DeepTrack - Example 6 - Tracking vesicles

March 10, 2019

Example 6: Tracking of vesicles 1.0

Example code to use DeepTrack to track fluorescent particles representing biological vesicle. Image data generated with the open bioimage informatics platform Icy (http://dx.doi.org/10.1038/nmeth.2075). See also Figure 5.

Note that this is an example file to track the case of SNR 4 and medium density. The same training file can be used to track other cases but the parameters frame_enhance, particle_radial_distance_threshold and particle_maximum_interdistance might have to be adjusted.

DeepTrack 1.0
Digital Video Microscopy enhanced with Deep Learning version 1.0
30 November 2018 © Saga Helgadottir, Aykut Argun & Giovanni Volpe
Soft Matter Lab

1. INITIALIZATION

```
In [1]: import deeptrack
```

2. PLAY VIDEO TO BE TRACKED

The video to be tracked is played.

Video file: DeepTrack - Example 6 - Vesicles SNR4 mid.mp4

Note that the video file must be in the same folder as this notebook.

```
In [2]: %%HTML

        <video width="400" height="400" controls>
        <source src="DeepTrack - Example 6 - Vesicles SNR4 mid.mp4" type="video/mp4">
        </video>
```

```
<IPython.core.display.HTML object>
```

3. CHECK IMAGE GENERATION ROUTINE

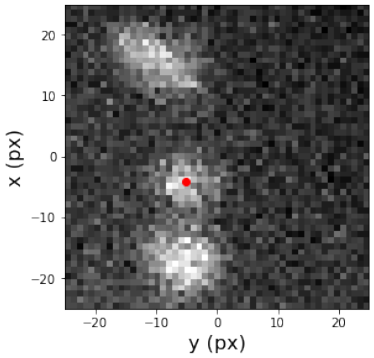
Here, we simulate images of multiple particles similar to those we want to track.

Comments:

1. The image_parameters_function is a lambda function that determines the kind of particle images for which the deep learning network will be trained. Tuning its parameters is the simplest way to improve the tracking performance.
2. The image_generator is a lambda function that works as image genrator. It does not need to be changed in most cases.
3. The parameter number_of_images_to_show determines the number of sample images that are shown.
4. The red symbol superimposed to the images represents the ground truth particle position.

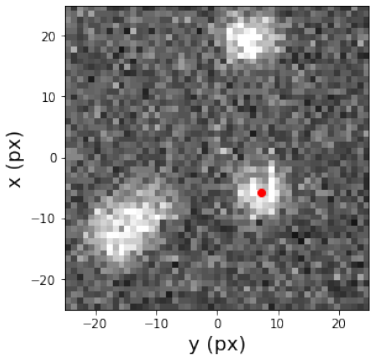
```
In [3]: ### Define image properties
        %matplotlib inline
        from numpy.random import randint, uniform, normal, choice
        from math import pi

        particle_number = 4
        first_particle_range = 10
        other_particle_range = 25
        particle_distance = 10


        def get_image_parameters_optimized():
            (particles_center_x, particles_center_y) = deeptrack.particle_positions(particle_num
            image_parameters = {}

            image_parameters['Particle Center X List'] = particles_center_x
            image_parameters['Particle Center Y List'] = particles_center_y
            image_parameters['Particle Radius List'] = uniform(2, 3, particle_number)
            image_parameters['Particle Bessel Orders List'] = [[1, ],
                                                               [1, ],
                                                               [1, ],
                                                               [1, ]]
            image_parameters['Particle Intensities List'] = [[uniform(0.4, 0.7, 1), ],
                                                             [uniform(0.4, 0.7, 1), ],
                                                             [uniform(0.4, 0.7, 1), ],
                                                             [uniform(0.4, 0.7, 1), ]]
            image_parameters['Image Half-Size'] = 25
            image_parameters['Image Background Level'] = uniform(.2, .4)
            image_parameters['Signal to Noise Ratio'] = uniform(2, 8)
            image_parameters['Gradient Intensity'] = uniform(0, 0.2)
            image_parameters['Gradient Direction'] = uniform(-pi, pi)
            image_parameters['Ellipsoid Orientation'] = uniform(-pi, pi, particle_number)
            image_parameters['Ellipticity'] = 1
```

```
    return image_parameters


image_parameters_function = lambda : get_image_parameters_optimized()

### Define image generator
image_generator = lambda : deeptrack.get_image_generator(image_parameters_function)

### Show some examples of generated images
number_of_images_to_show = 10

for image_number, image, image_parameters in image_generator():
    if image_number>=number_of_images_to_show:
        break

    deeptrack.plot_sample_image(image, image_parameters)
```



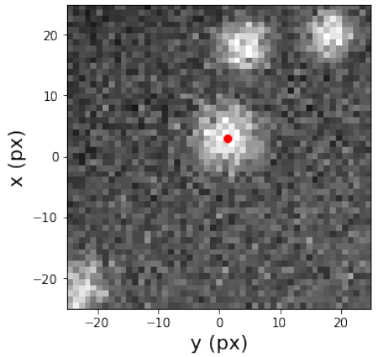
particle center x = -4.15 px
particle center y = -5.11 px
particle radius = 2.46 px
Bessel order = 1.00
particle intensity = 0.50
ellipsoidal_orientation = -1.88
ellipticity = 1.00

image half size = 25.00 px
image background level = 0.20
signal to noise ratio = 5.88
gradient intensity = 0.14
gradient direction = -2.38



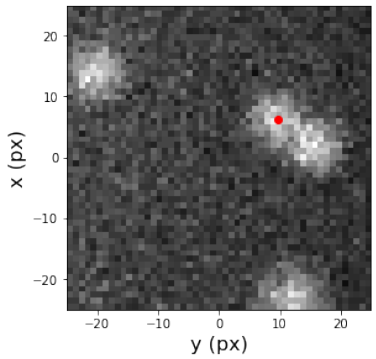
particle center x = -5.78 px
particle center y = 7.34 px
particle radius = 2.26 px
Bessel order = 1.00
particle intensity = 0.65
ellipsoidal_orientation = 2.01
ellipticity = 1.00

image half size = 25.00 px
image background level = 0.35
signal to noise ratio = 6.01
gradient intensity = 0.06
gradient direction = 2.47



