Video Article

Image Processing Protocol for the Analysis of the Diffusion and Cluster Size of Membrane Receptors by Fluorescence Microscopy

Carlos Oscar S. Sorzano*[1,2], Laura Martínez-Muñoz*[3,4], Graciela Cascio[3,5], Eva M. García-Cuesta[3], J. Vargas[1,6], Mario Mellado[3], Jose Miguel Rodriguez Frade[3]

[1]Department of Macromolecular Structures, Centro Nacional de Biotecnología, Campus Univ. Autónoma de Madrid

[2]Campus Urb. Montepríncipe s/n, Univ. San Pablo CEU

[3]Department of Immunology and Oncology, Centro Nacional de Biotecnología, Campus Univ. Autónoma de Madrid

[4]Department of Cell Signaling, Centro Andaluz de Biología Molecular y Medicina Regenerativa (CSIC)

[5]Meyer Cancer Center

[6]Department of Anatomy and Cell Biology, McGill Univ.

*These authors contributed equally

Correspondence to: Carlos Oscar S. Sorzano at coss@cnb.csic.es, Jose Miguel Rodriguez Frade at jmrfrade@cnb.csic.es

URL: https://www.jove.com/video/59314
DOI: doi:10.3791/59314

Keywords: Immunology and Infection, Issue 146, SPT, chemokine receptor, tracking, clusters, diffusion, analysis software

Date Published: 4/9/2019

Citation: Sorzano, C.O., Martínez-Muñoz, L., Cascio, G., García-Cuesta, E.M., Vargas, J., Mellado, M., Rodriguez Frade, J.M. Image Processing Protocol for the Analysis of the Diffusion and Cluster Size of Membrane Receptors by Fluorescence Microscopy. *J. Vis. Exp.* (146), e59314, doi:10.3791/59314 (2019).

Abstract

Particle tracking on a video sequence and the posterior analysis of their trajectories is nowadays a common operation in many biological studies. Using the analysis of cell membrane receptor clusters as a model, we present a detailed protocol for this image analysis task using Fiji (ImageJ) and Matlab routines to: 1) define regions of interest and design masks adapted to these regions; 2) track the particles in fluorescence microscopy videos; 3) analyze the diffusion and intensity characteristics of selected tracks. The quantitative analysis of the diffusion coefficients, types of motion, and cluster size obtained by fluorescence microscopy and image processing provides a valuable tool to objectively determine particle dynamics and the consequences of modifying environmental conditions. In this article we present detailed protocols for the analysis of these features. The method described here not only allows single-molecule tracking detection, but also automates the estimation of lateral diffusion parameters at the cell membrane, classifies the type of trajectory and allows complete analysis thus overcoming the difficulties in quantifying spot size over its entire trajectory at the cell membrane.

Video Link

The video component of this article can be found at https://www.jove.com/video/59314/

Introduction

Membrane proteins embedded in the lipid bilayer are in continuous movement due to thermal diffusion. Their dynamics are essential to regulate cell responses, as intermolecular interactions allow formation of complexes that vary in size from monomers to oligomers and influence the stability of signaling complexes. Elucidating the mechanisms controlling protein dynamics is thus a new challenge in cell biology, necessary to understand signal transduction pathways and to identify unanticipated cell functions.

Some optical methods have been developed to study these interactions in living cells[1]. Among these, total internal reflection fluorescence (TIRF) microscopy, developed in the early 1980s, allows the study of molecular interactions at or very near the cell membrane[2]. To study dynamic parameters of membrane protein trajectories obtained from TIRF data in living cells, a single particle tracking method (SPT)is required. Although several algorithms are available for this, we currently use those published by Jaqaman et al.[3] that address particle motion heterogeneity in a dense particle field by linking particles between consecutive frames to connect the resulting track segments into complete trajectories (temporary particle disappearance). The software captures the particle merging and splitting that result from aggregation and dissociation events[3]. One of the output data of this software is detection of the particles along the entire trajectory by defining their X and Y positions in each frame.

Once particles are detected, we apply different algorithms to determine the short timelag diffusion coefficient (D_{1-4})[4,5]. By applying the Moment Scaling Spectrum (MSS)[6,7,8] analysis or by fitting the 'alpha' value by adjustment of the Mean Square Displacement (MSD) to the curve[9], we also classify the particles according to the type of trajectory.

Analysis of spot intensity in fluorescence images is a shared objective for scientists in the field[10,11]. The most common algorithm used is the so-called Number and Brightness. This method nonetheless does not allow correct frame-by-frame intensity detection in particles in the mobile fraction. We have, thus, generated a new algorithm to evaluate these particle intensities frame-by-frame and to determine their aggregation state.

Once the coordinates of each particle are detected using U-Track2 software[3], we define its intensity in each frame over the complete trajectory, also taking into account the cell background in each frame. This software offers different possibilities to determine the spot intensity and the cell background and, using known monomeric and dimeric proteins as references, calculates the approximate number of proteins in the particle detected (cluster size).

In this article, we describe a careful guide to perform these three steps: 1) detecting and tracking single particles along a video of fluorescence microscopy using U-track; 2) analyzing the instantaneous diffusion coefficient (D_{1-4}) of those particles and the type of movement (confined, free, or directed) of particles with long trajectories by MSS; 3) measuring the spot intensity along the video corrected by the estimated background fluorescence for each spot. This allows cluster size estimation and identification of the photobleaching steps.

The use of this protocol does not require specialized skills and can be performed in any laboratory with cell culture, flow cytometry and microscopy facilities. The protocol uses ImageJ or Fiji (a distribution of ImageJ[12]), U-track[3], and some ad hoc made routines (http://i2pc.es/coss/Programs/protocolScripts.zip). U-track and ad hoc routines run over Matlab that can be installed in any compatible computer.

Protocol

1. Preparation of Biological Samples

1. Grow Jurkat cells in RPMI 1640 medium supplemented with 10% FCS, NaPyr and L-glutamine (complete RPMI). Electroporate Jurkat cells (20 x 10^6 cells/400 µL of RPMI 1640 with 10% FCS) with a monomeric GFP-labelled chemokine receptor vector (CXCR4-AcGFP, 20 µg) to allow its detection using fluorescence microscopy.
   NOTE: It is possible to use other monomeric fluorescent proteins such as mCherry, mScarlet, etc.
2. 24 h after transfection analyze cells in a flow cytometer to determine both cell viability and CXCR4-AcGFP expression.
3. Select cells expressing low CXCR4-AcGFP levels by cell sorting of GFP^{low} positive cells (**Figure 1**), as low expression of transfected receptor is required in TIRFM experiments in order to ensure single particle tracking for tracing individual trajectories[9].
4. Quantify the number of receptors in the cell surface.
   NOTE: As an example[13], ~8,500 - 22,000 AcGFP-labeled receptors/cell, correspond to ~2 - 4.5 particles/µm^2.
5. **Resuspend sorted cells in complete RPMI and incubate for at least 2 h at 37 °C, 5% CO_2. Centrifuge cells (300 *x g*, 5 min), and resuspended them in TIRF buffer (HBSS, 25 mM HEPES, 2% FCS, pH 7.3).**
   1. Plate on 35 mm glass-bottomed microwell dishes (2-3 x 10^5 cell/dish) coated with fibronectin (20 µg/mL, 1 h, 37 °C) in the presence or absence of appropriate ligand (i.e., CXCL12, 100 nM, 1 h, 37 °C). Incubate cells (20 min at 37 °C, 5% CO_2) prior to image acquisition.
6. Perform experiments using a TIRF microscope, equipped with an EM-CCD camera, a 100x oil-immersion objective (HCX PL APO 100x/1.46 NA) and a 488 nm diode laser. The microscope allows temperature control and incubation with CO_2. Locate and focus cells with the coarse and fine focus knobs, using bright field to minimize photobleaching effects. For fine focus adjustment in TIRF mode use a low laser intensity, insufficient for single-particle detection or to induce photobleaching effects (5% laser power, 28 µW).
7. Acquire movies (image sequences) of approximately 50 s minimizing the time interval between frames. Penetration of the evanescent field should be 70-90 nm of depth. Save the acquired movies for each experimental condition as ".lif" (video.lif).
   NOTE: Movies in the described example were acquired at 49% at laser power (2 mW) with an exposure time of 90 ms, and a time interval of 98 ms, for 49 s (500 frames). Penetration of the selected evanescent wave was 90 nm.

2. Selection of Images and Creation of Masks

1. For each experimental condition (video.lif), create a new folder (VideoName) that must contain different folders for every series. Each folder will contain a "videoSeq" folder for the video images and a "results" folder for the results of the analysis. Make sure that the file structure at this moment is the following:
   VideoName/video.lif
   VideoName/Series1/**videoSeq**
   VideoName/Series1/**results**
   NOTE: From the microscope different .lif files are obtained with several videos for every treatment condition (i.e. FN, FN+SDF). "video.lif" corresponds to the input.lif video file with all TIRF movies acquisitions (Series) performed at the microscope. "**videoSeq**" folder will contain the 500 frames of the movie that we are analyzing. The "**results**" folder will contain all files resulting from the analysis performed. Accurate nomenclature and localization of the different folders are essential for the correct function of the algorithms. Names in bold in the list above are fixed (i.e., they have to be called in this way because these are the names sought by the scripts). Names not in bold can change to reflect the experiment performed.
2. Open the TIRFM video (.lif file) with Fiji or ImageJ by dragging and dropping the file on the Fiji menu bar and click on **OK** to import the lif file using BioFormats (**Supplementary Figure 1**).
3. Select the series to process and click **OK** (**Supplemental Figure 2A**). To design a mask for the analysis of this video, import also a multichannel image with the different chromophores (in the example, Series 1 is the multichannel image and Series 2 the corresponding video). The video (and the multichannel image) should open as an ImageJ stack. In the example (**Supplemental Figure 2B**), the image is on the left and the video is on the right.
   NOTE: If creation of a mask for the video is not needed, go to Step 2.5.
4. **Create a mask. Create a single image with the channels useful for the design of the mask. In this case, the interesting channels are the red, green and gray ones.**
   1. Split the channels from the multichannel image (**Supplementary Figure 3A**): select **Image** in the bar menu and click **Color | Split channels**. The different channels will show as separate images (**Supplementary Figure 3B**).

2. Merge again the three channels in a single image (**Supplemental Figure 4A**): select **Image** in the bar menu and select **Color | Merge channels**. Select the appropriate channels and press **OK** (**Supplemental Figure 4B**). A new non-stacked image will be generated (**Supplemental Figure 4C**).
3. Synchronize the two windows by using the **Synchronize Windows** tool (**Supplemental Figure 5A**): select **Analyze** in the bar menu **| Tools | Sync Windows**. A new window with the synchronize image possibilities will be shown (**Supplemental Figure 5B**).
4. With the two windows synchronized (only the video if there is no multichannel image associated), the same region in both windows can be cropped. Draw the region of interest with the rectangular selection tool of ImageJ floating menu. Select **Image** in the bar menu and select **Crop** (**Supplemental Figure 6A**). The two cropped images will show individually (**Supplemental Figure 6B**).
5. Unsynchronize both windows (**Supplemental Figure 6B**) by pressing the **Unsynchronize All** button in the **Sync Windows** manager.

5. If a mask has not been created as in step 2.4, draw the region of interest with the rectangular selection tool of ImageJ floating menu.
6. Save the video as an **Image sequence** in the directory **videoSeq** under the corresponding video directory (**Supplemental Figure 7A**): select **File** in the bar menu and click **Save as | Image Sequence….** Rename the labels for the video sequence as video0000.tif, video0001.tif, …, video0499.tif (**Supplemental Figure 7B**): in the **Name** box, rename as **video** and click **OK**. The sequence must be alone in its directory to be successfully used by U-track.
   NOTE: If not designing a mask for the video, go to Step 2.8.
7. **Design a Mask. Select the multichannel image and open the Segmentation Editor plugin (Supplemental Figure 8A): select Plugins in the bar menu and select Segmentation | Segmentation editor. Add and rename the labels of the segmentation as necessary by right clicking on the labels of the Segmentation editor (Supplemental Figure 8B,C).**
   1. Choose the appropriate selection tool in ImageJ floating menu (here, use freehand), select a label (Green) and design first the outermost mask (**Supplemental Figure 9A**). Once designed, press the **+** button in Selection option of **Composite** window, and the selected mask will be displayed on the viewer (**Supplemental Figure 9A**). Repeat this step with next labels (Interior, in red) (**Supplemental Figure 9B**).
      NOTE: After designing the mask for the Green and Interior labels, the Exterior mask will occupy the rest of the image.
   2. Masks are coded in the image as regions 0, 1, 2, … according to the order of the labels in the RGB labels window. When all masks for the different labels are designed, save the mask with the same filename as the video, with the name **mask.tif** (**Supplemental Figure 9C**): select **File** in the bar menu and select **Save as | Tiff…**.
      NOTE: The selected masks will be employed in the calculation of the diffusion coefficients and classification of the trajectories (see step 4.2).

8. Check that the file structure at this moment is the following:
   VideoName/video.lif
   VideoName/Series1/ **mask.tif**
   VideoName/Series1/ **videoSeq/ video0000.tif**
   VideoName/Series1/ **videoSeq/ video0001.tif**
   ...
   VideoName/Series1/ **videoSeq/ video0499.tif**
   VideoName/Series1/ **results**
   NOTE: The mask.tif is an image with the mask as designed in Steps 2.4 and 2.7. The video*.tiff is the video as saved in Step 2.6. As above, the names in bold in the list above are fixed, i.e., they have to be called in this way because these are the names sought by the scripts. Names not in bold can change to reflect the experiment performed.

