Tutorial 1: First *in-silico* microscopy image

Subhamoy Mahajan

22 Dec, 2020

1. Generate the the PSF

   The point spread function (PSF) is generted using the following command

```
term$ python run_genpsf.py
```

   The maximum half-angle as seen from immersion oil, β, is calculated using $\beta = \sin^{-1}(NA/\mu)$, where NA is the numerical aperture of the objective lens and μ is the refractive index of immersion oil.

```
NA=1.3
meu=1.51
beta=np.arcsin(NA/meu)
```

   The PSF is calculated in a 3D grid with grid spacing of 0.1 in two lateral direction and 0.2 in axial direction.

```
dl = 0.1
dm = 0.1
dn = 0.2
```

   The dimension of the 3D box over which the PSF was calculated is $15 \times 15 \times 50$ nm^3.

```
Pl = 15
Pm = 15
Pn = 50
```

   A full-width-at-half-maximum (FWHM) scaling factor of 800 was used, for a wavelenght of 670 and 518 nm.

```
lambd = 518
gpsf.psf_gandy(beta,lambd,dl,dm,dn,Ll,Lm,Lm,fs,'PSF_gandy_lam'+str(lambd)+'_fs'
    +str(fs)+'.dat')
lambd = 670
gpsf.psf_gandy(beta,lambd,dl,dm,dn,Ll,Lm,Lm,fs,'PSF_gandy_lam'+str(lambd)+'_fs'
    +str(fs)+'.dat')
```

It will create two PSF files for wavelength 670 nm ("img100_lam670_fs800.dat") and 518 nm ("img100_lam670_fs800.dat").

2. Generate *in-silico* monochrome images.

   (a) Image data files

   The image data file containing resultant fluorescence intensity for each pixel can be calculated using the following commands,

   ```
   term$ ../../gen_mono -p parameters.dat -f dp100.gro -o img100
   ```

   ```
   term$ ../../gen_mono -p parameters.dat -f dp2000.gro -o img2000
   ```

   To use the PSF generated in the previous step following parameters were used in parameters.dat.

   ```
   f = 800
   lam1 = 670
   lam2 = 518
   dx = 0.1 0.1 0.2
   ```

   The dimension of PSF box used in parameter.dat can be smaller than the dimension of box for which the PSF data file was generated.

   ```
   Lpsf = 15 15 25
   ```

   The maximum box dimension for the structure files was found to be 24.3, therefore a maximum box length was chosen to be 25 nm.

   ```
   maxlen = 25 25 25
   ```

   The z axis was chosen as the optical axis for this tutorial, and the *in-silic* microscope was focused at $z = 15$ nm.

   ```
   focus_cor = 15
   opt_axis = 2
   ```

   The atoms that emit light (for each wavelength) was decided using the commands,

   ```
   lam_names1 = BB1 BB2 BB3 SC1 SC2 SC3
   lam_names2 = QPri Pri QSec Sec Ter
   ```

