Title: Pixel level and Subpixel level particle detection

Project Assignment 3

Student Names : Guruprasad Raghavan, Philip Lee , Priyanka Raja

4/4/2016

Introduction:

For this project, we performed both pixel-level and subpixel-level particle detection. The pixel level detection was performed by first filtering the original image with a Gaussian filter, then determining the local max/min by checking the center of submatrix, then associating the local max/min with Delaunay triangulation, and then selecting statistically significant associations with T-test to obtain particles' coordinates and relative intensities. The subpixel-level detection was performed by filtering the original image with a Gaussian filter again, then oversampling the resulting image with linear interpolation, and then fitting a Gaussian curve to obtain particles' sub-pixel coordinates. To assess the performance of subpixel-level detection, synthetic image was created by combining the result from pixel-level detection with realistic white noise and then applying subpixel-level detection algorithm. The coordinates from the synthetic image and the coordinates from the detection algorithm were then compared to determine accuracy and precision.

Code execution:

Open Main.m in Matlab.
The code is composed of multiple sections.
Each section can be run one by one with "Run Section" found at Editor>Run.

For B.2.1, run the first and second section sequentially. The resulting dark noise mean and standard deviation can be called with bkgdMean and bkgdStd, respectively.

For B.2.2, run the third section. The number of local maxima/minima found by the 3x3 or 5x5 submatrix can be found by numel(localMax_3(:,1)), numel(localMin_3(:,1)), numel(localMax_5(:,1)), and numel(localMin_5(:,1)), respectively.

For B.2.3, run the fourth section. The number of Delaunay triangles found will be displayed in the Command Window.

For B.2.4, run the fifth section.
The section that evaluates the statistically significant local maxima's is titled: %% Statistical selection of local maxima.

Having set the Quantile score, the following code snippet, evaluates the statistical significance of each local maxima, and the output is a finalMaxima matrix, which contains the coordinates of the local Maxima as well as its intensities.

```
[finalMaxima] = statisticalTest(Quantile, associationMat, localMin_3, Tri_indices, bkgdStd, I);
```

The script that follows can be used to test the algorithm implemented, by overlaying the original image and the detected points (as included in the finalMaxima matrix).

```
figure,
imshow(I,[])
hold on
scatter(finalMaxima(:,2),finalMaxima(:,1));
```

B.2.1 Calibration of Dark Noise

To determine the dark noise of the image, we calculated the mean and standard deviation of the background. Multiple regions in the background were selected and the intensity values from all regions were collected into a single array. The mean and standard deviation of this array was then calculated. Such dark noise calibration was done for each frame. For the first frame, the dark noise was found to have a mean of 305.1188 and standard deviation of 24.5577 when the background regions were selected as in Figure 1.