3. Tracking the Particles

1. Track all the particles seen in the selected videos using U-track.
2. Open Matlab and add U-track directory to the path by using **Set Path | Add** with Subfolders option in the menu. Save the path so that in future executions of Matlab U-track is in the path. This path setting needs to be done only once.
3. Change the working directory to the directory containing the Series to be analyzed. Invoke U-track by typing in the console (**Supplemental Figure 10**) **movieSelectorGUI** and press enter. The **Movie selection** window will be opened (**Supplemental Figure 11A**).
4. Press on the **New** movie button and the **Movie edition** window will appear (**Supplemental Figure 11B**).
5. Press on **Add channel** to choose the directory with the video (VideoName/Series1/video) and fill the movie information parameters. Set the output directory for the results of U-track to **Results** (videoName/Series1/results).
   NOTE: The movie information parameters can be obtained from the microscope and the acquisition conditions.
6. Press on the **Advanced channel settings** and fill the parameters related to the acquisition. Look the values of the parameters in the example (**Supplemental Figure 11C**).
7. Press **Save** on the **Advanced channel settings** window and **Save** on the **Movie edition** window. The program will ask for confirmation of writing the file called **movieData.mat** on the **results** directory. Confirm.
8. **After creating the movie, press on Continue in the movie selection window. U-track will ask about the type of object to be analyzed. Choose Single-particles (Supplemental Figure 12). The Control panel window will appear (Supplemental Figure 13A).**
   1. Select the first step **Step 1: Detection** and press on **Setting**. The **Setting Gaussian Mixture-Model Fitting** window will appear (**Supplemental Figure 13B**). In the example, "Alpha value for comparison with local background" is set to 0.001 and "Rolling-Window time averaging" to 3 (**Supplemental Figure 13B**).
   2. Press on **Apply** in the **Settings Gaussian Mixture-Model Fitting** window and **Run** in the **Control panel**. With the configuration in **Supplemental Figure 13**, only the **Detection** step runs. This step takes a few (2-5) minutes. Check the results by pressing the **Result** button of the Step 1 (Detection, **Supplemental Figure 14**).
      NOTE: As shown above, the movie shows red circles on the detected particles. If no red circle is shown, then this step has not worked correctly.

9. Perform the identification of tracks, that is, merging the particles detected in the previous step into tracks that span multiple frames. This is the **Step 2: Tracking** of U-track whose settings have to be defined as shown in **Supplemental Figure 15A-C**. The Step 2 cost function settings for **frame-to-frame linking** and **Gap closing, merging and splitting** in the example are shown in **Supplemental Figure 15B** and **C**, respectively.
10. After setting the parameters for Step 2, press **Run** in the **Control panel** and only Step 2 runs (**Supplemental Figure 16**).
11. Perform track analysis, Step 3. Define the settings as shown in **Supplemental Figure 17** (right panel). Then, press **Apply** in the panel of Setting-Motion Analysis, and **Run** in the Control Panel-U-Track. This step takes a few seconds.
12. Verify with the **Result** button of Step 3 that the process has correctly identified all the tracks. For doing so, click on **Show track number** of the **Movie options** window, and check frame by frame that each track has been correctly identified (**Supplemental Figure 18**). Manually annotate those particles that are not true particles.
    NOTE: If this manual selection is not done, a weaker automatic selection can be performed later when the diffusion coefficient is calculated (see Step 4).

4. Calculation of the Diffusion Coefficients and Classification of Trajectories

1. Be sure that all the scripts are invoked from the directory of the video being analyzed (in the example, VideoName/Serie1).
2. Read all the trajectories to compute the diffusion coefficients by issuing in the Matlab console the command: **trajectories=readTrajectories(0.1)**, where 0.1 is the time in seconds between two consecutive frames (time interval, shown in panel Movie Information, **Supplemental Figure 11B)**.
3. Exclude trajectories corresponding to incorrectly identified spots/trajectories. Give a list of the spots to exclude. For instance, to exclude spots 4, 5 and 28, type: trajectories=readTrajectories(0.1, [4, 5, 28]).
4. Calculate the instantaneous diffusion coefficients for each one of the tracks of this cell. In this case, calculate the diffusion coefficient for a time lag = 4, called $D_{1\text{-}4}$. For doing so, run in the Matlab console the command: **D=calculateDiffusion(trajectories, 113.88e-3, 0.0015, 'alpha')** where trajectories are the trajectories obtained in Step 3, 113.88e-3 is the pixel size of the acquired images in microns, 0.0015 is an upper bound for the diffusion coefficients of immobile particles measured in $\mu m^2/s$, and 'alpha' is the fitted model as explained below.
    NOTE: When using a faster camera and need more frames to calculate the diffusion parameter increase it, e.g. to 20, by **D=calculateDiffusion(trajectories, 113.88e-3, 0.0015, 'alpha', '', 20)**. The string parameter before 20, in the example above, '', is the suffix added to the output files. This suffix may be used to differentiate different analyses.
5. Fit the MSD with a different function by calling the calculateDiffusion function again with a different fitting mode ('confined', 'free', or 'directed'). In this example, 'confined': **D=calculateDiffusion(trajectories, 113.88e-3, 0.0015, 'confined')**.
6. Obtain the fitting results for the directed model, as shown in **Supplemental Figure 20**.
7. Decompose the trajectories into short and long trajectories. Use the command: **[shortTrajectories, longTrajectories]=separateTrajectoriesByLength(trajectories,50)** where 50 is the minimum length in frames of a trajectory to be considered long (in the example shown).
8. Study short trajectories using the same fitting procedure described in Step 4.3: **D=calculateDiffusion(shortTrajectories, 113.88e-3, 0.0015, 'directed', 'Short')**. Analyze short and anomalous trajectories with the command: **D=calculateDiffusion(shortTrajectories, 113.88e-3, 0.0015, 'alpha', 'Short')**.
9. Analyze long trajectories to classify the type of motion through their Moment Scaling Spectrum (MSS)[7]. The command: **trajectoriesClassification=classifyLongTrajectories(longTrajectories,113.88e-3,0.0015,'Long'**) shows the analysis in screen and generates a file called **trajectoryClassification<Suffix>.txt** in the directory **results\TrackingPackage\tracks**.

5. Calculation of Cluster Ssize Through the Particle Density

NOTE: Be sure that all the scripts are invoked from the directory of the video being analyzed (in the example shown, VideoName/Serie1).

1. **Analyze the intensity of each particle along their trajectory. For doing so, invoke the script by typing in the Matlab console: analyzeSpotIntensities that takes as input the trajectories calculated by U-track in the first section. In its most basic form, simply call the script without any argument from the directory of the video being analyzed (in the example shown, VideoName/Series1) analyzeSpotIntensities(). Configure this basic behavior in many different ways by providing arguments to the script as in: analyzeSpotIntensities(`Arg1´, Value1, `Arg2´, Value2, …). Valid arguments with their corresponding variable values (`ArgN´, ValueN) are listed.**
    1. ('spotRadius´, 1)
       Analyze the fluorescence intensity using the spotRadius of 1 pixel (by default) that corresponds to a patch of size 3x3 centered at the spot ((2*spotRadius+1)x(2*spotRadius+1)).
       NOTE: For a patch of 5x5 centered at the spot, choose a spotRadius of 2, etc.
    2. ('onlyInitialTrajectories´, 1)
       If this argument is given (true, value of 1), analyze only the trajectories that start in the first frame of the video. This is useful to analyze control images (by default 0, false).
    3. ('trackTrajectory´, 0)
       If this argument is set to 0 (false), then keep the coordinate of the spot in its first frame for all frames (this is useful for immobile spots). If the argument is set to 1 (by default 1, true), then the spot is tracked along the video following the coordinates calculated by U-track.
    4. ('excludeTrajectories´, [4,5,28])
       Include the trajectory number of those trajectories excluded in Step 4.3 (in the example 4, 5, 28).
    5. ('extendTrajectory ´, 1)
       If this argument is set to 1 (true), then analyze the intensity in the patch to the end of the video (even if the trajectory stops earlier). The coordinate of the spot is either the last coordinate in the trajectory (if trackTrajectory is true) or the first coordinate in the trajectory (if trackTrajectory is false). This argument is false (0) by default.
    6. ('subtractBackground´, 1)

If this parameter is set, then correct the raw fluorescence measured at each spot by the estimate of the background fluorescence for that spot (see below). This argument is true (1), by default.

7. ('meanLength´, frame number)
   If this parameter is set, then the mean intensity spot is measured at the length indicated. Set `meanLength', 20 to measure the mean spot intensity at the first 20 frames. If the argument is not set, then the spot intensity is calculated at the whole trajectory (by default, full length).
8. ('showIntensityProfiles´, 1)
   Set this parameter as 1 (by default 0, false), to plot the intensity profile along the different frames as well as their background.
   NOTE: These plots are very useful to identify photobleaching as shown in the **Supplemental Figure 21**. For every path, the routine automatically analyzes if it is possible that there has been photobleaching. This is done by comparing the intensity values with a Student's t in the first and last N frames along the path. By default, N is 10, but this value can be modified through the argument `Nbleach´.
9. ('backgroundMethod´, value)
   Set this parameter to determine the background of each spot. This can be done in several ways, and which one to use can be selected changing the "value":
   1. ('backgroundMethod´, 0)
      Use this value to manually identify the background for the whole video. Allow to select 8 points in the first frame of the video. A patch around these points is analyzed along the whole video, and the 95% quantile of all these intensities is chosen as the background intensity for all spots.
   2. ('backgroundMethod´, 1)
      Use this value to manually identify the background for each frame. Choose 8 points for every spot and every frame. This is a time consuming task but it gives a lot of control to the user. The 95% quantile of the intensities in these patches is chosen as the background intensity for this spot in this frame.
   3. ('backgroundMethod´, 2)
      Use this value to calculate the background of each spot estimated from 8 points located in a circle around the spot with a radius controlled by the argument `backgroundRadius´ (by default, 4*spotRadius).
   4. ('backgroundMethod´, 3)
      Use this value to calculate the background for each frame by first locating the cell in the video and then analyzing the intensities of the cell in each frame (**Supplemental Figure 22**).
      NOTE: The background is chosen as the gray value at a given quantile of this distribution (by default 0.5 (=50%), although this parameter can be controlled through the argument `backgroundPercentile´, this value can be set higher, for instance, 0.9 (=90%) if wanting most of the cell to be considered as background. To help in the identification of the cell, indicate which is the maximum background value expected along the frames using the argument `maxBackground´ (for instance, in all the analyzed videos, the background value normally never goes beyond 6000)[13]. By default, this option is set to 0, meaning that this help is not used by default. See which is the cell detection and the area selected for the background estimation by setting the argument `showImages´ to 1 (stop the execution at any time by pressing CTRL-C).

2. Gather the diffusion and intensity information for all the trajectories calculated in the Steps 4 and 5.1, respectively, using gatherDiffusion.AndIntensity (). Gather only the diffusion and intensity information for short trajectories. For doing so, use the suffixes used in Step 4.7, and type: **gatherDiffusionAndIntensity (`Short´)** where `Short´ is the suffix used in Step 4.7.
3. Gather the Moment Spectrum Scaling and the intensity informationby typing: **gatherTrajectoryClassificationAndIntensity('Long')** where `Long' is the suffix used in Step 4.7. A summary of all the files generated using this protocol is shown in **Figure 2**.

Representative Results

The use of this protocol allows the automated tracking of particles detected in fluorescence microscopy movies and the analysis of their dynamic characteristics. Initially, cells are transfected with the fluorescently-coupled protein to be tracked. The appropriate level of receptors presents on the cell surface that allows SPT is obtained by cell sorting (**Figure 1**). Selected cells are analyzed by TIRF microscopy that generates videos in a format that can be studied with the tools described in this protocol (**Supplemental Video 1**).

The videos generated cannot be directly analyzed, and independent frames of each video are required. The use of ImageJ generates files that can be interpreted using Matlab software such as video frames in .tiff format or masks. U-track tracks the particles seen in the selected video and saves the information in Matlab (.mat) files (*movieData.mat*, *Channel_1_detection_result.mat, channel_1.mat, Channel_1_tracking_result.mat*), see **Figure 2**.

As shown in **Figure 2**, the calculation of the diffusion coefficients and classification of trajectories commands, generates different files (*diffusionCoefficients.txt*, *diffusionCoefficientsMobile.txt*, *diffusionCoefficientsShort.txt, trajectoryClassificationLong.txt*, etc). The information contained in these files are best managed using Excel and Prism software. Information that can be obtained from this analysis includes:

1. Percentage of immobile spots. You can use as reference of immobile particles: the 95% percentile of D_{1-4} values obtained (1) from single particles in fixed cells and/or (2) from purified fluorescent protein attached to the coverslips (**Figure 3A**).
2. Percentage of long trajectories (>50 frames) (**Figure 3B**).
3. Type of movement of the long trajectories: directed, free, confined (**Figure 3C**).
4. Diffusion coefficient (D_{1-4}) of mobile particles in cells treated with different stimulus (**Figure 4A**).

The step 5 of the protocol analyzes the intensities of each particle along the trajectory. The script **analyzeSpotIntensities** will print on screen all the information analyzed. For each spot and frame, the script shows the coordinate of the spot in that frame (x,y) in pixel units, the estimate of the background fluorescence (k0), the raw spot intensity in the 3x3 patch, the corrected intensity (calculated as the raw intensity minus its background), the maximum value of intensity observed in the patch, and the region number within the mask where this spot is located at this frame. An example of the kind of output produced is

spot=43 frame=184
x=78.0397
y=72.5395
k0=1571
spotRawIntensity=5550.1111
spotCorrectedIntensity=3979.1111
maxCorrectedSpotIntensity=6243
maskRegion=2

All this information is stored in a log file (*results/TrackingPackage/tracks/log.txt*) and a table that can be read from Excel (*results/TrackingPackage/tracks/spotIntensitiesByFrame.txt*). After analyzing each spot, the script prints the average intensity along the trajectory and the majoritarian region within the mask

spot=43
meanCorrectedSpotIntensity along frames=4762.303
majoritarian region=3

This information is stored in the log file above and a table that can be read from a spreadsheet (*results/TrackingPackage/tracks/meanSpotIntensities.txt*). As an example, mean spot intensities (msi) for every particle along their trajectory in cells stimulated with different conditions is shown in **Figure 4B**. We may now use this information to roughly estimate the size of the fluorescent cluster. As a spot's mean corrected intensity is related with the number of fluorescent proteins present in this spot, and the fluorescence of a monomer can be measured through a similar but independent experiment, directly calculate the number of receptors per particle. The frequency distribution of receptor number per particle using as reference the intensity of the monomeric protein expressed in the same cells is shown in **Figure 4C**.