   Periodic boundary condition was applied in x, y and z directions,

   ```
   pbc = xyz
   ```

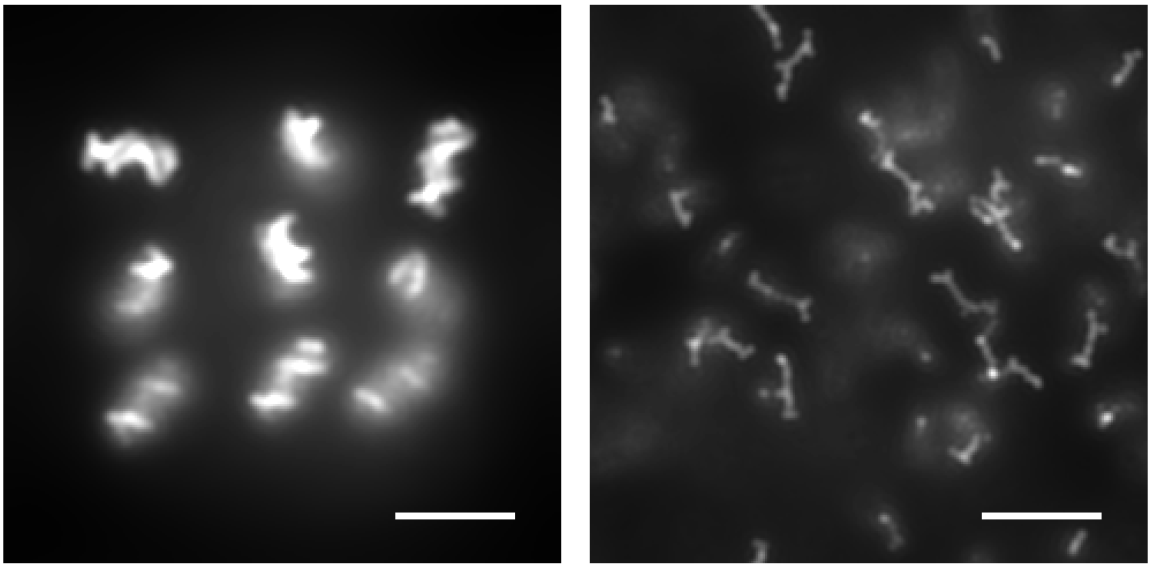
   Since the PSF files generated began with "PSF_gandy", the value of psfheader was taken to be "PSF_gandy".

   Running gen_mono, generate two pairs of files "img100_lam670_fs800.dat", "img100_lam518_fs800.dat", "img2000_lam670_fs800.dat", and "img2000_lam518_fs800.dat".

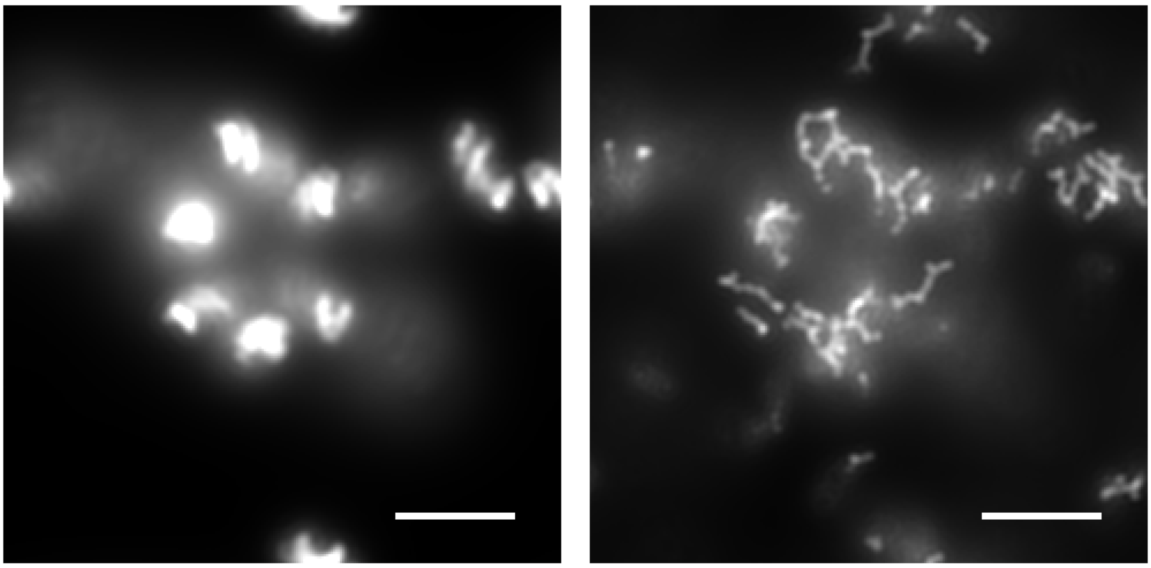
(b) Render grey-scale images

In-silico monochrome images can be rendeted using the following commands,

```
term$ python ../../render_mono.py -f img -p png_param.dat -t 100
```



```
term$ python ../../render_mono.py -f img -p png_param.dat -t 2000
```