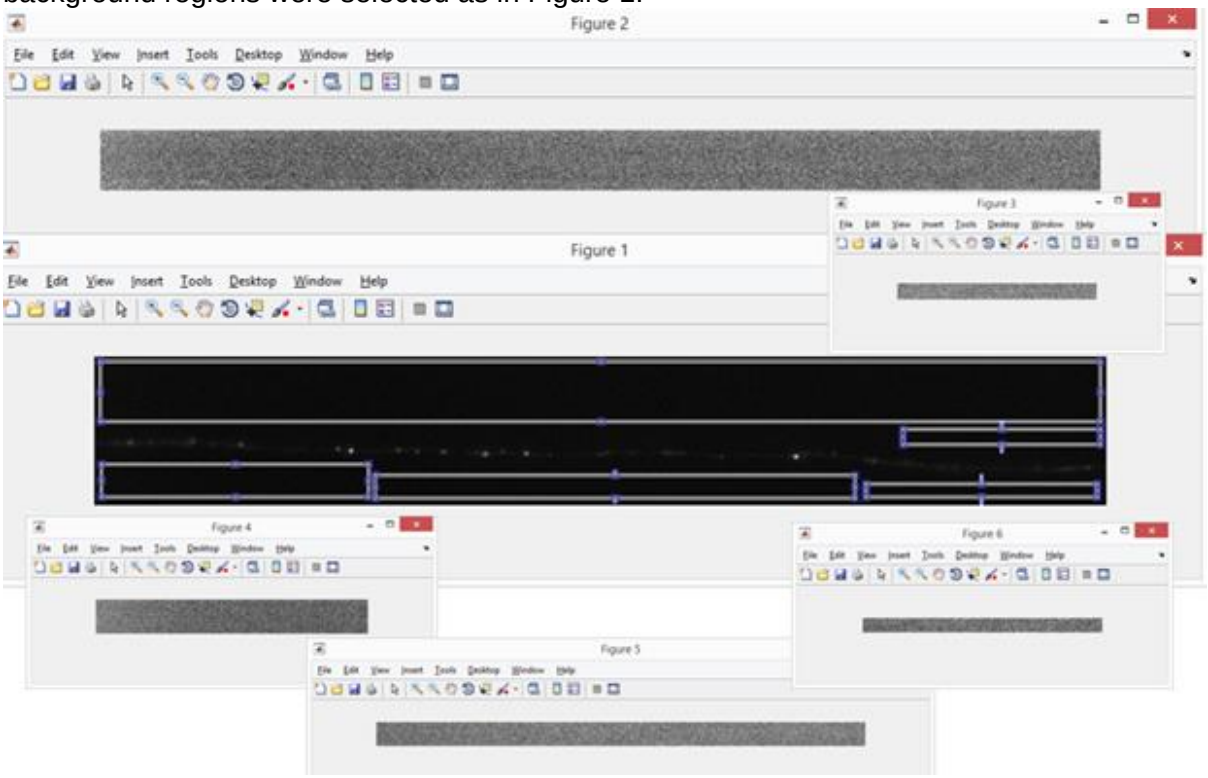


Figure 1. Multiple regions were selected in background. All Intensity values were collected from these regions and the mean and standard deviation were subsequently calculated.

B.2.2 Detection of Local Maxima and Local Minima

The original image had significant speckles reminiscent of white noise. To properly find local maxima and minima, the original image was first filtered with a Gaussian filter.

Gaussian Filtering

We filtered the original image with a Gaussian mask with sigma = (radius/3)/pixel-size and with mask size of NxN, where N was set to 2*ceil(3*sigma)+1 in order to obtain an odd mask with minimal truncation. Applying this filter significantly reduced the nose, as apparent by the absence of speckles in the filtered image as shown in figure 2.

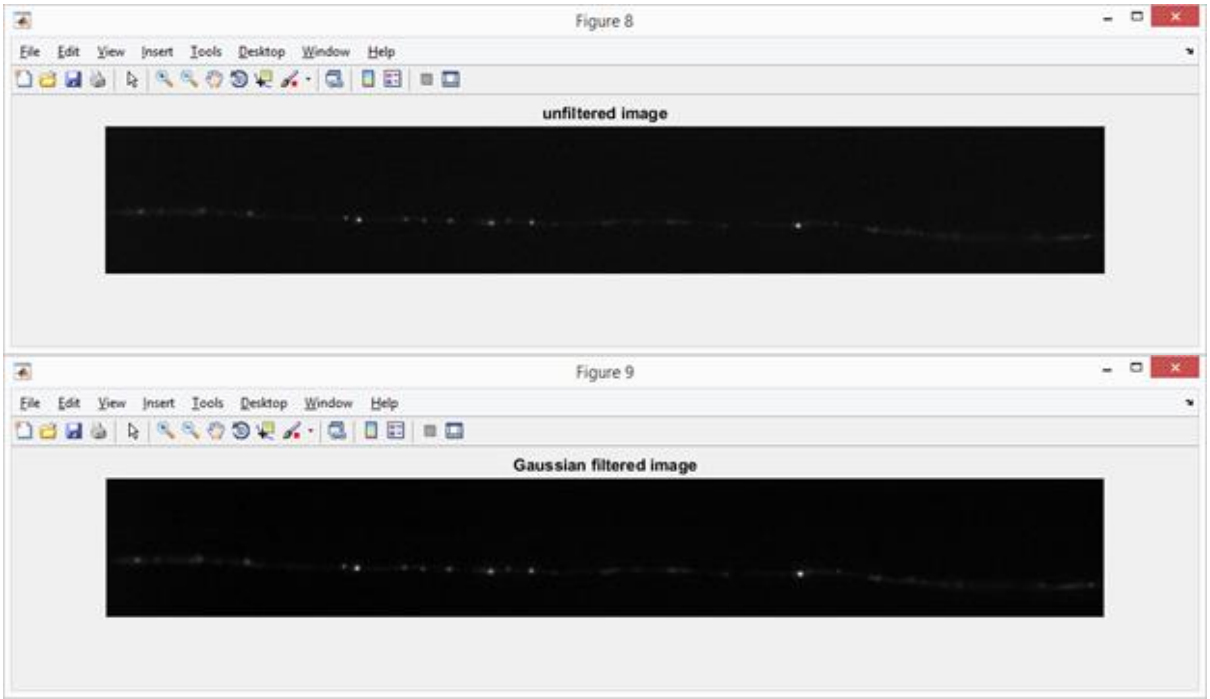


Figure 2. The original unfiltered image was heavily speckled. These speckles were no longer observable after Gaussian filtering, which implied good filtering.