The gather diffusion and intensity command create two files: one called *diffusionCoefficientsAndIntensitiesShort.txt*and another called *log_diffusionCoefficientsAndIntensitiesShort.txt* in the directory *results/TrackingPackage/tracks*. Both files contain 1) the spot index, 2) the diffusion coefficient, 3) the intensity, and 4) the region number within the mask. These files can be read from a spreadsheet.

Similarly, the gather trajectory classification and intensity command will create two files one called *trajectoryClassificationAndIntensitiesLong.txt* and another *log_trajectoryClassificationAndIntensitiesLong.txt* in the directory *results/TrackingPackage/tracks*. The first one contains 1) the spot index, 2) the movement type, 3) the spot first moment, 4) the intensity, 5) $D_{1\text{-}4}$ and 6) the region number within the mask. This file can be read from Excel.

A summary of all the files generated using this protocol is shown in **Figure 2**. Other analysis that can be performed using this protocol includes the comparison of the dynamic parameters of small vs bigger spots, i.e. monomers vs oligomers, variations on these dynamic parameters induced by ligands, inhibitors, membrane composition, alteration of signaling pathways, etc. A complete analysis of CXCR4 behavior in response to its ligand CXCL12 under different experimental conditions has been performed using this protocol[13].

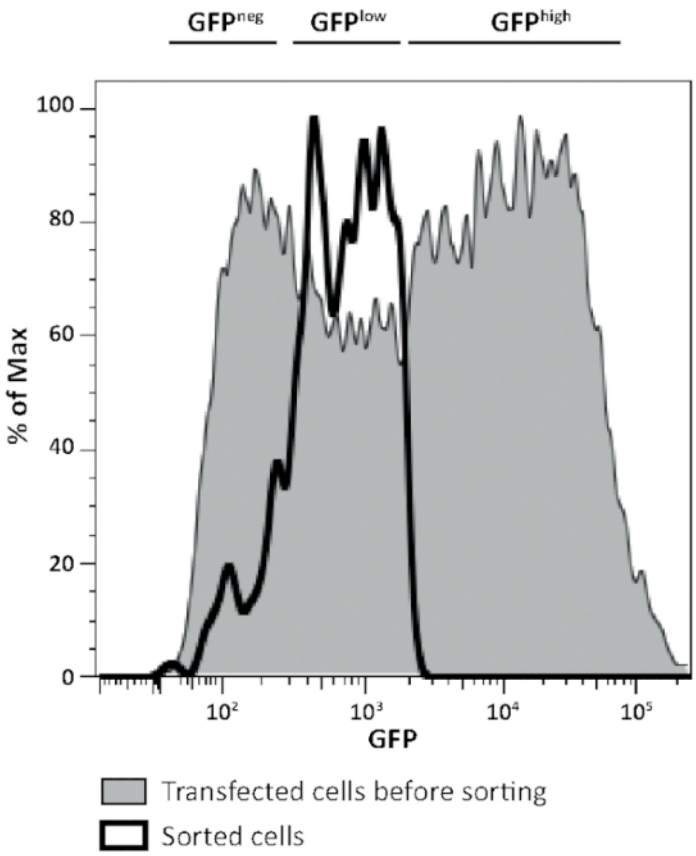

Figure 1: Cell sorting. Expression of GFP in Jurkat cells transfected with CXCR4-AcGFP before and after cell sorting. Cell expressing low levels of GFP (GFP^{low}) are selected and employed for TIRF experiments.

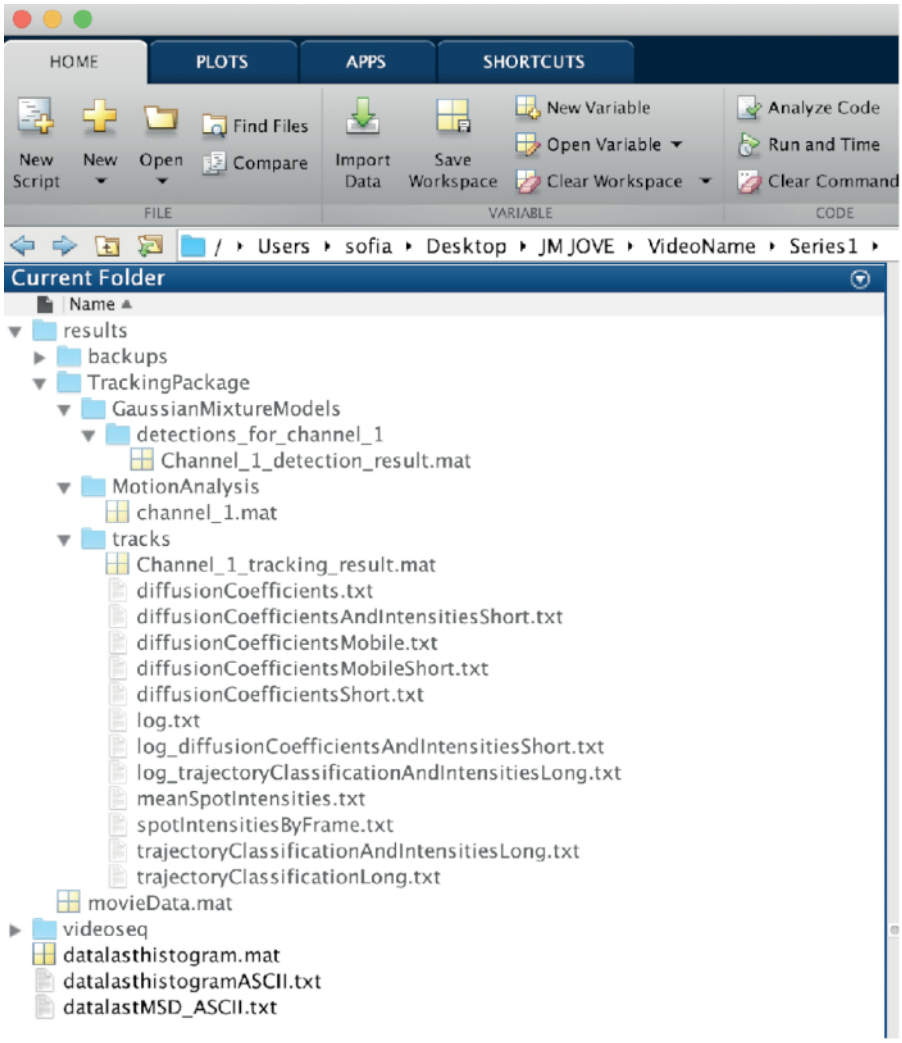

Figure 2: Summary of the files generated. The figure shows all the files generated using the Matlab routine described. Please click here to view a larger version of this figure.

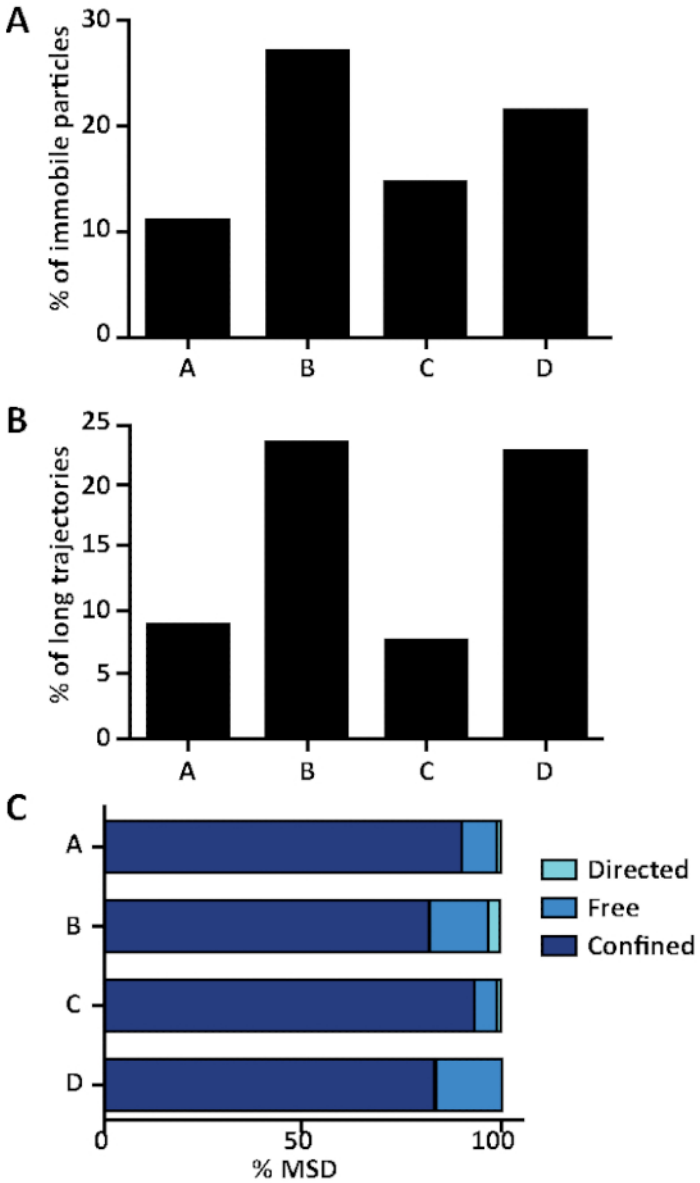

Figure 3: Classification of trajectories. Number of the different types of trajectories corresponding to cells treated with different stimuli. (**A**) Percentage of immobile spots, (**B**) percentage of long trajectories and (**C**) type of movement of the long trajectories.

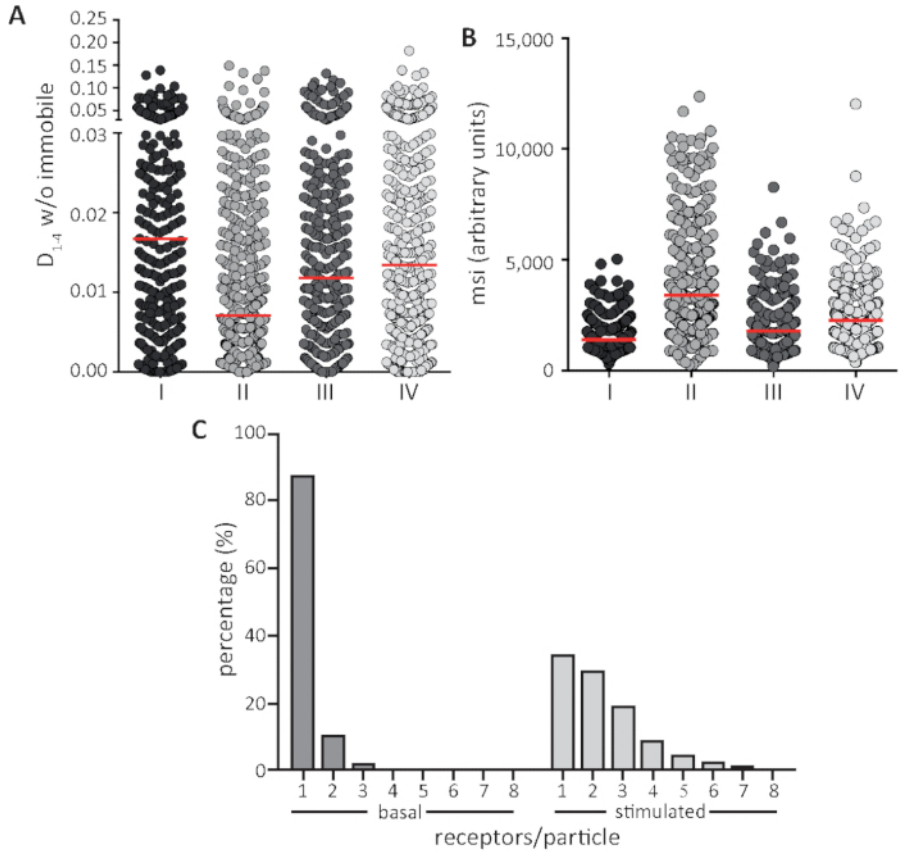

Figure 4: Diffusion coefficient, mean spot intensities and number of receptors per particle. (**A**) Distribution of the short diffusion coefficients (D_{1-4}) values for each spot in response to different stimuli (I, II, II and IV). Red line represents the median value for D_{1-4}. (**B**) Mean spot intensities (msi) values for each spot along its first 20 frames in response to different stimuli (I, II, II and IV). Red line represents the mean intensity value (SD). (**C**) Percentage of receptors/particle as extracted from intensity distribution of the individual particles, taking as bin width the monomeric protein intensity value. Please click here to view a larger version of this figure.

Supplemental Figure 1: Opening a TIRFM file in Fiji or ImageJ. Options that appear on the Bio-formats window upon dragging and dropping a .lif video. Please click here to download the figure.

Supplemental Figure 2: Select series. Images present in the .lif video. (**A**) Window that permits selection of the series to be analyze, including multichannel images. (**B**) Result example of series selection, including multichannel image (left) and corresponding video (right). Please click here to download the figure.

Supplemental Figure 3: Split channels. (**A**) Window capture of the ImageJ commands needed for splitting channels of a multichannel image and (**B**) result example of the channel split. Please click here to download the figure.

Supplemental Figure 4: Merge Channels. (**A**) Merging different channels in a single image. (**B**) Channel selection for the merging. (**C**) Result example of the channel merge. Please click here to download the figure.

Supplemental Figure 5: Synchronize windows. (**A**) Localization in the ImageJ menu of the commands required for image synchronization and (**B**) resulting window. Please click here to download the figure.

Supplemental Figure 6: Select the region of interest. (**A**) Selection of the window of interest using the rectangular selection tool and (**B**) result of the image crop. Please click here to download the figure.

Supplemental Figure 7: Save the video. Save the region of interest as an image sequence and parameters for the Save as Image sequence window. Please click here to download the figure.

Supplemental Figure 8: Segmentation. (**A**) Open the Segmentation editor plugin in the ImageJ's plugins menu. (**B**) Add labels/materials for the segmentation. Please click here to download the figure.

