then the result is a small (60 x 60) region from the original image.

  How to change the zoom region?
  Going into the "frame" variable, changing the section wanted from the original array.

```
frame = frame[475:535,  # y
              555:615]  # x
```

- "set_exposure": sets a new "exposure time" for the CCD in ms! This will be used in the subsequent pictures obtained from it.

2. Applications

2.a) Bragg beams:

```
def bragg_beams(lamb_interf, v, act_time,
                c = 0.2,
                cx = 0,
                cy = 0.2,
                use_phase_map = True,
                use_int_map = False,
                save_images = False):
  '''


    Parameters
    ----------
    lamb_interf : int, float
        Desired wavelength (in micron!) for the interf. pattern.
    v : int, float
        Desired velocity (in micron/s !) for the interf. pattern.
    act_time : int
        Acting time (in seconds!) for the DMD to be on.
    c : int, float; optional
        Fine-tuning parameter. The default is 0.2.
    cx : int, float, optional
        Fine-tuning parameter, x-direction. The default is 0.
    cy : int, float, optional
        Fine-tuning parameter, y-direction. The default is 0.
    use_phase_map : bool; optional
        The default is True.
    use_int_map : bool; optional
        The default is False.
    save_images : Bool, optional
        The default is False.

  '''
```

The *docstring* already contains explanation for the parameters that need to be used as input when calling the function "*bragg_beams.py*". Other things that can be changed are:

- "num_img": number of pictures stored in the DMD's RAM for this application. Currently using "num_img = 9" (as used in the main experiment, acc. to DPetter's Thesis); more images mean a "more smooth" dynamics of the interference pattern;

- "r": radius of the grated disk region (related to the "waist" of the Bragg beams). Be aware that a larger region at the DMD means a smaller region at the atoms' position! Currently at 20 px.

2.b) Moving bars:

```
def move_bars(v, act_time,
              c=0.2,
              cx=0.2,
              cy=0,
              use_phase_map = True,
              use_int_map = False,
              save_images = False):
  '''


    Parameters
    ----------
    v : int, float
        Velocity of the bars.
    act_time : int, float
        Time interval (in seconds) to keep the DMD on.
    c : int, float, optional
        Fine-tuning parameter. The default is 0.2.
    cx : int, float, optional
        Fine-tuning parameter, x-direction. The default is 0.5.
    cy : int, float, optional
        Fine-tuning parameter, y-direction. The default is 0.
    use_phase_map : Bool, optional
        Use (or not) the phase-map. The default is True.
    use_int_map : Bool, optional
        Use (or not) the intensity-map. The default is False.
    save_images : Bool, optional
        Save a sequence of images. Requires an Ueye CCD!
        The default is False.

    Returns
    -------
    Two-bar structure acting on the fourier plane.

  '''
```

What else can be changed (besides the inputs shown in the *docstring*):

- "num_img": same explanation as before (see **(2.a)** for explanation);

- "dimensions": changes the (x,y)-dimensions of each bar in the DMD -- means the opposite at the focus position.

2.c) Impurity:

```
def impurity(diameter, omega, act_time,
                cx = 0.2, cy = 0,
                use_phase_map = True,
                use_int_map = False,
                save_images = False):
  '''


    Parameters
    ----------
    diameter : int, float
        Desired diameter (in micron!) for the impurity circulation.
    omega : int, float
        Desired angular velocity (in rad/s !) for the impurity.
    act_time : int
        Acting time (in seconds!) for the DMD to be on.
    cx : int, float; optional
        x fine-tuning parameter. The default is 0.
    cy : int, float; optional
        y fine-tuning parameter. The default is 0.
    use_phase_map : bool; optional
        The default is True.
    use_int_map : bool; optional
        The default is False.
    save_images : Bool, optional
        The default is False.

  '''
```

What else can be changed (besides the inputs shown in the *docstring*):

- "num_img": same explanation as before (see **(2.a)** for explanation);

- "r": same explanation as before (see **(2.b)** for explanation). Warning: differently than for the Bragg beams, here we want the smallest possible light structure at the atoms' position to act as a spinning impurity.