Finding local maxima and minima

The local maxima and minima were found by using a 3x3 or 5x5 submatrix. If the center of the submatrix had the highest or lowest value within the submatrix, that center was assigned local maxima or minima, respectively. For the first frame, when a 3x3 submatrix was used, 6179 local maxima and 6406 local minima were found. When a 5x5 submatrix was used, 4502 local maxima and 4334 local minima were detected.

B.2.3 Establishing Local Association of Maxima and Minima

On determining the local maxima's and local minima's, by using a 3x3 or a 5x5 mask, the next step in the pipeline was to associate each local maximum to its nearest local minima. This was implemented by the following procedure:

1. *Perform delaunay triangulation of local minima:*

   Delaunay triangulation is better known as the dual of the voronoi pattern determination. The Number of triangles found this way was 12771.

2. *Associate each local maxima obtained to one of the triangles calculated*

   This association was established by looping over each triangle obtained, and by using the MATLAB built-in function "inpolygon", which checked if a local maxima was present in any of the evaluated triangles. A matrix data-structure is used to store all these associations (local Maxima and its corresponding triangle).

B.2.4
Having determined a large number of local maxima using the 3x3/ 5x5 mask, we have to select significant local maximas. In order to do so, we implement hypothesis testing.

A local Maxima is significant only if the difference between the local maxima intensity and its nearest local minima(s) is high. Mathematically, we can formulate this as follows:

I = IlocalMax-Ibkgd Here, I_bkgd is the average of the 3 local minima associated with a specific local maxima (point of interest) [where, Ibkgd = (Ibkgd1 + Ibkgd2+Ibkgd3)/3]. The association is achieved in question B.2.3, first by performing a delaunay triangulation of the local minima, followed by the association with local maxima's.

The 2 hypotheses are as follows:

1. Null Hypothesis: Ho= IlocalMax-Ibkgd = 0. If the point abides by this hypothesis, the point is classified as insignificant.
2. Alternate Hypothesis: H1=(IlocalMax-Ibkgd)>0 If the point follows this hypothesis, it is classified as significant.

To solve this further, we implement the T-test.

T = (IlocalMax - Ibkgd)/I
T ~ t(0,1) . This T-statistic is of infinite degree of freedom, as the sample points used to make a decision is = 1.

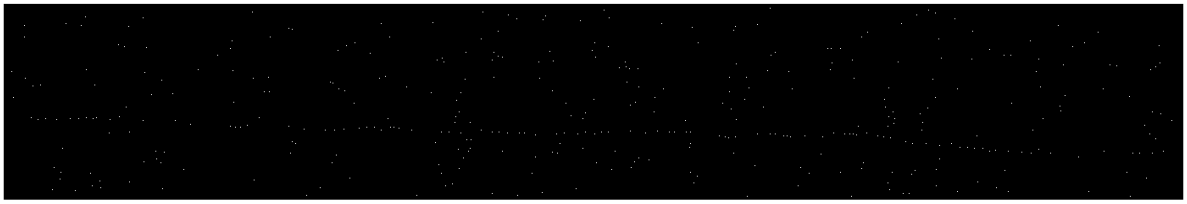
Definition of variables used in the evaluation of the T-statistic:

Variable	Definition
IlocalMax	Local Maxima intensity (considering one local maxima)
Ibkgd	Ibkgd = (Ibkgd1 + Ibkgd2+Ibkgd3)/3 (average of the 3 local minima associated with a specific local maxima)
I	IlocalMax-Ibkgd
I	2I = 2IlocalMax + 2 Ibkgd/3 ; Here, 2IlocalMax is evaluated to zero, while 2 Ibkgdis obtained from the standard deviation of the cropped background.