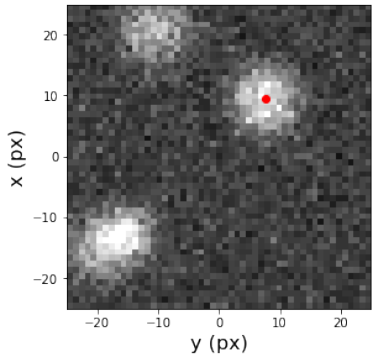
particle center x = 3.03 px
particle center y = 1.33 px
particle radius = 2.64 px
Bessel order = 1.00
particle intensity = 0.67
ellipsoidal_orientation = -0.48
ellipticity = 1.00

image half size = 25.00 px
image background level = 0.31
signal to noise ratio = 7.22
gradient intensity = 0.15
gradient direction = -1.09



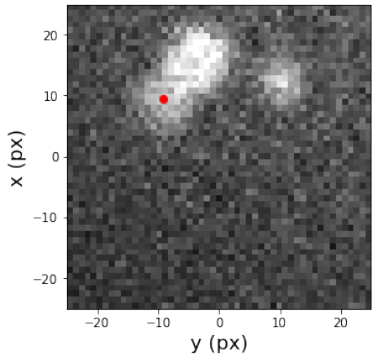
particle center x = 6.26 px
particle center y = 9.79 px
particle radius = 2.40 px
Bessel order = 1.00
particle intensity = 0.49
ellipsoidal_orientation = 2.18
ellipticity = 1.00

image half size = 25.00 px
image background level = 0.24
signal to noise ratio = 7.66
gradient intensity = 0.15
gradient direction = 2.93



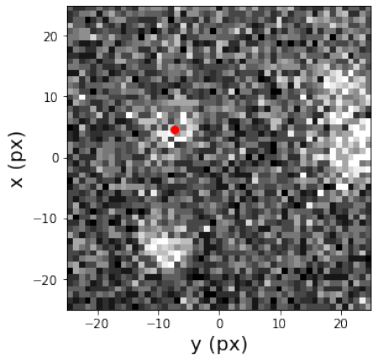
particle center x = 9.40 px
particle center y = 7.71 px
particle radius = 2.75 px
Bessel order = 1.00
particle intensity = 0.69
ellipsoidal_orientation = 0.44
ellipticity = 1.00

image half size = 25.00 px
image background level = 0.24
signal to noise ratio = 7.66
gradient intensity = 0.00
gradient direction = 0.70



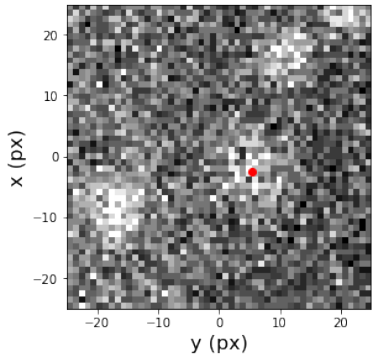
particle center x = 9.49 px
particle center y = -9.20 px
particle radius = 2.36 px
Bessel order = 1.00
particle intensity = 0.45
ellipsoidal_orientation = -2.53
ellipticity = 1.00

image half size = 25.00 px
image background level = 0.29
signal to noise ratio = 7.72
gradient intensity = 0.14
gradient direction = 1.05



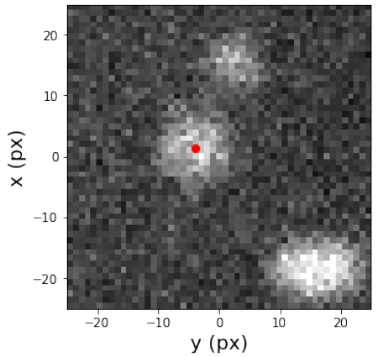
particle center x = 4.62 px
particle center y = -7.30 px
particle radius = 2.38 px
Bessel order = 1.00
particle intensity = 0.44
ellipsoidal_orientation = -0.34
ellipticity = 1.00

image half size = 25.00 px
image background level = 0.34
signal to noise ratio = 3.28
gradient intensity = 0.16
gradient direction = 1.17



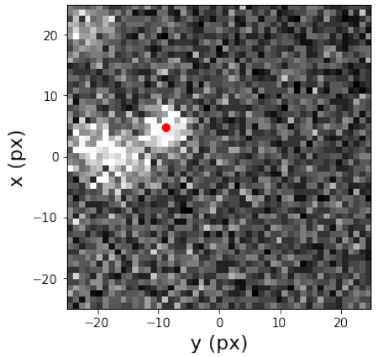
particle center x = -2.50 px
particle center y = 5.44 px
particle radius = 2.93 px
Bessel order = 1.00
particle intensity = 0.41
ellipsoidal_orientation = -1.05
ellipticity = 1.00

image half size = 25.00 px
image background level = 0.40
signal to noise ratio = 3.62
gradient intensity = 0.19
gradient direction = -3.01



particle center x = 1.32 px
particle center y = -3.99 px
particle radius = 2.84 px
Bessel order = 1.00
particle intensity = 0.58
ellipsoidal_orientation = -1.22
ellipticity = 1.00

image half size = 25.00 px
image background level = 0.25
signal to noise ratio = 7.16
gradient intensity = 0.08
gradient direction = 3.01



particle center x = 4.76 px
particle center y = -8.85 px
particle radius = 2.04 px
Bessel order = 1.00
particle intensity = 0.70
ellipsoidal_orientation = -1.98
ellipticity = 1.00

image half size = 25.00 px
image background level = 0.30
signal to noise ratio = 3.80
gradient intensity = 0.10
gradient direction = -2.14

4. USE A PRETRAINED DEEP LEARNING NETWORK

The pretraiend networks saved in the files "DeepTrack - Example 6 - Vesicles SNR4 mid.h5" and DeepTrack - Example 4 - Particles - Pretrained network.h5 are loaded and their performance on selected video is tested.