Supplemental Figure 9: Mask design. (**A**) Selection of the appropriate labels and definition of the green mask. (**B**) Selection of the red mask. Example of two masks corresponding to two labels. (**C**) Save the masks as a .tiff file. Please click here to download the figure.

Supplemental Figure 10: Matlab working directory. Selection of the correct directory containing the series to be analyzed. Please click here to download the figure.

Supplemental Figure 11: U-track main menu. (**A**) Movie selection, (**B**) movie edition and (**C**) channel Settings menus. Please click here to download the figure.

Supplemental Figure 12: Select the type of object to track. Select tracking of single particles. Please click here to download the figure.

Supplemental Figure 13: Detection of particles. (**A**) U-track, control panel. (**B**) Example of settings for the detection of particles. Please click here to download the figure.

Supplemental Figure 14: Example of results for particle detection. Different windows that include a viewer menu, movie options menu and the movie. Please click here to download the figure.

Supplemental Figure 15: Tracking menu. Example of settings. (**A**) Tracking, (**B**) setting - frame to frame linking and (**C**) gap closing, merging and splitting menus. Please click here to download the figure.

Supplemental Figure 16: Example of results for particle tracking. Please click here to download the figure.

Supplemental Figure 17: Track analysis menu. Example of settings. Please click here to download the figure.

Supplemental Figure 18: Verification of track analysis. Screen displayed after track analysis, including a video window showing the particles detected and their corresponding tracks. Please click here to download the figure.

Supplemental Figure 19: Calculation of diffusion coefficients. Histograms of the diffusion coefficients calculated (left) and the mean squared displacement (MSD, right). Please click here to download the figure.

Supplemental Figure 20: Calculation of diffusion coefficients including the different fitting modes. Histograms of the diffusion coefficients calculated (left) and the mean squared displacement (MSD, right) for the confined model. Please click here to download the figure.

Supplemental Figure 21: Intensity Profiles. Example of spot intensity along its trajectory (blue line) and background (red line). Please click here to download the figure.

Supplemental Figure 22: Background for each frame. Left: Sample cell image. Middle: Automatically detected cell. Right: Area automatically selected as background. Please click here to download the figure.

Supplemental Video 1: Example of a typical TIRF video microscopy showing the presence of particles with different intensities and types of movements. Please click here to download the video.

Supplemental Material 1: Files containing all the protocol scripts employed in the *ad hoc* routines employed for the classification of trajectories and analysis of the cluster size. Please click here to download the materials.

Discussion

The described method is easy to perform even without having any previous experience working with Matlab. However, Matlab routines require extremely accuracy with the nomenclature of the different commands and the localization of the different folders employed by the program. In the tracking analysis routine (step 3), multiple parameters can be modified. The "Setting Gaussian-Mixture Model Fitting" window (step 3.8) controls how U-track will detect single particles on the video. This is done by fitting a Gaussian mixture model as described in[3]. One of the key parameters for this fitting defines a filter to help identifying local maxima. The success of this operation depends on the image contrast and the noise present in the images. The first parameter (Alpha value for comparison with local background) controls the confidence of a maxima being a real spot, while the second helps to reduce noise during the identification of spots. Important parameters in the "Tracking" step (3.9) are the number of frames to close gaps (that is a track may span over frames in which the particle is not actually seen, but it is seen before these frames and after these frames; in the example, 0), and the number of frames that a track must span in order to be considered as a successful track (in our example, 20; this parameter is related to the camera acquisition speed and the number frames in the track must be such that it allows the calculation of the diffusion coefficient). It is also important to decide whether to merge and split segments (in the example these two possibilities were chosen). Then, the parameters for the Step 1 in Cost functions (frame-to-frame linking) must be set. This function controls how the particles are tracked along the frames. The most important parameters at this point is the selection of the Brownian search radius, which control how far each spot is expected to be in the next frame. In our example we chose 0 and 5 as lower and upper bounds, respectively. Note that these parameters are very specific to the nature of the particles being tracked, and that they may have to be tuned in each specific case. Particularly important are the scaling power in the Brownian and Linear search radii. These scaling powers depend on the kind of movement of the particles (free or confined diffusion). In the example, the values (0.5, 0.01) were chosen for free diffusion. For the specific documentation of these parameters, see[3].

In the calculation of the diffusion coefficients command (step 4.4), the upper bound of the diffusion coefficient of immobile particles can be known from previous experiments or by running the calculateDiffusion function on a separate project with immobile particles (purified monomeric fluorescent protein) and seeing the diffusion coefficients reported. This function takes two extra parameters: `outputSuffix´, which is added to the output filenames and by default takes the empty value, and `plotLength´, which by default is 13 and is the number of time lag (seconds) in which the diffusion calculation is performed. The fitting mode 'alpha' implies an adjustment of the MSD to the curve

$$MSD = MSD_0 + 4Dt^{\alpha}$$

that is, an offset (MSD_0) and a power function of the time lag. The exponent of this power function, α, determines whether the movement is confined ($0 < \alpha < 0.6$), anomalous ($0.6 < \alpha < 0.9$), free ($0.9 < \alpha < 1.1$), or directed ($\alpha > 1.1$). For a review of this kind of analysis the reader is referred to Manzo et al.[9]

The calculation of the diffusion coefficients[4] produces two output figures (see **Supplemental Figure 19**). The first one shows a histogram of the diffusion coefficients calculated (note that at this point, there is an independent diffusion coefficient for each trajectory). The mean and 95% percentile of these coefficients are shown in the Matlab console. The second plot shows the MSD (red curve) and the number of steps considered to calculate it as a function of the time lag for those trajectories considered to be mobile (those whose diffusion coefficient is above the threshold given in the command line). The average and standard deviation of the mobile trajectories is computed (these are the values for the trajectories in a single cell). In the example, the exponent is 0.59 meaning that the movement is confined. For this curve fitting, the program reports the uncertainty associated to each one of the parameters (shown as the standard deviation of each one) and the goodness of fit (a perfect fit would reach zero). At this point and after checking the value of alpha we may decide to fit the MSD with a different function:

$$MSD = MSD_0 + <r^2>\left(1 - Ae^{-\frac{4BDt}{<r^2>}}\right)$$ if $0 < \alpha < 0.6$ (confined)

$$MSD = MSD_0 + 4Dt$$ if $0.9 < \alpha < 1.1$ (free)

$$MSD = MSD_0 + 4Dt + (vt)^2$$ if $\alpha > 1.1$ (directed)

Anomalous trajectories cannot be fitted with a different function. In case of confined particles you may calculate the confinement size (in microns) as in Destainville et al.[14]

$$L = \sqrt{3(<r^2> + MSD_0)}$$

A good indicator of a correct fitting is that the goodness-of-fit should decrease from the alpha fitting to the final fitting (in the example, the goodness fit falls from 0.30474 to 0.15749, **Supplemental Figure 19-20**). calculateDiffusioncreates two files in the "results\TrackingPackage \tracks" inside the series directory called "*diffusionCoefficients.txt*" and "*diffusionCoefficientsMobile.txt*". These files contain three columns: 1) the index of each input trajectory, 2) their corresponding diffusion coefficient, 3) the majoritarian region in the mask where this track belongs to (the regions are obtained from the file "mask.tif" that we generated in Steps 2.7; if this file does not exist, then this column is not present). You may use this file and the analogous files generated for other cells to analyze the distribution of the diffusion coefficient for a set of cells under similar experimental conditions.

In the calculation of the diffusion coefficient for short trajectories (steps 4.7-4.8), the last parameter 'Short' is a suffix added to the output filename so that you may analyze different subsets of trajectories without overwriting the *diffusionCoefficients.txt* files. The actual name of the output files is "diffusionCoefficients<Suffix>.txt" and "diffusionCoefficientsMobile<Suffix>.txt". If no suffix is given, as in the general analysis performed above, then an empty suffix is assumed. The model parameters (D, v, and MSD_0) may differ from those fitted to all trajectories. Most remarkably, the standard deviation of the parameters as well as the Goodness fit normally increase. The reason is that short trajectories are more unstable and a reliable fitting is more difficult.

The analysis of long trajectories (step 4.9) classify them into confined (1), free (2), or directed (3) according to their first moment and its location with respect to the 2.5% and 97.5% percentiles of the first moment of 500 random paths with Brownian motion, the same diffusion coefficient and length as the trajectory being analyzed and simulated by Monte Carlo. If the first moment of the path being analyzed is below the 2.5% of the first moments observed in the simulations, the path being studied is classified as confined. If it is above the 97.5% of the simulated first moments, it is classified as directed; otherwise, the path is classified as Brownian. In the command employed, 113.88e-3 is the pixel size in μm, 0.0015 is the upper bound of the diffusion coefficient of immobile spots measured in $\mu m^2/s$, and 'Long' is the suffix for the output filename. The first column of this file ("*trajectoryClassificationLong.txt*") is the trajectory number, the second is its classification (1=confined, 2=free, 3=directed), and the third is the trajectory first moment.

The script for calculating the intensity of each particle (step 5.1) is highly flexible and allows tracking the fluorescence intensities in many different ways (each one well suited for different situations like tracking immobile spots of a control experiment or tracking highly movable spots over a cell with variable fluorescent background). The script will analyze all trajectories along all frames. For each spot at each frame it will measure the intensity of the pixels around the spot (it analyzes a square patch around the spot, by default of a size 3x3 pixels), estimate the spot background and calculate the fluorescence difference between the spot and the background. The percentage of photobleaching particles and the number of fluorescent particles in a single cluster, seen as a single spot, can be estimated in this way.

This automated method (gatherTrajectoryClassificationAndIntensity) produces information on multiple parameters (spot intensity, lateral diffusion, type of motion), that can help to study the relation between spot size and its dynamic at basal conditions, and how different treatments can modify these parameters.

The main limitation of the method is that it requires cell transfection with the fluorescent protein to be tracked. Usually transfection leads to protein overexpression, a fact that hinders single protein tracking. A cell sorting step must be included to select cells expressing a number of receptors that allow detection and tracking of single particles by SPT-TIRF microscopy. This method could also be employed for tracking biomolecules labelled with quantum dots or Fab fragments. The described protocol requires a number of controls that must be previously analyzed in order to ensure that the conclusions driven from the analysis are correct. First of all, it is critical to determine the appropriate

expression conditions that allow detection and tracking of single particles. Movies with densities of ~ 4.5 particles/μm^2, corresponding to 8,500-22,000 receptors/cell, were used to analyze the spatio-temporal organization of the cell membrane receptor[13].

It is important to establish the minimum detectable diffusion coefficient by using purified monomeric AcGFP proteins or fixed cells. In both cases it is assumed that there is no diffusion and therefore we estimate that the diffusion values of particles analyzed in these conditions correspond to immobilized particles and used to discriminate between mobile and immobile trajectories[13].

The analysis we have presented here is a general trajectory analysis tool, that can be applied to the analysis of diffusion of receptors by superresolution, to the analysis of diffraction limited images and to the analysis of cellular movement by standard microscopy. The main advantage of this method with others previously described, such as analysis of number and brightness[11], is that it allows the evaluation of the mean intensity values of each individual spot along its trajectory, taking into account the time of survival of the AcGFP monomeric protein before photobleaching. The survival time of a molecule before photobleaching will strongly depend on the excitation conditions.

Disclosures

The authors have nothing to disclose.

Acknowledgments

We are thankful to Carlo Manzo and Maria García Parajo for their help and source code of the diffusion coefficient analysis. This work was supported in part by grants from the Spanish Ministry of Science, Innovation and Universities (SAF 2017-82940-R) and the RETICS Program of the Instituto de salud Carlos III (RD12/0009/009 and RD16/0012/0006; RIER). LMM and JV are supported by the COMFUTURO program of the Fundación General CSIC.

References

1. Yu, J. Single-molecule studies in live cells. *Annual Review of Pysical Chemistry.* **67** (565-585) (2016).
2. Mattheyses, A. L., Simon, S. M., Rappoport, J. Z. Imaging with total internal reflection fluorescence microscopy for the cell biologist. *Journal of Cell Science.* **123** (Pt 21), 3621-3628 (2010).
3. Jaqaman, K. et al. Robust single-particle tracking in live-cell time-lapse sequences. *Nature Methods.* **5** (8), 695-702 (2008).
4. Bakker, G. J. et al. Lateral mobility of individual integrin nanoclusters orchestrates the onset for leukocyte adhesion. *Proceedings of the National Academy of Sciences U S A.* **109** (13), 4869-4874 (2012).
5. Kusumi, A., Sako, Y., Yamamoto, M. Confined lateral diffusion of membrane receptors as studied by single particle tracking (nanovid microscopy). Effects of calcium-induced differentiation in cultured epithelial cells. *Biophysical Journal.* **65** (5), 2021-2040 (1993).
6. Ferrari, R. M., Manfroi, A.J., Young, W. R. Strongly and weakly self-similar diffusion. *Physica D.* **154** 111-137 (2001).
7. Sbalzarini, I. F., Koumoutsakos, P. Feature point tracking and trajectory analysis for video imaging in cell biology. *Journal of Structural Biology.* **151** (2), 182-195 (2005).
8. Ewers, H. et al. Single-particle tracking of murine polyoma virus-like particles on live cells and artificial membranes. *Proceedings of the National Academy of Sciences U S A.* **102** (42), 15110-15115 (2005).
9. Manzo, C., Garcia-Parajo, M. F. A review of progress in single particle tracking: from methods to biophysical insights. *Report on Progress in Physics.* **78** (12), 124601 (2015).
10. Calebiro, D. et al. Single-molecule analysis of fluorescently labeled G-protein-coupled receptors reveals complexes with distinct dynamics and organization. *Proceedings of the National Academy of Sciences U S A.* **110** (2), 743-748 (2013).
11. Digman, M. A., Dalal, R., Horwitz, A. F., Gratton, E. Mapping the number of molecules and brightness in the laser scanning microscope. *Biophysical Journal.* **94** (6), 2320-2332 (2008).
12. Schindelin, J. et al. Fiji: an open-source platform for biological-image analysis. *Nature Methods.* **9** (7), 676-682 (2012).
13. Martinez-Munoz, L. et al. Separating Actin-Dependent Chemokine Receptor Nanoclustering from Dimerization Indicates a Role for Clustering in CXCR4 Signaling and Function. *Molecular Cell.* **70** (1), 106-119 e110 (2018).
14. Destainville, N., Salome, L. Quantification and correction of systematic errors due to detector time-averaging in single-molecule tracking experiments. *Biophysical Journal.* **90** (2), L17-19 (2006).