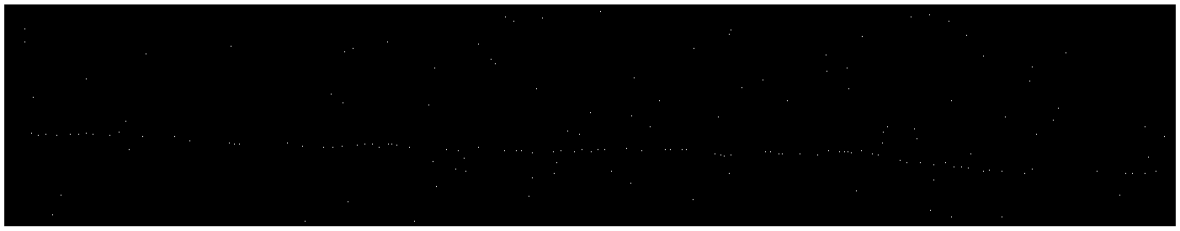
Based on the Quantile score chosen, we can make a decision as to whether a point is statistically significant or not. I've chosen a quantile score = 6.0, to reduce the error of misclassification of a local Maxima.

Statistical significant points, at different Quantile scores:

Quantile = 4.0



Quantile = 5.0

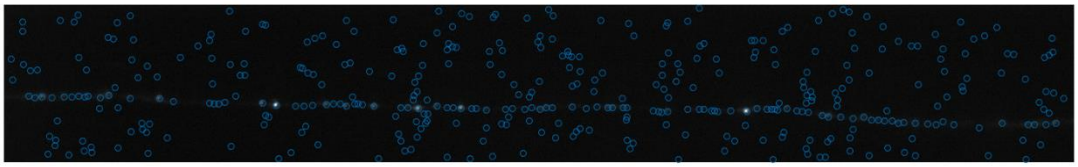


Quantile = 6.0

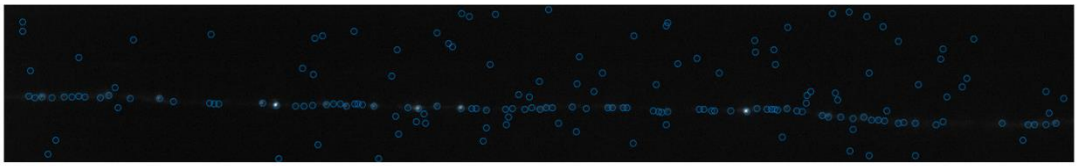
Quantile = 7.0

Overlay Detected points and original Image:

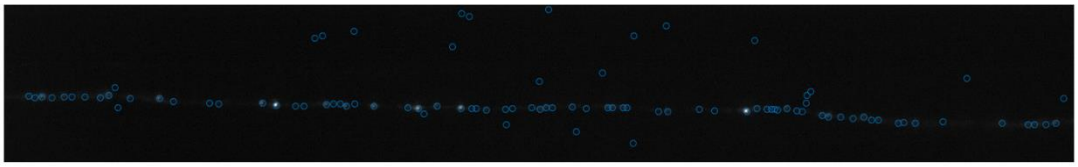
Quantile = 4.0



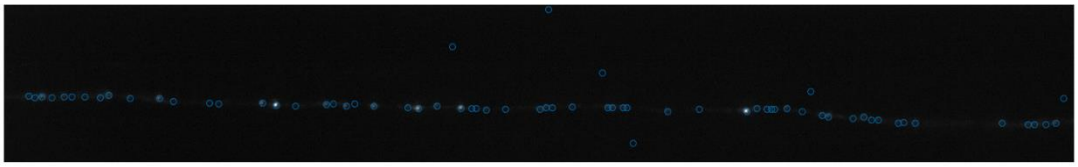
Quantile = 5.0



Quantile = 6.0



Quantile = 7.0



B3: Sub-pixel Resolution particle detection

B3.1: Creating a synthetic image from particles detected from B2.4

Summary:
The synthetic image is created by using the detected particles that are detected in B2.4. These particles are set to 1 and the background is set to zero. This raw image is convolved with a Gaussian kernel chosen to approximate the PSF, in order to create a synthetic image. White noise is generated by choosing random values from a normal distribution with mean equal to the average background noise and standard deviation is set to be 0.25 times the average intensity of the detected particles. White noise is generated from the aforementioned noise distribution to create an image I_noise of the same size as that of the original image (I). The noisy image is now added to the synthetic image to simulate an actual noisy image of signal to noise ratio of 4. This process is repeated with different standard deviations in the noise distributions thereby resulting in images

with different SNR ratios. The results are shown below, the image looks less sharper(more fuzzy) with a higher SNR as should be expected.