Video file: DeepTrack - Example 6 - Vesicles SNR4 mid.mp4

Note that the pretrained network files and the video file must be in the same folder as this notebook.

Comments:

1. number_frames_to_be_tracked can be changed to track different number of frames. If number_frames is equal to 0 then the whole video is tracked.
2. box_half_size is half the size of the box to be scanned over the frames. The resulting sample should be comparable to the training image.
3. box_scanning_step is the step that is used to scan the box over the frame. It can be increased for higher accuracy or decreased for lower computational time.

4. frame_normalize gives the option to normalize the frames. Set to 1 to normalize, 0 otherwise.
5. frame_enhance gives the option to enhance the frames. Set to 1 to track the original frames.

In [4]:
```
### Define the video file to be tracked
video_file_name = 'DeepTrack - Example 6 - Vesicles SNR4 mid.mp4'

### Define the number of frames to be tracked
number_frames_to_be_tracked = 2

### Define the size of the box to be scanned over the frames
box_half_size = 10

### Define the scanning step over the frame
box_scanning_step = 3

### Preprocess the images
frame_normalize = 0
frame_enhance = 5

### Load the pretrained network
saved_network_file_name = 'DeepTrack - Example 6 - Tracking vesicles.h5'
network = deeptrack.load(saved_network_file_name)

### Track the video
(number_tracked_frames, frames, predicted_positions_wrt_frame, predicted_positions_wrt_b
    video_file_name,
    network,
    number_frames_to_be_tracked,
    box_half_size,
    box_scanning_step,
    frame_normalize,
    frame_enhance)
```

```
Using TensorFlow backend.
```

5. SHOW EXAMPLES OF TRACKED SCANNING BOXES

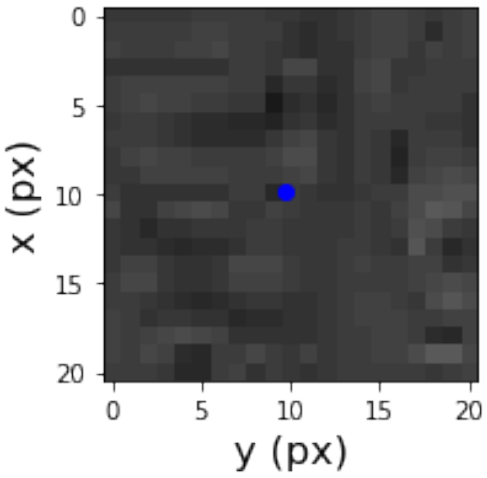
The tracked scanning boxes are plotted over a range of frames, rows and columns.

Comments:

1. frame_to_be_shown can be changed to view different frames.
2. rows_to_be_shown can be changed to view different rows of each of the frames.
3. columns_to_be_shown can be changed to view different columns of each of the frames.
4. The blue symbol is the deep learning network prediction for the position (x, y).

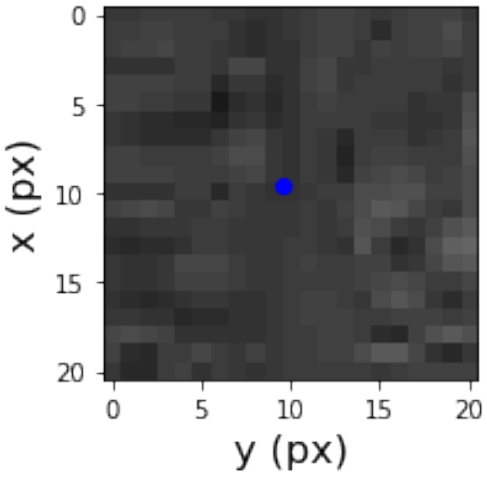
In [5]:
```
### Define frames, rows and columns of the samples to be shown
frames_to_be_shown = range(1)
```

```
rows_to_be_shown = range(15,20)
columns_to_be_shown = range(5,10)

### Show boxes
deeptrack.plot_tracked_scanning_boxes(
    frames_to_be_shown,
    rows_to_be_shown,
    columns_to_be_shown,
    boxes_all,
    predicted_positions_wrt_box)
```



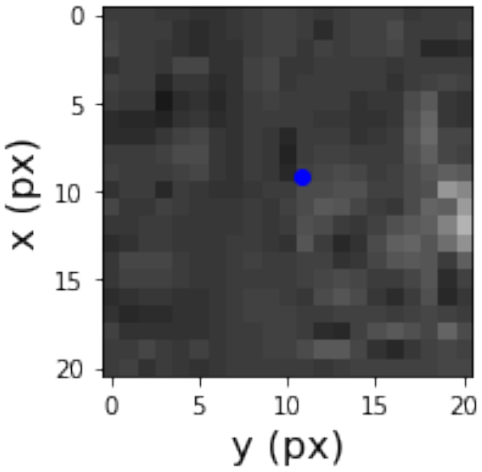
frame = 0
row = 15
column = 5

particle center x = 9.87 px
particle center y = 9.74 px
particle radius = 4.91 px



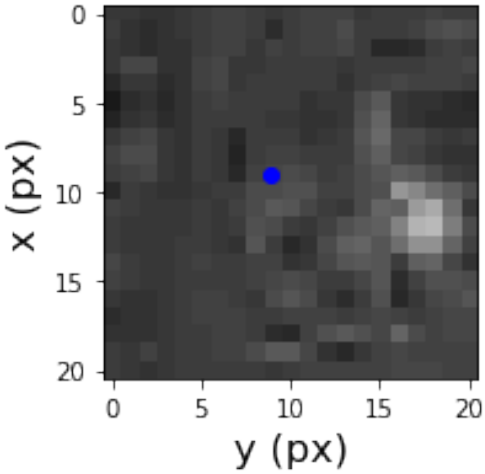
frame = 0
row = 15
column = 6

particle center x = 9.53 px
particle center y = 9.60 px
particle radius = 4.92 px