To use the image data files generated in the previous step, png_param.dat has the following lines (size = second value in maxlen in parameters.dat),

```
fs = 800
lam1 = 670
lam2 = 518
size = 25
```

No time averaging was performed.

```
T = 1
```

Maximum intensity I_0 of 0.13 and 0.25 was used for lam1 and lam2 respectively.

```
lam1_I0 = 0.13
lam1_I0 = 0.25
```

A scale bar of 5 nm was added using,

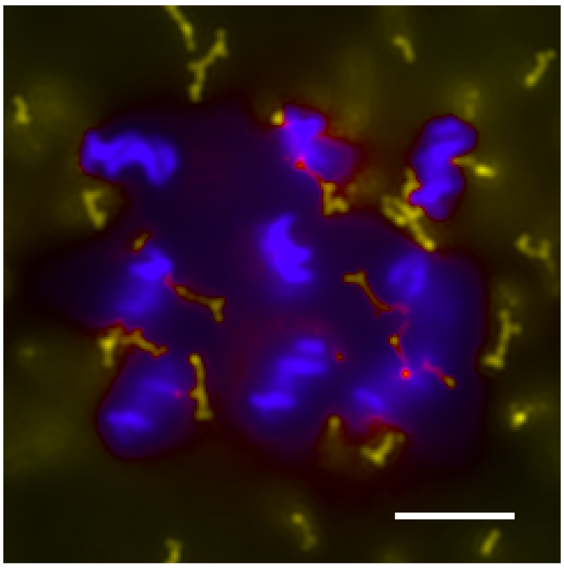
```
scale = 5
```

The parameters for hues are ignore while running render_mono.py. Four files are generated: "mono_img100_lam670_fs800_I0.13.png", "mono_img100_lam518_fs800_I0.25.png", "mono_img2000_lam670_fs800_I0.13.png", and "mono_img2000_lam518_fs800_I0.25.png"

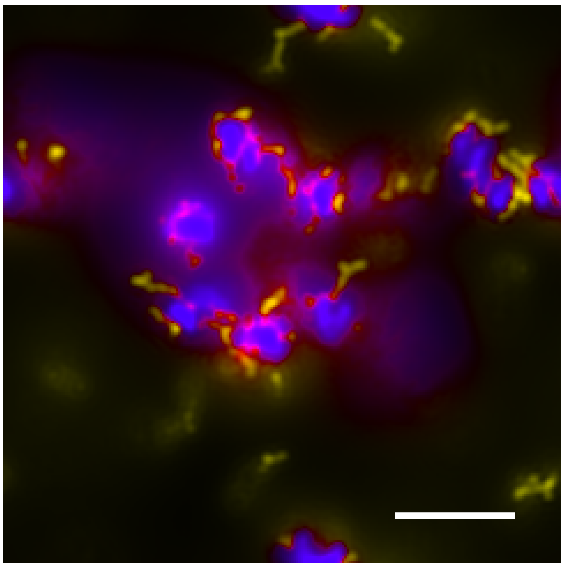
3. Generate colored *in-silico* microsocpy image.

   Colored *in-silico* microsocpy images can be generated using the following commands,

```
term$ python ../../mono2color.py -f img -p png_param.dat -t 100
```



```
term$ python ../../mono2color.py -f img -p png_param.dat -t 2000
```



In addition to parameters of png_param.dat mentioned in the previous section, mono2color.py also needs the hue values. The indigo (255°) and yellow (60°) hue was assigned to lam1 and lam2 respectively.

```
lam1_hue=255
lam2_hue=60
```

mono2color.py generates two files "img100_fs800_T1_I_0.13_0.25.png", and "img2000_fs800_T1_I_0.13_0.25.png".