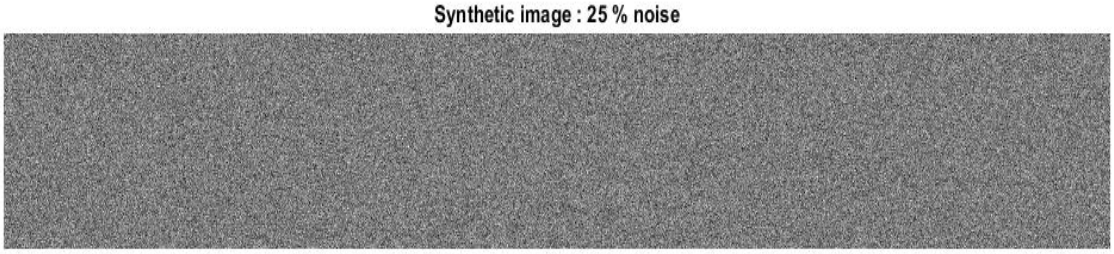
Code Execution Instructions:

In order to generate the synthetic image run the "`Creation of synthetic Image`" section in the `Main. m` file. This will generate the synthetic image `I_synthetic`.

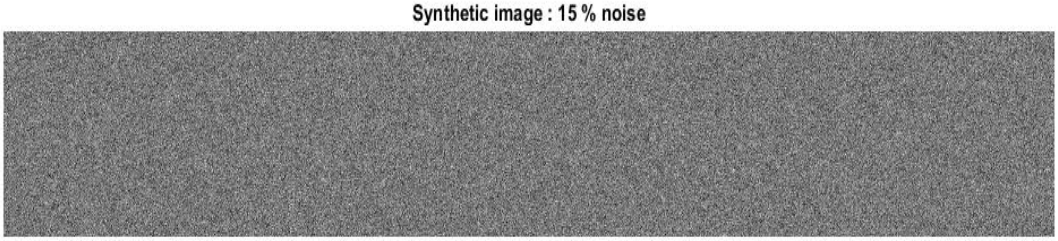
- The image is obtained by calling the `createSynthetic.m` function that requires as input the following :
    - *finalMaxima*: The x and y co-ordinates of the detected particles along with their intensity values.
    - *I* : original image.
    - *bkgdMean/noise_mean*: average background noise previously calculated.
    - *noise* : 10-25% of the average intensity that is assigned as the level of noise , Eg: 0.25 (indicates a noise level of 25% of the mean intensity of the detected particles).

Results: Set of synthetic images for different SNR ratios.

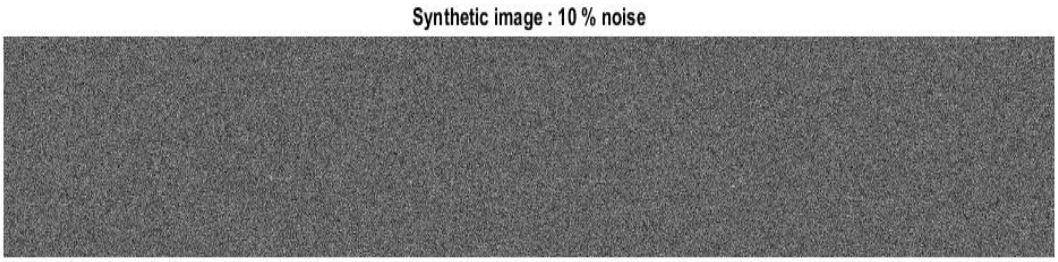
1. 25% Noise (SNR :4)



2. 15% Noise (SNR : ~ 7)



3. 10% Noise (SNR : 10)



B3.2 Sub-pixel detection algorithm using oversampling

Summary: In order to perform subpixel detection on the original image (first frame of the video). We implement the Gaussian fitting algorithm as described by Cheezum et al. The co-ordinates of the particles that were previously detected are obtained from the `detectMaxima.mat` file. Only the pixel co-ordinates of the detected particles and their neighbors (3x3 neighborhood) are oversampled. Gaussian fitting is carried out in this oversampled image space. The Gaussian kernel sigma = (PSF radius /3)/13nm. This Gaussian kernel is swiped across the oversampled image space and the pixel co-ordinates that minimizes the least square fit with the Gaussian are returned as sub pixel co-ordinates of the detected particles.