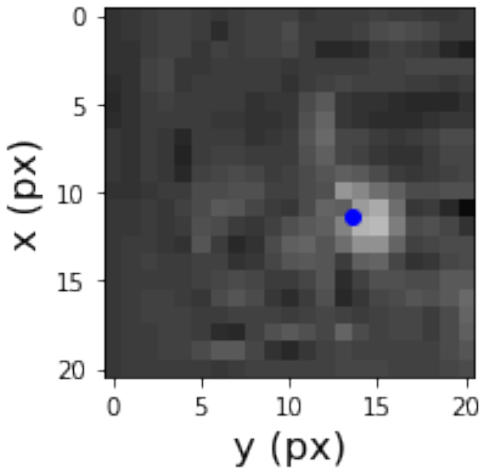
frame = 0
row = 15
column = 7

particle center x = 9.15 px
particle center y = 10.84 px
particle radius = 4.67 px



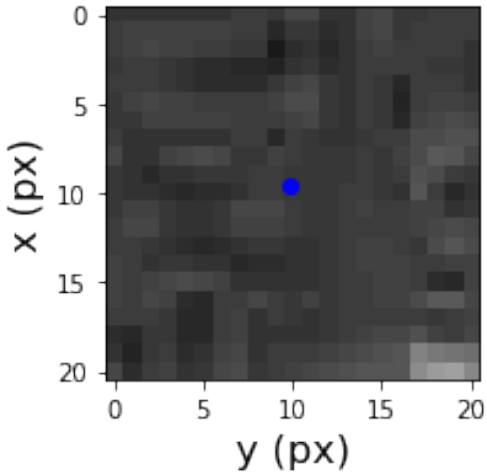
frame = 0
row = 15
column = 8

particle center x = 9.05 px
particle center y = 8.88 px
particle radius = 4.68 px



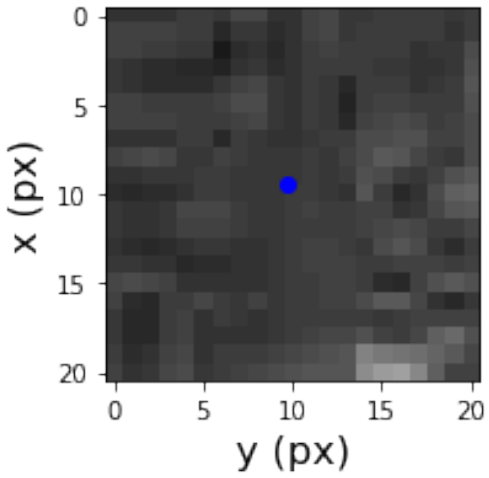
frame = 0
row = 15
column = 9

particle center x = 11.32 px
particle center y = 13.63 px
particle radius = 4.28 px



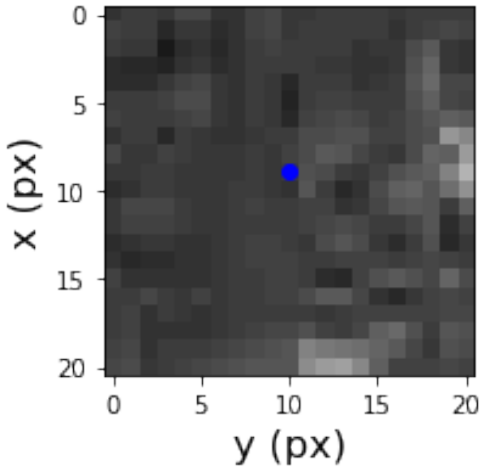
frame = 0
row = 16
column = 5

particle center x = 9.53 px
particle center y = 9.81 px
particle radius = 4.92 px



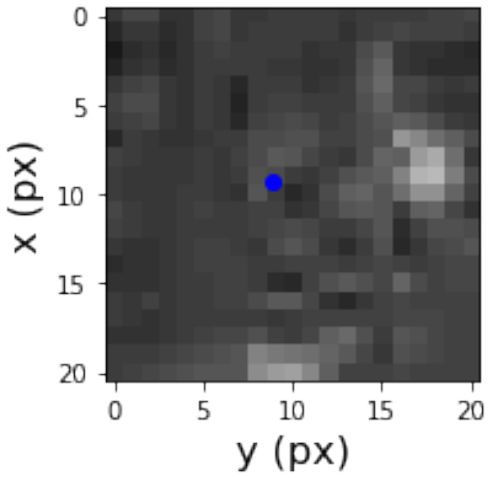
frame = 0
row = 16
column = 6

particle center x = 9.49 px
particle center y = 9.72 px
particle radius = 4.97 px



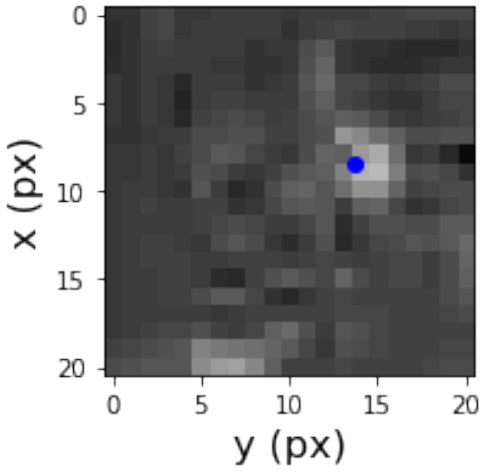
frame = 0
row = 16
column = 7

particle center x = 8.90 px
particle center y = 10.04 px
particle radius = 4.87 px