Image Processing Protocol for the Analysis of the diffusion of isolated particles by fluorescence microscopy

C.O.S. Sorzano[1,2,*,†], L. Martínez-Muñoz[1,*], G. Cascio[1], J. Vargas[1], J.M. Rodríguez-Frade[1], M. Mellado[1]

[1] Centro Nacional de Biotecnología, CSIC. c/Darwin, 3, Campus Univ. Autónoma de Madrid, 28049 Cantoblanco, Madrid, Spain
[2] Univ. San Pablo CEU, Campus Urb. Montepríncipe s/n, 28668 Boadilla del Monte, Madrid, Spain

[*] These authors have contributed equally
[†] Corresponding author: coss@cnb.csic.es

Abstract

Particle tracking on a video sequence and the posterior analysis of their trajectories is nowadays a common operation in many biological studies. In this article, we present a detailed protocol for this image analysis task using Fiji (ImageJ) and Matlab routines to: 1) define regions of interest and designing masks adapted to these regions; 2) tracking the particles in fluorescence microscopy videos; 3) analyzing the diffusion characteristics of selected tracks. We illustrate our protocol with the analysis of cell membrane receptor clusters. However, the image analysis protocol is not restricted to this kind of images. Some routines for the trajectory analysis have specifically been for this protocol and they are freely available at http://i2pc.es/coss/Programs/protocolScripts.zip.

Introduction

The cell membrane is a heterogeneous lipid bilayer with numerous embedded proteins. These molecules show thermal diffusion motion given the membrane dynamic properties with many intermolecular interactions that form complexes that vary in size from dimers to oligomers, and in stability, from microseconds to hours. Elucidating the mechanisms involved in nanocluster formation and dynamics is a new challenge in cell biology, necessary to understand signal transduction pathways and to identify unanticipated cell functions.

Some optical methods have been developed to study these interactions in living cells [Yu2016]. Of these, total internal reflection fluorescence microscopy, developed in the early 1980s, allows the study of molecular interactions at or very near the cell plasma membrane [Mattheyses2010]. With this technique, analysis of membrane protein trajectories is essential for determining their dynamism; a single-particle tracking (SPT) method is therefore needed to detect and study the motion of individual particles in live cells. Although several algorithms are available for this, we currently use those published by Jaqaman et al. [Jaqaman2008], who address particle motion heterogeneity in a dense particle field by linking particles between consecutive frames to connect the resulting track segments into complete trajectories (temporary particle disappearance). The software captures the particle

merging and splitting that result from aggregation and dissociation events [Jaqaman2008]. One of the output data of this software is detection of the particles along the entire trajectory by defining their X and Y positions in each frame.

Once particles are detected, we apply some algorithms to determine the diffusion coefficient (D_{1-4}) [Kusumi1993, Bakker2012]; by applying the moment scaling spectrum (MSS) [Ferrari2001, Sbalzarini2005] and cumulative probability distribution (CPD) [Schuetz1997, Bakker2012], we also classify the particles by the type of trajectory used.

Analysis of spot intensity in fluorescence images is a shared objective for scientists in the field [Digman2008, Calebiro2013]. The most common algorithm used is the so-called Number and Brightness. This method nonetheless does not allow correct frame-by-frame intensity detection in particles in the mobile fraction. We have thus generated a new algorithm to evaluate these particle intensities frame-by-frame and to determine their aggregation state as well as their interactions and dissociations through time. Once the coordinates of each particle are detected using U-Track2 software, we define its intensity in each frame over the complete trajectory, also taking into account the cell background in each frame. Our MatLab software offers different possibilities to determine the cell background and, using known monomeric and dimeric proteins as references, calculates the approximate number of proteins in the particle detected (cluster size).

In this article we describe a careful guide to perform these three steps:

1) Tracking single particles along a video of fluorescence microscopy using U-track.
2) Analyzing the diffusion coefficient of those particles and the type of movement (confined, free, or directed) of particles with long trajectories.
3) Measuring the spot intensity along the video comparing it to the intensity of the spot background. This allows estimating the cluster size and identifying photobleaching.

The protocol uses Fiji (a distribution of ImageJ [Schindelin2012], https://imagej.net/Fiji), U-track [Jaqaman2008] (http://www.openmicroscopy.org/site/products/partner/u-track), and some *ad hoc* made routines (http://i2pc.es/coss/Programs/protocolScripts.zip). U-track and our routines run over Matlab (remind to include the directories and subdirectories of U-track and our routines in Matlab's path).

Tracking of single particles

1) Open the TIRFM video (.lif file) with Fiji by drag and dropping the file on the Fiji menu bar (Fig. 1).

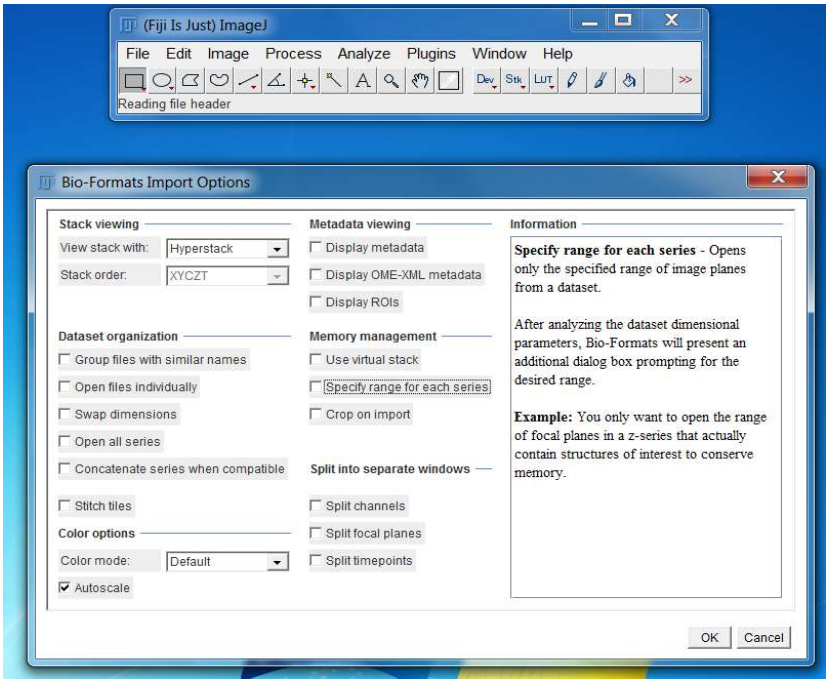

Fig. 1. Opening the TIRFM file in Fiji.

2) Click on OK to import the lif file using BioFormats.
3) Select the series that you want to process and click on OK (Fig. 2). If you want to design a mask for the analysis of this video, then you need to import also a multichannel image with the different chromophores (in the example below, Series 3 is the multichannel image and Series 4 the corresponding video).

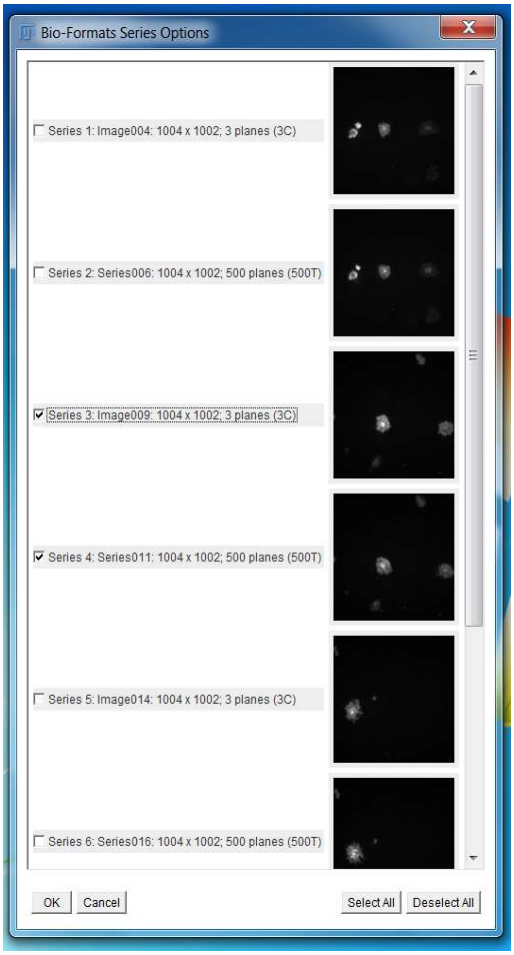

Fig. 2. Series selection.

The video (and the multichannel image) should open as an ImageJ stack. In the example below (Fig. 3), the image is on the left and the video is on the right.

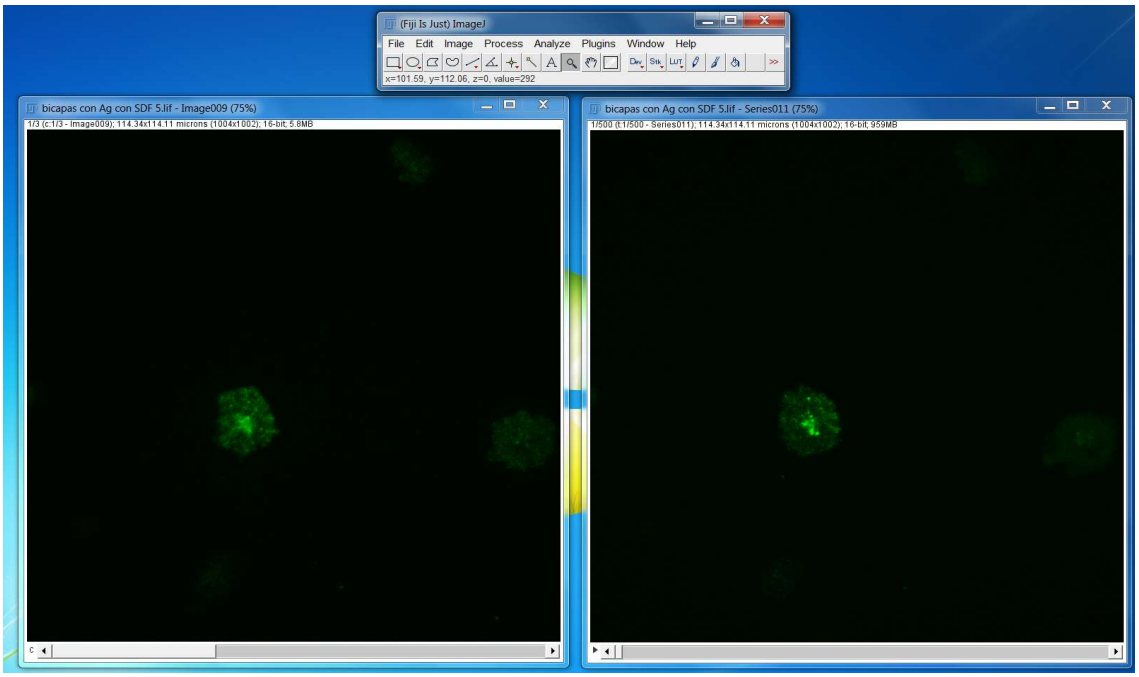

Fig. 3. Result example of series selection.

4) If you do not want to create a mask for the video, go to Step 6. To create the mask, we need to create a single image with the channels useful for the design of the mask. In this case, the interesting channels are the red, green and gray ones. Split the channels from the multichannel image (Fig. 4).

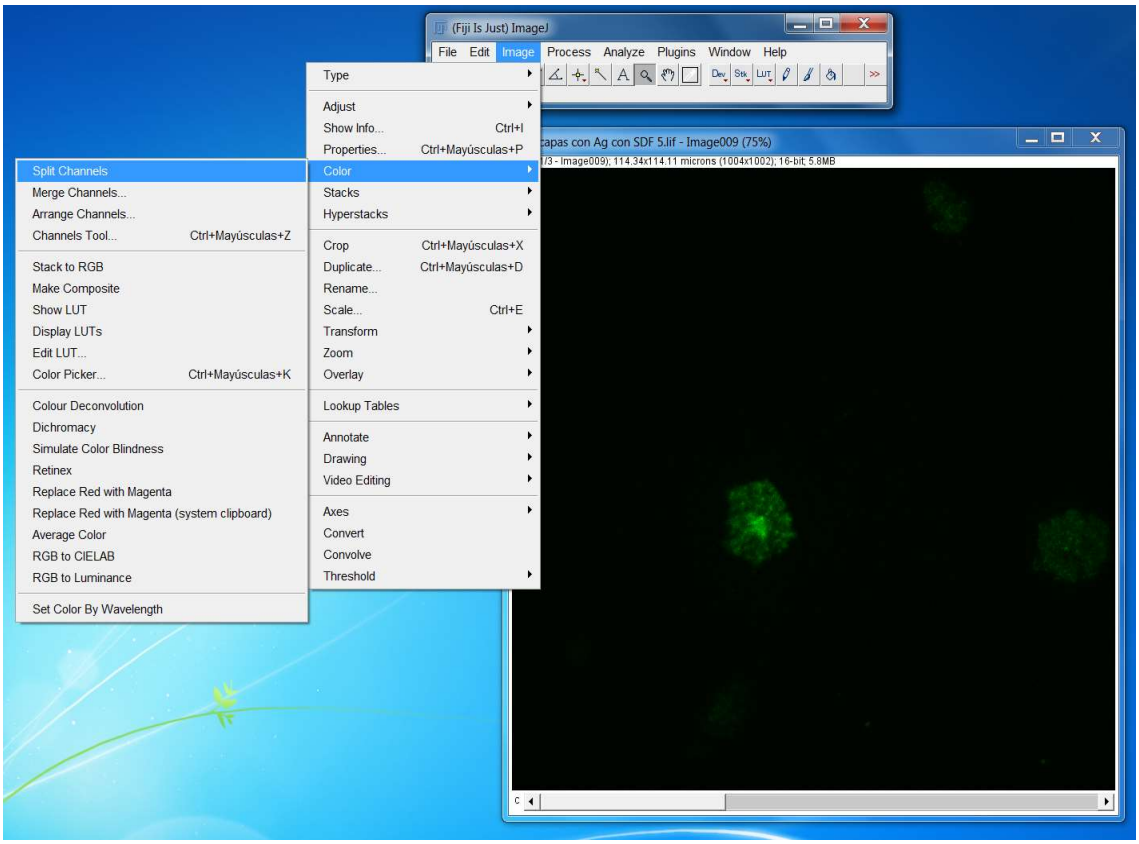

Fig. 4. Splitting the channels of a multichannel image.