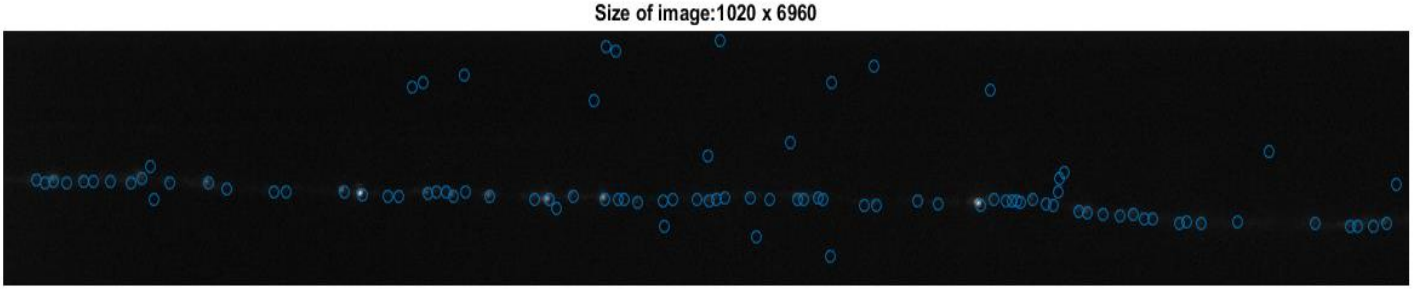
Code Execution Instructions:
In order to generate the synthetic image run the "`Subpixel Detection`" section in the `Main. m` file. This will generate the `finalsubPixelMaxima` variable that contain the pixel co-ordinates in the oversampled image(co-ordinates corresponding to when the whole image is resized) as well as the co-ordinates of the detected particles in the oversampled version of the image that only contained the detected particle pixel and its neighbors in a 3x3 neighborhood.

- The co-ordinates are obtained by running the `subPixelDetection.m` function which will execute when the aforementioned section is run in the main program. This function requires as input the following :
    - *finalMaxima*: Load `detectMaxima.mat` which contains contains the x, y co-ordinate of detected particles and the corresponding intensity value
    - *I* : Original Image
    - *I_syntheticNoNoise*: This is an optional parameter used in the extra credit section run in the Main.m function. This parameter is an image which contains only the detected particle intensities with no added noise (raw image).

The outputs generated:

- subPixelDetect : Detected sub-pixel co-ordinates that is returned by using gaussian fitting.
- realSubPixelMaxima : The actual sub-pixel co-ordinates obtained from the
- raw image.

Results: The following image shown below is the resulting subpixel co-ordinates in the oversampled image.



B3.3 : Benchmarking subpixel detection algorithm (Extra Credit)

Summary: The detected particle and its neighbors are oversampled using bilinear interpolation by a factor of 5.The accuracy and the precision of the Gaussian fitting algorithm is calculated in this oversampled space of the 3x3 neighborhood of the detected particle in the original image. In order to compute the accuracy a raw image is created with just the pixel intensities at their respective pixel co-ordinates without being convolved with PSF and no noise is added. This image serves as the ground truth. The predicted co-ordinates of all the detected particles are computed and the Euclidean distance between the predicted co-ordinates and the real co-ordinates are stored as the errors for each particle. The mean of these errors is the accuracy of the algorithm and the standard deviation of the errors is equal to its precision.

Code Execution Instructions: Run the `Extra Credit` section in the Main.m file.

- The output that is generated is the accuracy and precision values stored in the detectionError_mean and detectionError_std variables respectively.

Results: The results obtained are shown below

Synthetic Image: Noise Level	Accuracy (Mean of detection error)	Precision (Std dev of detection error)
10%	0.17	0.38
15%	0.44	0.54
20%	0.67	0.62
25%	0.98	0.60

References :

1) A. Ponti, P. Vallotton, W. C. Salmon, C. M. Waterman-Storer, and G. Danuser, Computational analysis of F-actin turnover in cortical actin meshworks using fluorescent speckle microscopy, *Biophysical Journal*, 84:3336-3352, 2003.

2) M. K. Cheezum, W. F. Walker, and W. H Guilford, Quantitative comparison of algorithms for tracking single fluororescent particles, *Biophysical Journal*, 81:2378-2388, 2001.