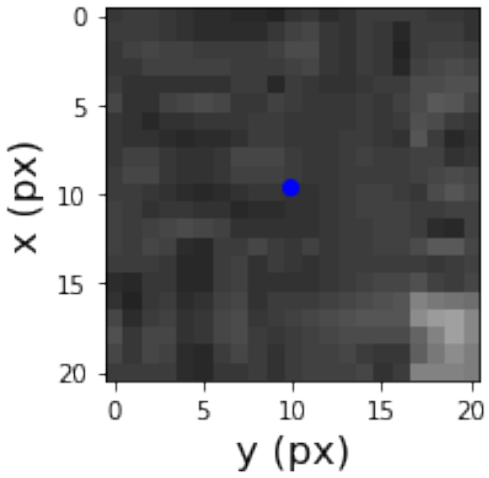
frame = 0
row = 16
column = 8

particle center x = 9.32 px
particle center y = 8.89 px
particle radius = 4.77 px



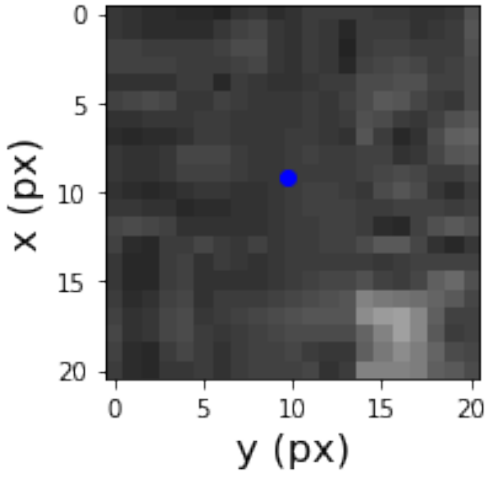
frame = 0
row = 16
column = 9

particle center x = 8.41 px
particle center y = 13.70 px
particle radius = 4.21 px



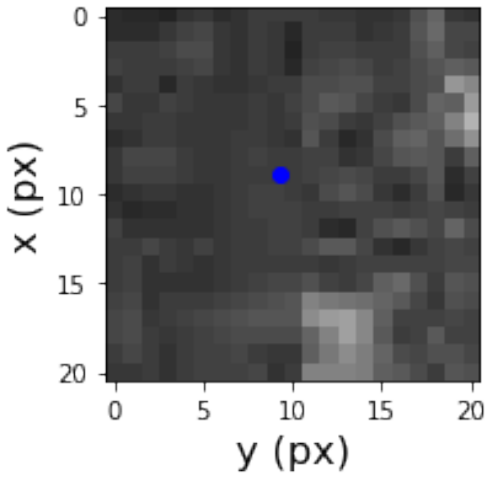
frame = 0
row = 17
column = 5

particle center x = 9.51 px
particle center y = 9.81 px
particle radius = 4.96 px



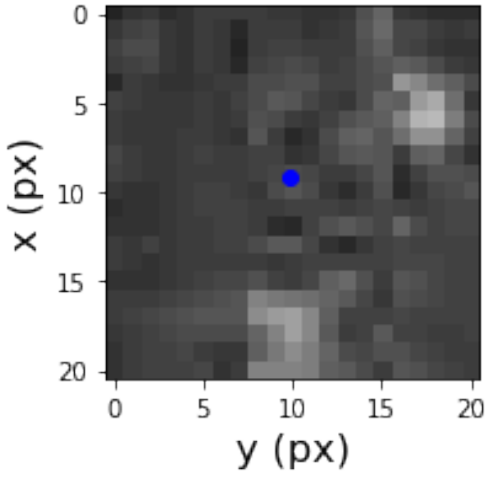
frame = 0
row = 17
column = 6

particle center x = 9.19 px
particle center y = 9.69 px
particle radius = 4.93 px



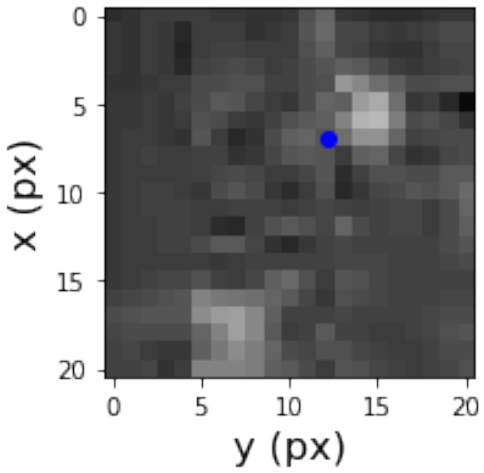
frame = 0
row = 17
column = 7

particle center x = 8.85 px
particle center y = 9.32 px
particle radius = 4.96 px



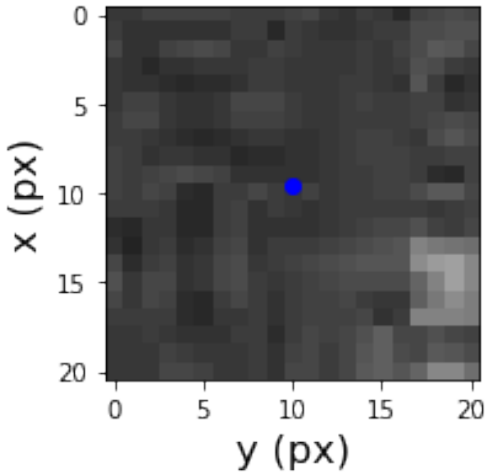
frame = 0
row = 17
column = 8

particle center x = 9.22 px
particle center y = 9.87 px
particle radius = 4.98 px