The different channels should be shown as separate images (Fig. 5)

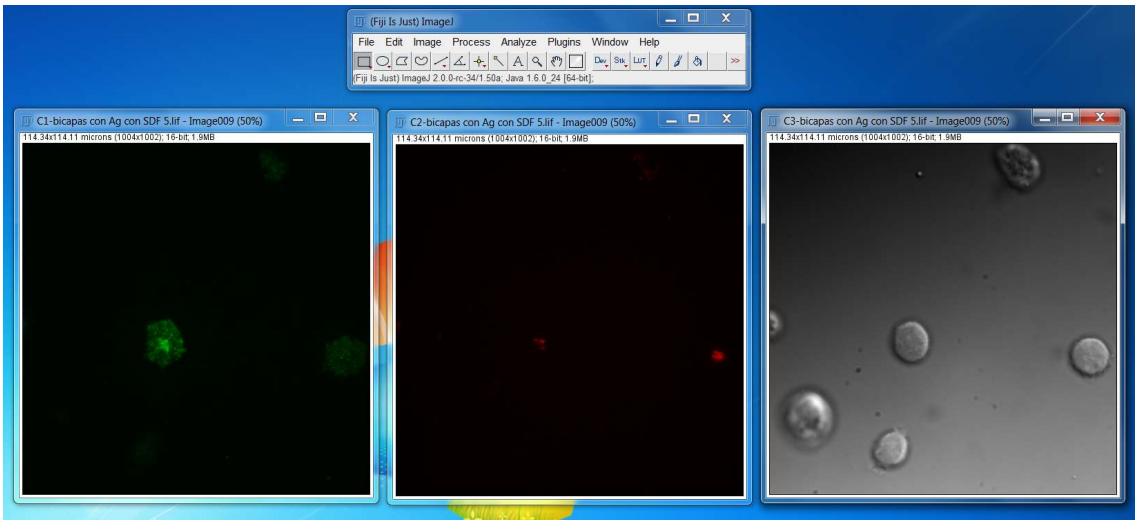

Fig. 5. Result example of the cannel split.

Now, we merge again the three channels in a single image (Fig. 6, note that the difference with the previous multichannel image is that the new image is not a stack).

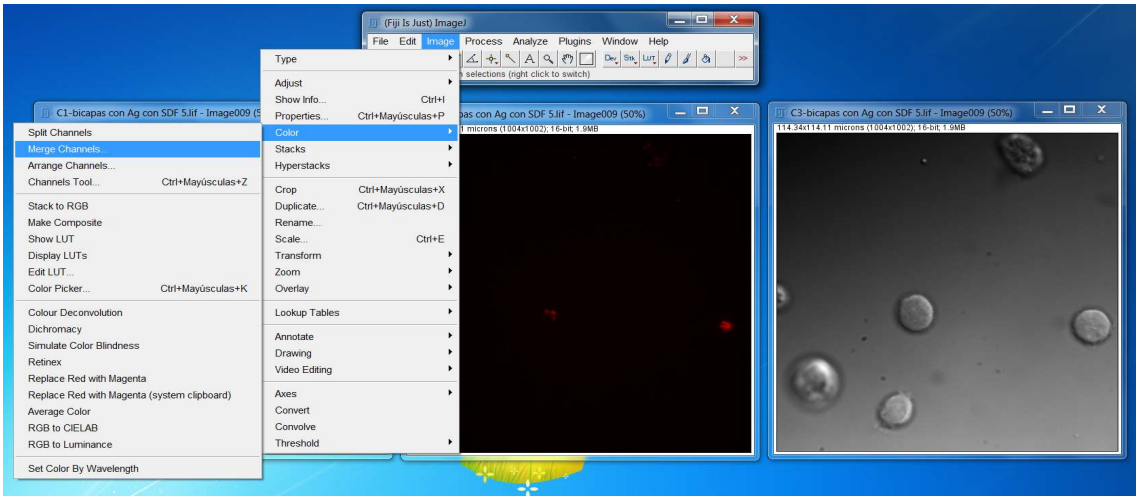

Fig. 6. Merging different channels into a single image.

Set the different images in the corresponding channels (Fig. 7)

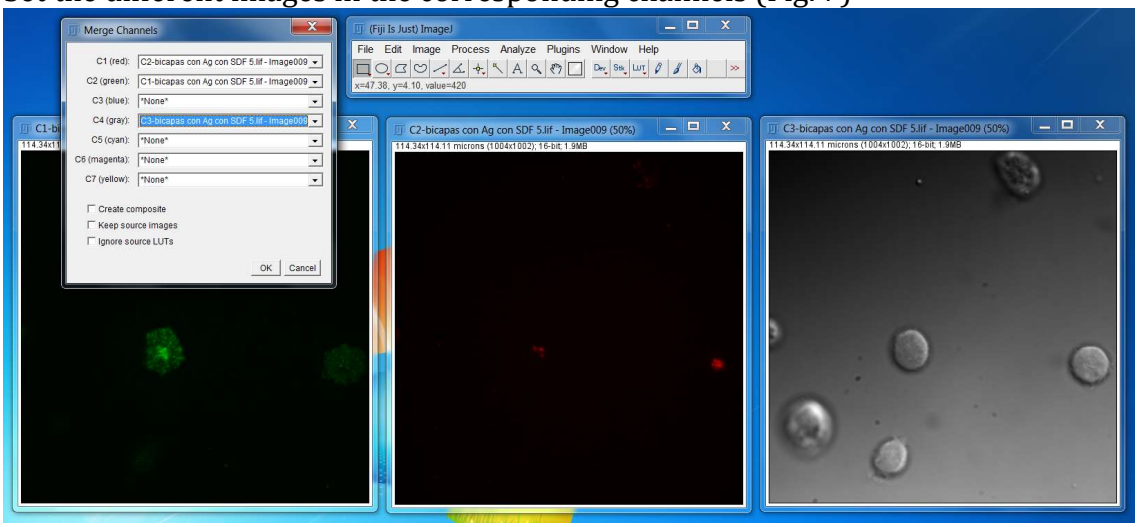

Fig. 7. Channel selection for the merging.

And press OK (Fig. 8).

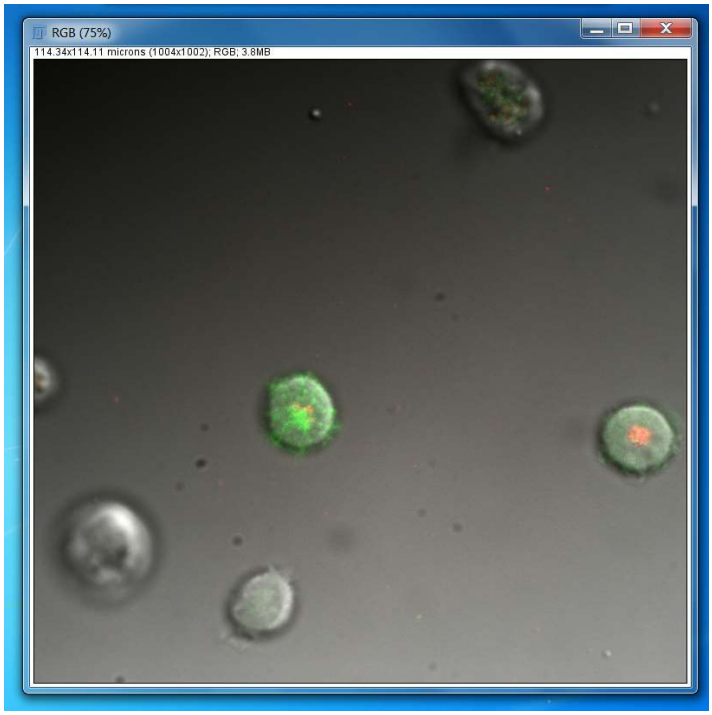

Fig. 8. Result example of the channel merge.

5) Now, we will synchronize the two windows so that we can crop the same region in both windows (Fig. 9).

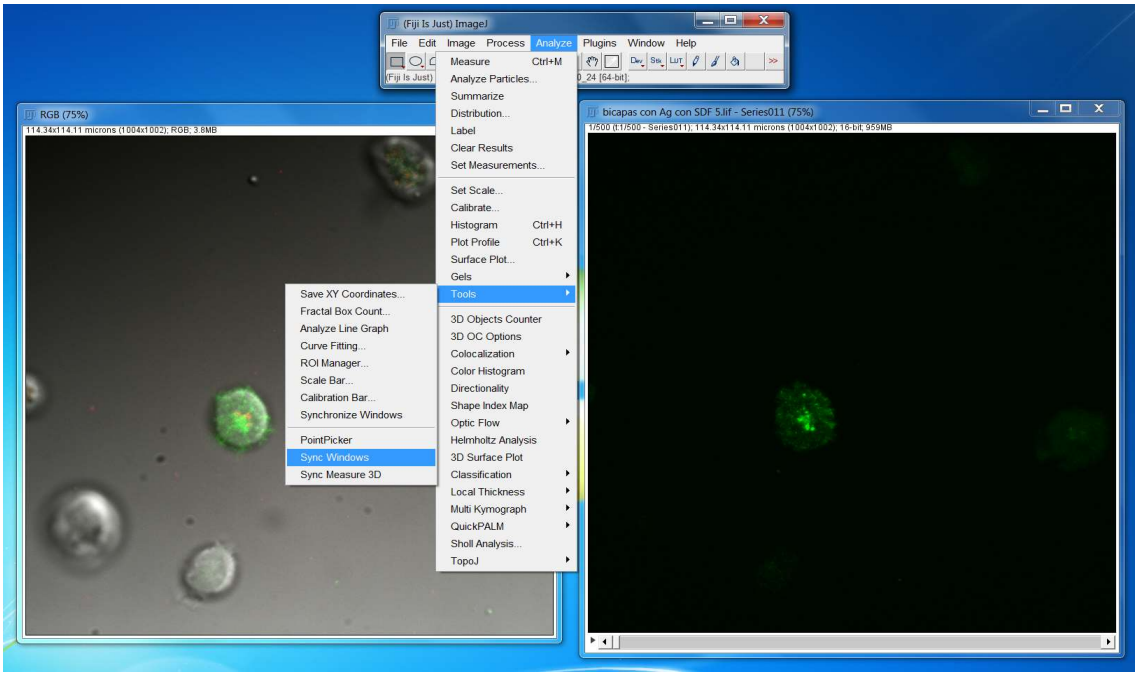

Fig. 9. Window synchronization.

6) With the two Windows synchronized (only the video if there is no multichannel image associated), draw the region of interest with the rectangular selection tool of ImageJ (Fig. 10).

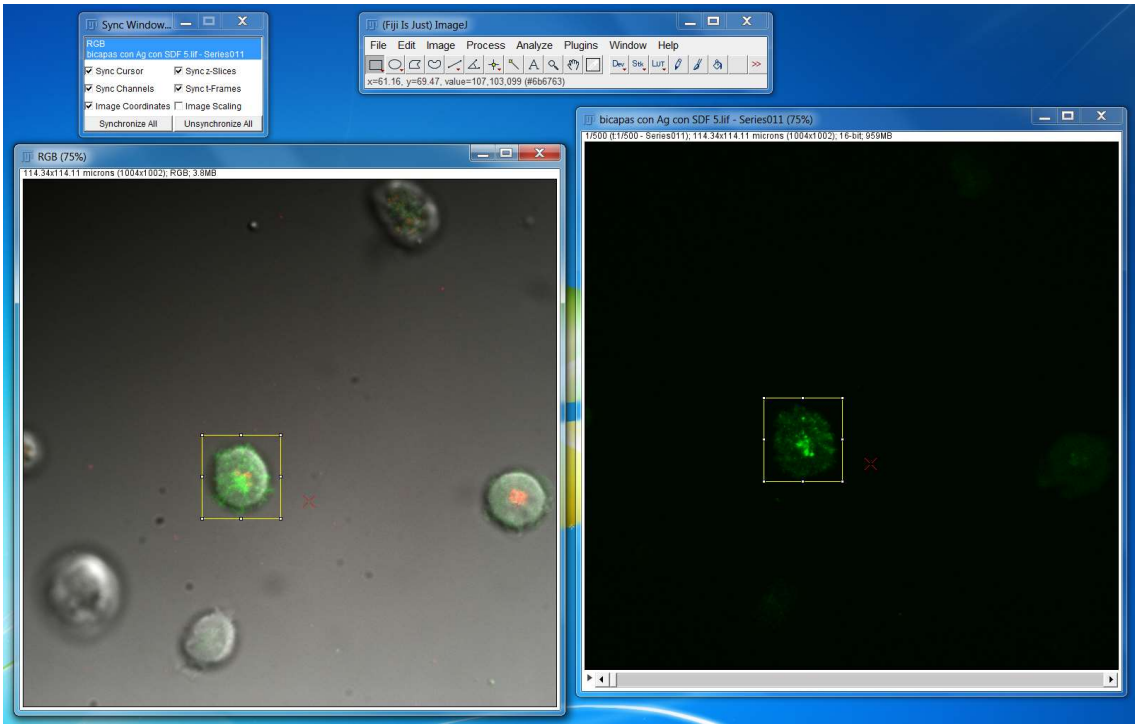

Fig. 10. Selection of the región of interest.