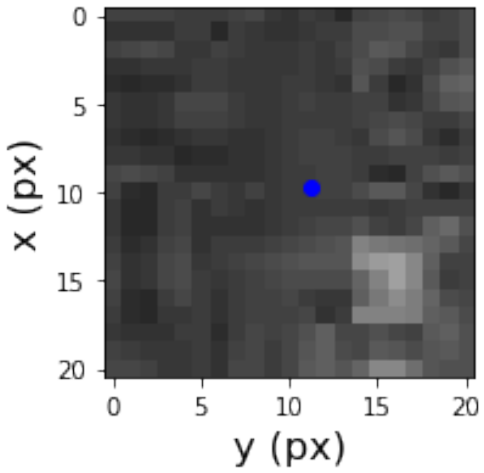
frame = 0
row = 17
column = 9

particle center x = 6.95 px
particle center y = 12.15 px
particle radius = 5.13 px



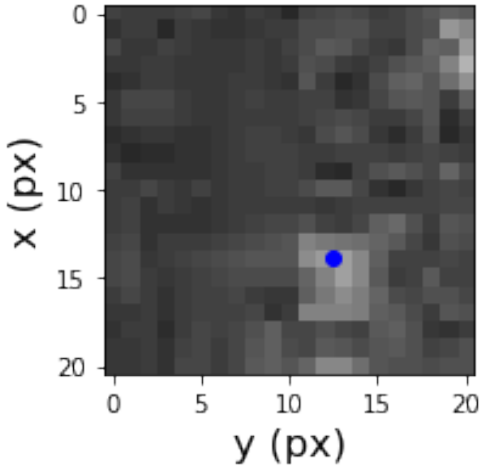
frame = 0
row = 18
column = 5

particle center x = 9.57 px
particle center y = 9.95 px
particle radius = 4.95 px



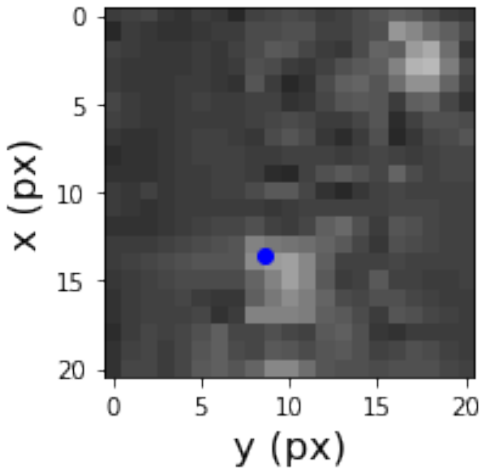
frame = 0
row = 18
column = 6

particle center x = 9.65 px
particle center y = 11.19 px
particle radius = 5.05 px



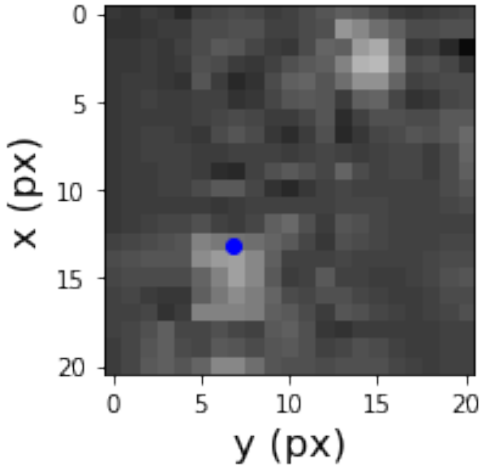
frame = 0
row = 18
column = 7

particle center x = 13.80 px
particle center y = 12.51 px
particle radius = 4.65 px



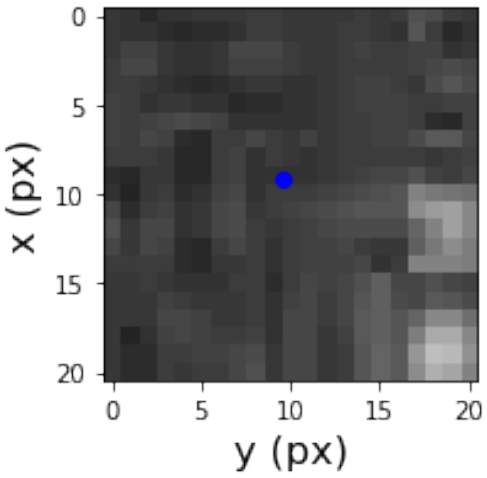
frame = 0
row = 18
column = 8

particle center x = 13.59 px
particle center y = 8.60 px
particle radius = 4.33 px



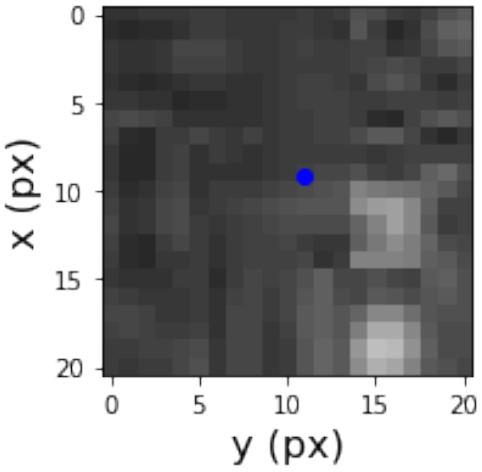
frame = 0
row = 18
column = 9

particle center x = 13.15 px
particle center y = 6.88 px
particle radius = 4.95 px



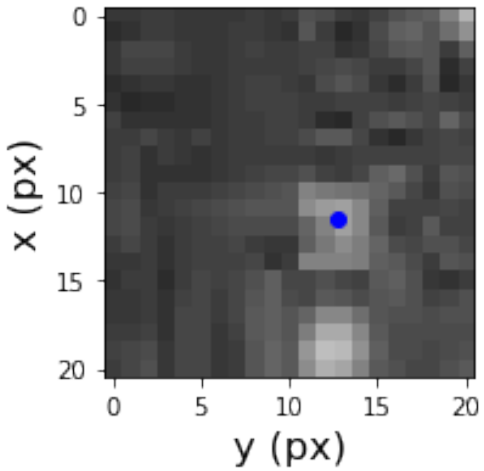
frame = 0
row = 19
column = 5

particle center x = 9.15 px
particle center y = 9.51 px
particle radius = 4.93 px