7) Crop the two (one) images (Fig. 11)

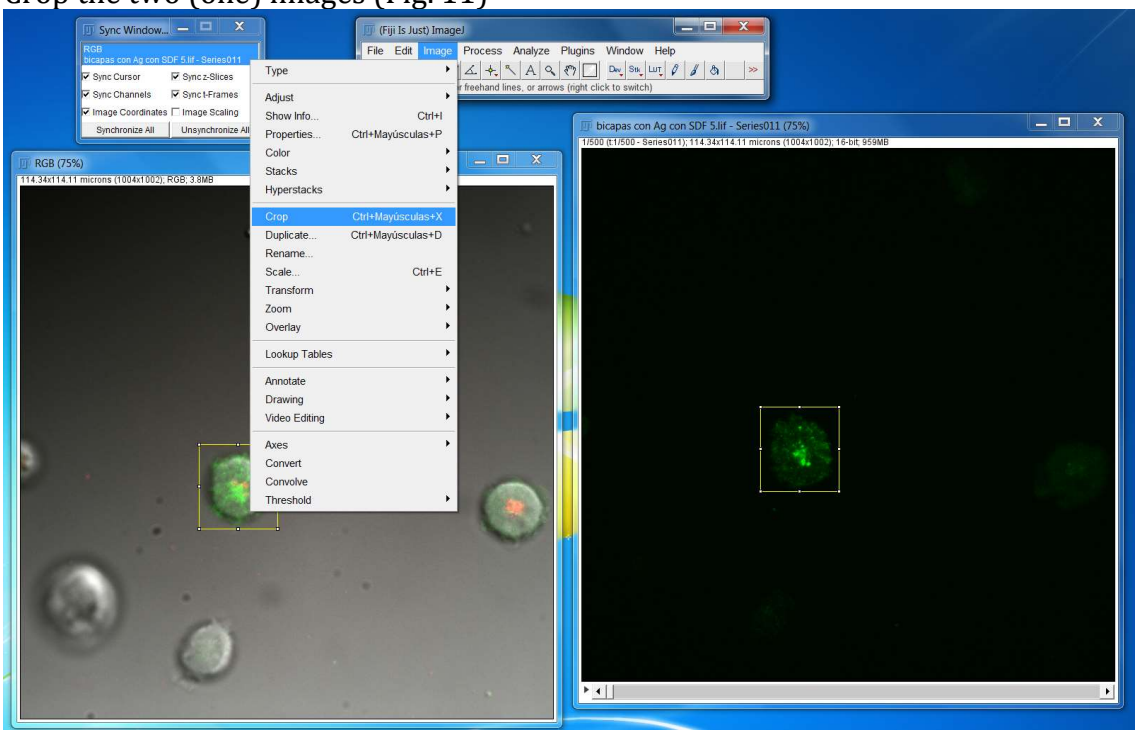

Fig. 11. Cropping the region of interest.

The video and multichannel image should now be of a smaller size (Fig. 12). At this moment you may unsynchronized both windows by pressing the "Unsynchronize" button in the "Sync Window" manager.

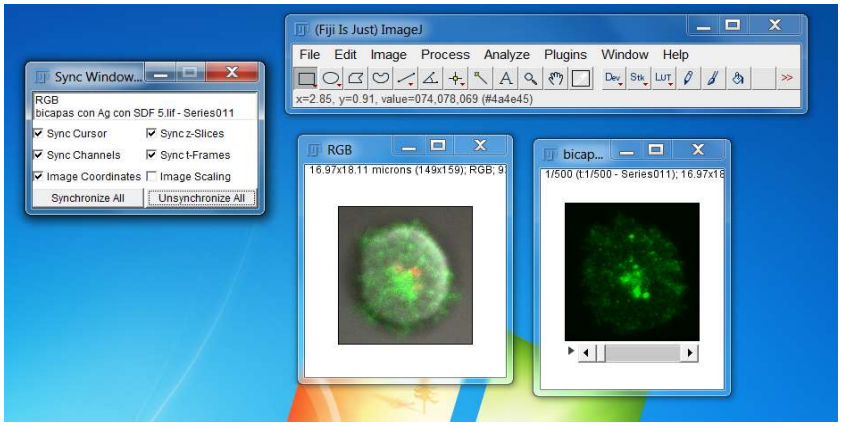

Fig. 12. Result example of the image crop.

8) Save the video as an "Image sequence" in a directory under the video directory (Figs. 13 and 14). The sequence must be alone in its directory to be successfully used by U-track.

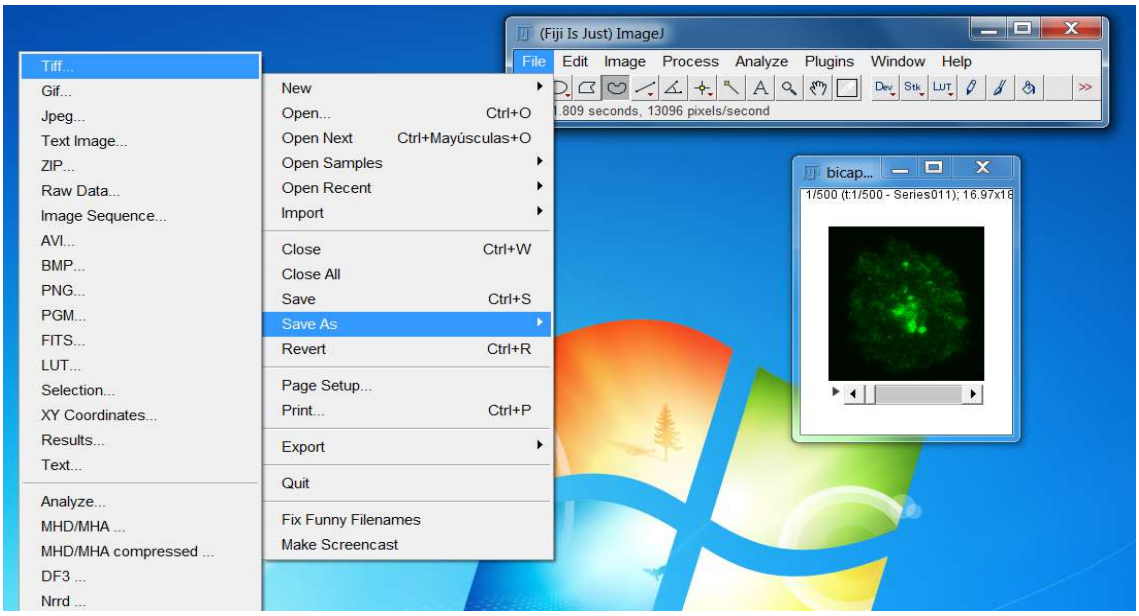

Fig. 13. Saving the region of interest video as an image sequence.

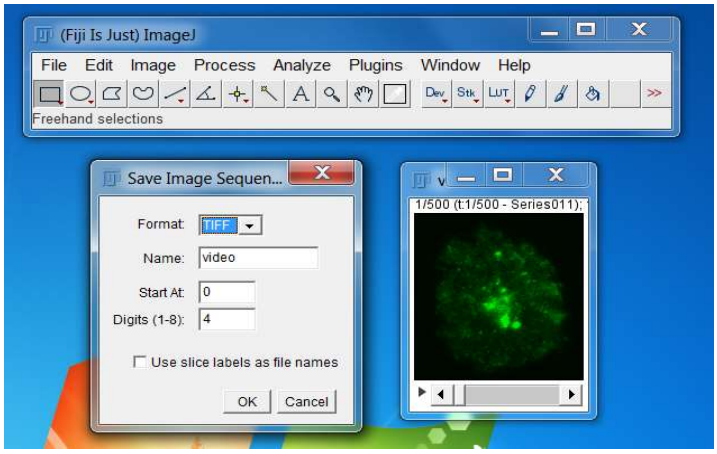

Fig. 14. Parameters for the Save as Image Sequence window.

9) If you are not going to design a mask for the video, go to Step 14.

10) With the multichannel image selected, open the Segmentation Editor plugin (Fig. 15)

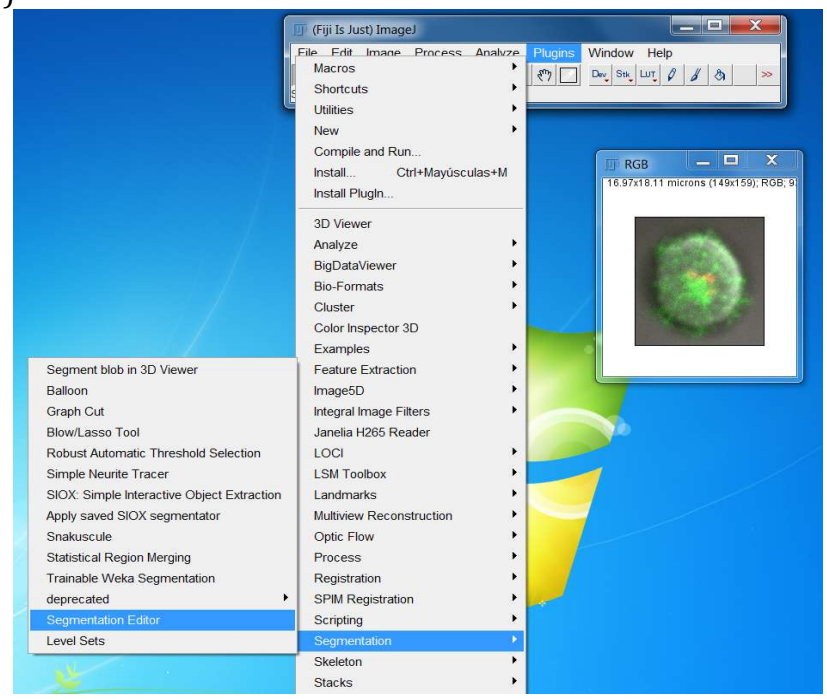

Fig. 15. Opening the segmentation editor.

11) Add and rename the labels of the segmentation as necessary. For doing so, right click on the labels of the Segmentation editor (Fig. 16).

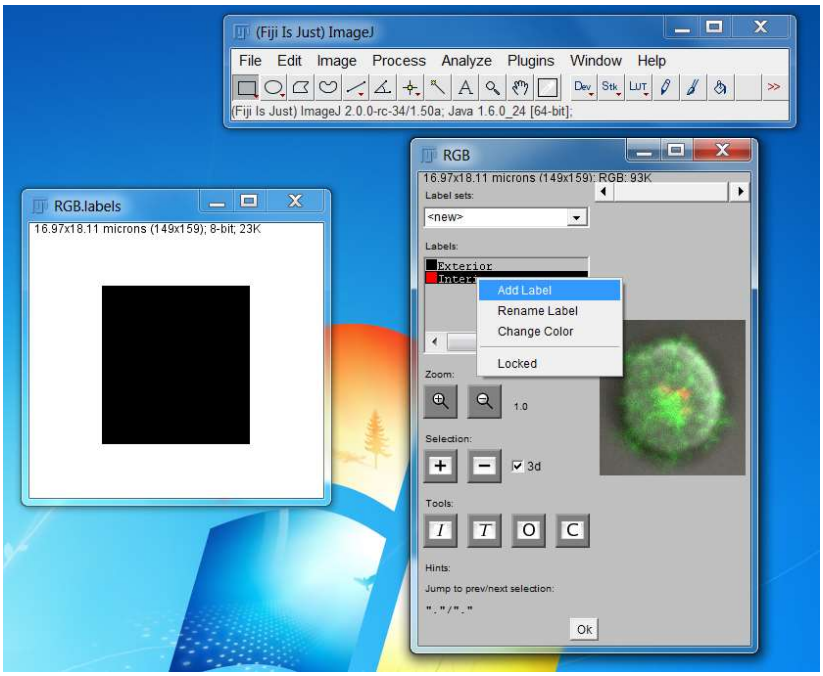

Fig. 16. Adding labels for the segmentation.

In our example, it should be as in Fig. 17.

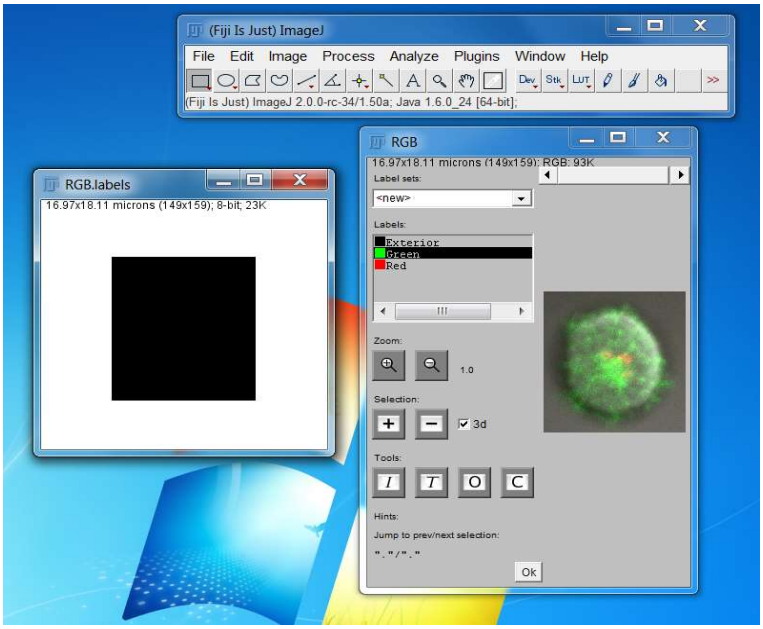

Fig. 17. Example of labels.

12) To design the mask, choose the appropriate selection tool in ImageJ (we will use freehand), and design first the outermost mask (Fig. 18). Once designed, press the + button. Actually, we only need to design the mask for the Green and Red labels, because the Exterior mask will occupy the rest of the image.