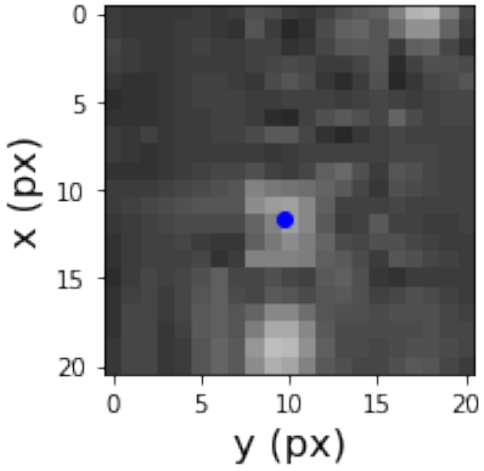
frame = 0
row = 19
column = 6

particle center x = 9.12 px
particle center y = 10.94 px
particle radius = 4.84 px



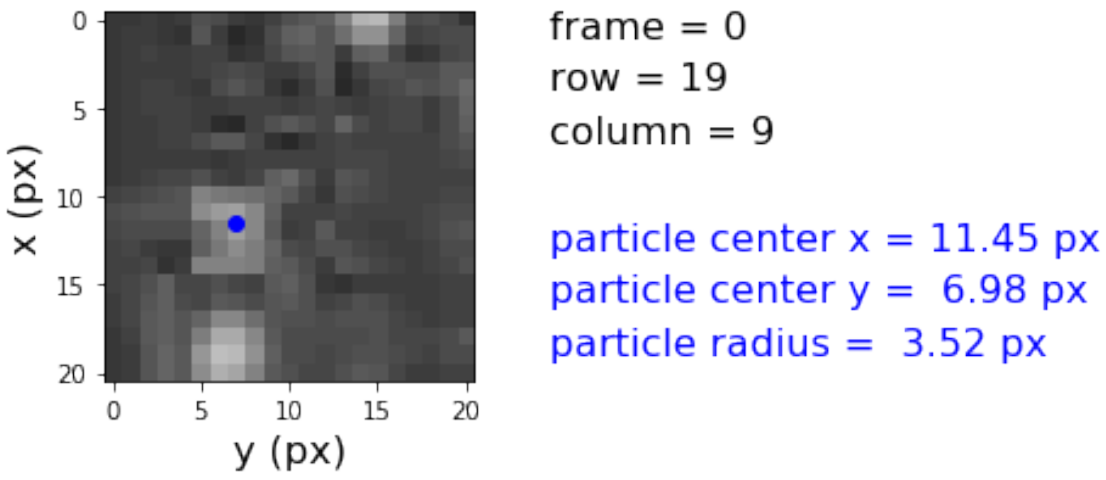
frame = 0
row = 19
column = 7

particle center x = 11.54 px
particle center y = 12.73 px
particle radius = 3.11 px



frame = 0
row = 19
column = 8

particle center x = 11.62 px
particle center y = 9.76 px
particle radius = 1.73 px



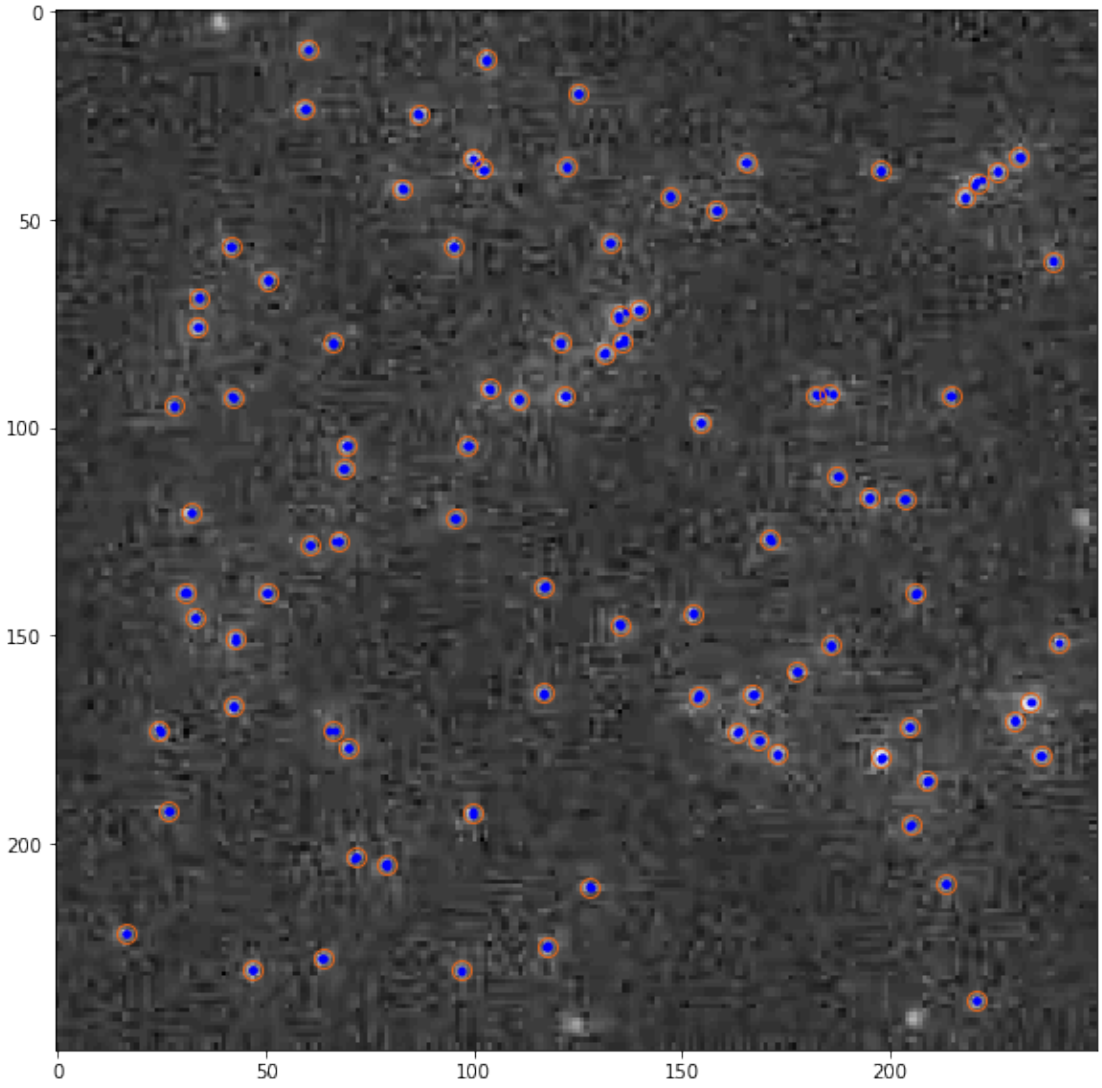
6. SHOW EXAMPLES OF TRACKED FRAMES

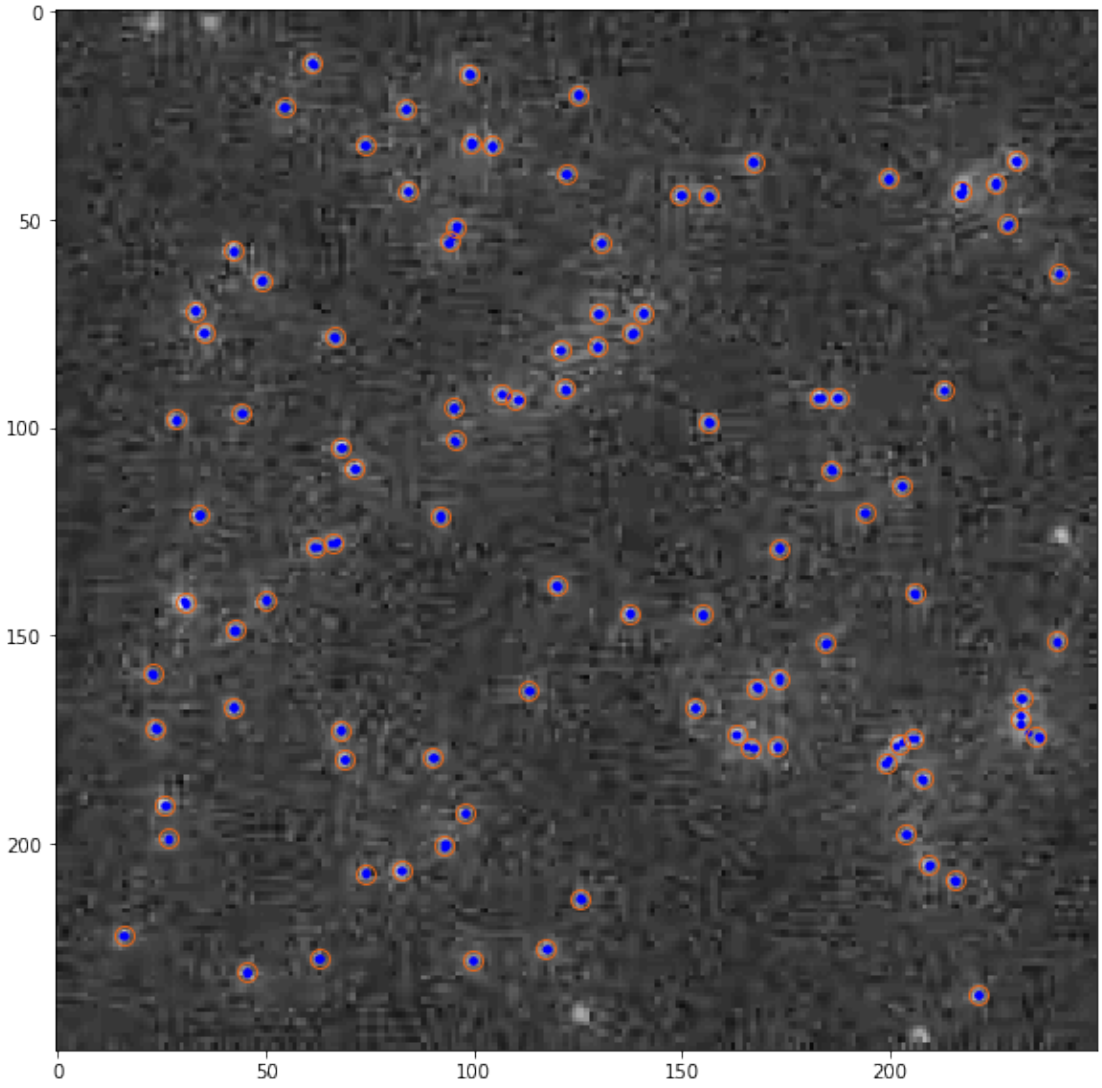
The tracked frames are shown.

Comments:

1. particle_radial_distance_threshold can be changed to choose which prediction points (blue dots) are to be used to calculate the centroid positions (orange circles). We used 3 pixels.
2. particle_maximum_interdistance can be changed to choose what predicted points (blue dots) belong to the same particle. We used 2.8 pixels.

```
In [6]: ### Define minimum radial distance from the center of the scanning boxes
        %matplotlib inline
        particle_radial_distance_threshold = 3

        ### Define the minumum distance between predicted scanning points for them belonging to
        particle_maximum_interdistance = 2.8

        ### Visualize tracked frames
        deeptrack.show_tracked_frames(
            particle_radial_distance_threshold,
            particle_maximum_interdistance,
            number_tracked_frames,
            frames,
            predicted_positions_wrt_frame)
```


0
50
100
150
200
0
50
100
150
200


0
50
100
150
200
0
50
100
150
200