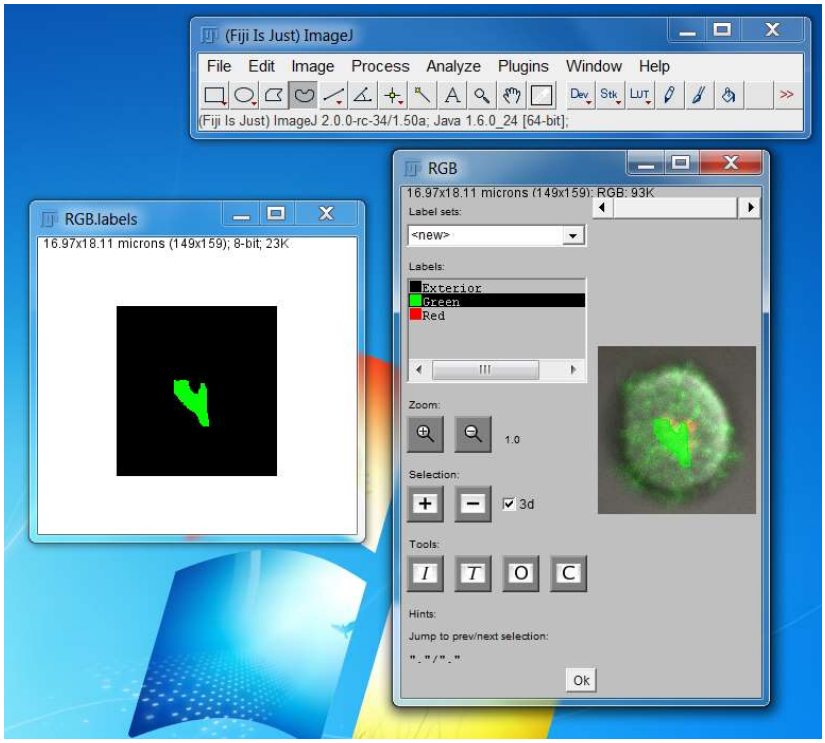

Fig. 18. Mask design.

Select the Red label and design also the corresponding mask (Fig. 19). Masks are coded in the image as regions 0, 1, 2, … according to the order of the labels in the RGB labels window.

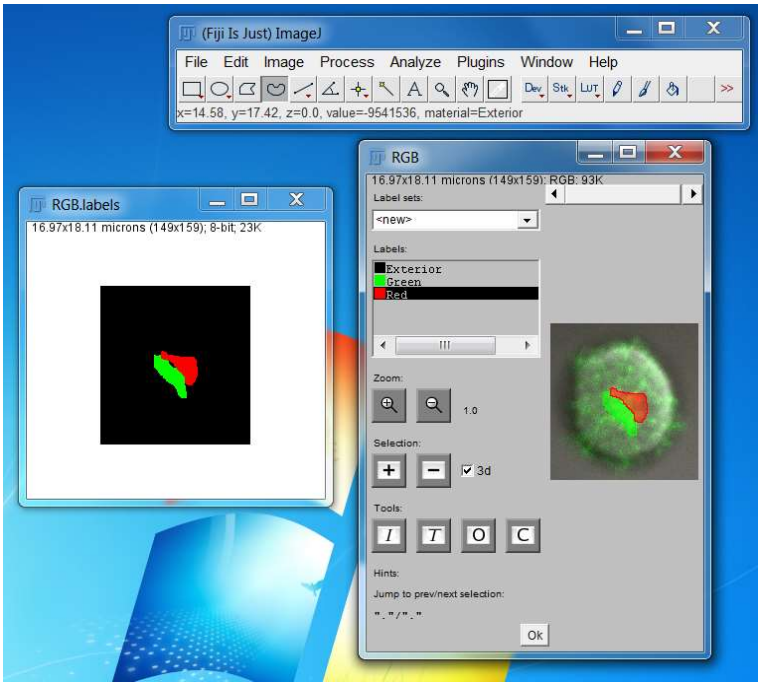

Fig. 19. Example of two masks corresponding to two labels.

13) When all masks for the different labels are designed, save the mask with the same filename as the video, with the name “mask.tif” (Fig. 20).

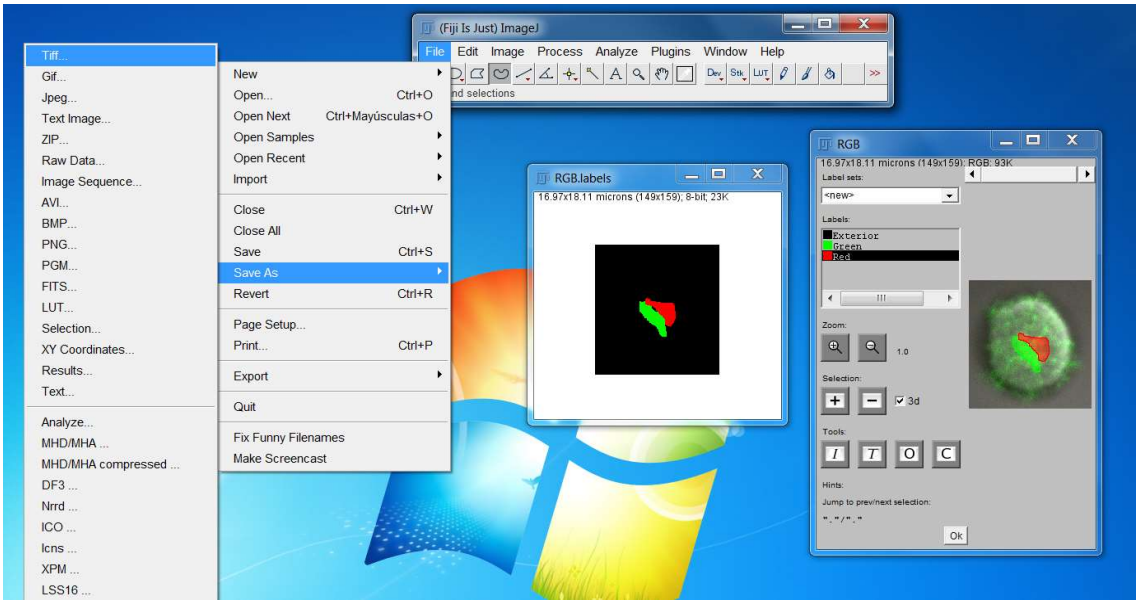

Fig. 20. Save the masks as a Tiff file.

14) Create a directory for the results of the analysis which will be called results. Make sure that the file structure at this moment is the following:

VideoName/video.lif
VideoName/Series4/mask.tif
VideoName/Series4/videoSeq/video0000.tif
VideoName/Series4/videoSeq/video0001.tif
...
VideoName/Series4/videoSeq/video0500.tif

VideoName/Series4/results/

The first file is the input .lif video with all the TIRF acquisitions performed at the microscope. The Series4 directory contains the series we are analyzing. The mask.tif is an image with the mask as designed in Steps 10-13. The video*.tif is the video as saved in Step 8. Names in green in the list above are fixed, i.e., they have to be called in this way because these are the names sought by the scripts. Names not in green can change to reflect the experiment performed.

15) We will now track all the particles seen in our video using a Matlab tool called U-track, which can be downloaded from http://lccb.hms.harvard.edu/software.html [Jaqaman2008]. For installing U-track, download .zip file from the web page above and uncompress the file in a directory of your choice. Then, open Matlab and add this directory to the Matlab Path by using Set Path -> Add with Subfolders option in the Matlab menu. Make sure to save the Matlab path so that in future executions of Matlab U-track is in the path. This path setting needs to be done only once.

16) Change the working directory in Matlab to the directory containing the Series to be analyzed (Fig. 21)

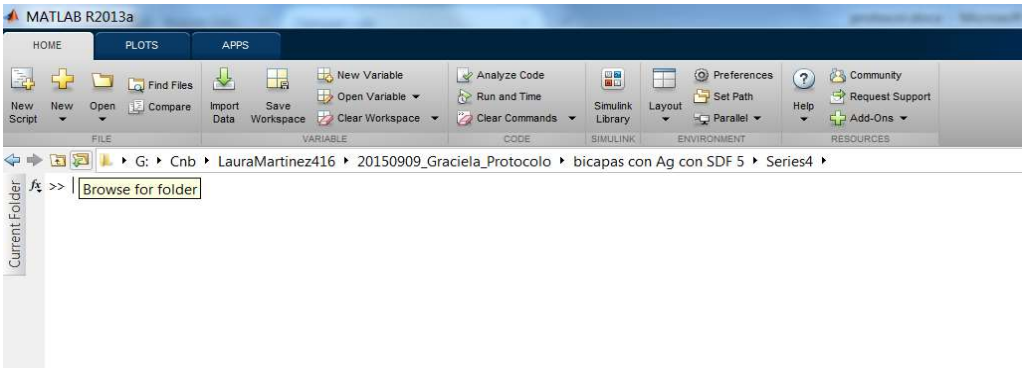

Fig. 21. Changing Matlab working directory.

17) Invoke U-track by typing `movieSelectorGUI` in the Matlab console. After pressing enter, the "Movie selection" window (see Fig. 22) will be opened. The press on the "New" movie button and the "Movie edition" window will appear. Press on "Add channel" choose the directory with the video (VideoName/Series4/video) and fill the movie information parameters. These parameters can be obtained from the microscope and the acquisition conditions. Set the output directory for the results of U-track to "results".

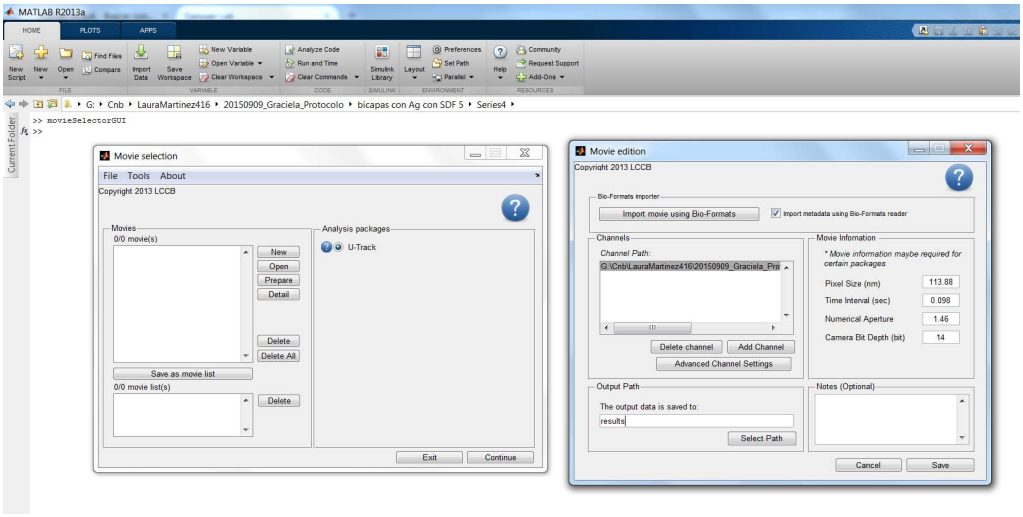

Fig. 22. U-track window.

18) Press on the "Advanced channel settings" and fill the parameters related to your acquisition. In our case, we do as shown in Fig. 23.

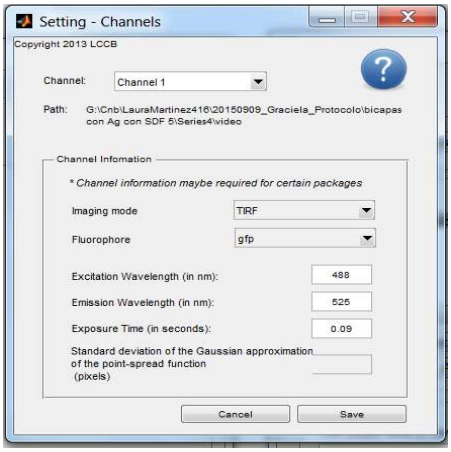

Fig. 23. Example of Channel settings.

Press "Save" on the "Advanced channel settings" window, and "Save" on the "Movie edition" window. The program will ask for confirmation of writing the file called "movieData.mat" on the "results" directory.

19) After creating the movie, press on "Continue" in the movie selection window. U-track will ask about the type of object to be analyzed. Choose "Single-particle". Then, the "Control panel" window will appear (see below). Select the first step "Step 1: Detection" and press on "Setting". The "Setting Gaussian-Mixture Model Fitting" window will appear. This window controls how U-track will detect single particles on the video. This is done by fitting a Gaussian mixture model as described in [Jaqaman2008]. One of the key parameters for this fitting is a filter to help identifying local maxima, this depends on the image contrast and the noise present in the images. In our videos, we set "Alpha value for comparison with local background" to 0.001 and "Rolling-Window time averaging" to 3 (see the image below). The first parameter controls the confidence of a maxima being a real spot, while the second helps to reduce noise during the identification of spots. Then press

on “Apply” in the “Settings” window and “Run” in the “Control panel”. With the configuration in Fig. 24, only the “Detection” step should be run. This step should take a few (2-5) minutes.

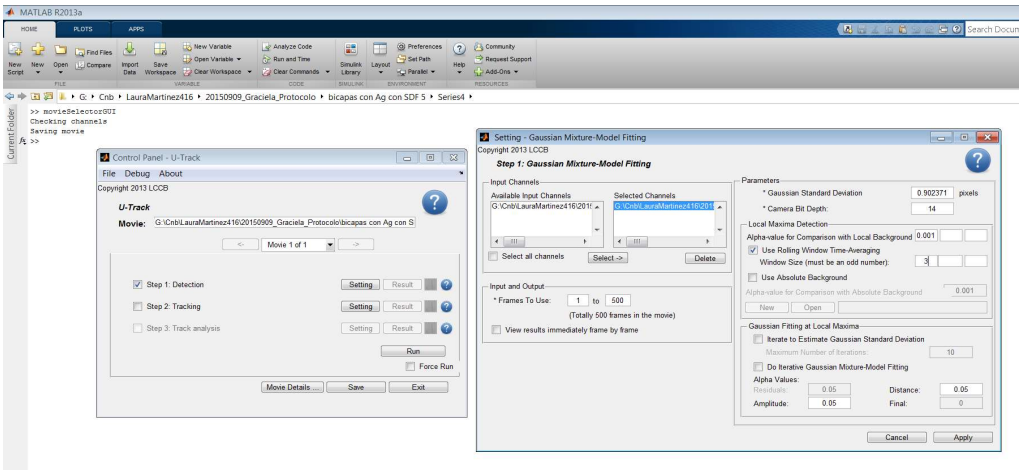

Fig. 24. Example of settings for the detection of particles.

We may check the results by pressing the “Result” button of the Step 1 (Fig. 25).

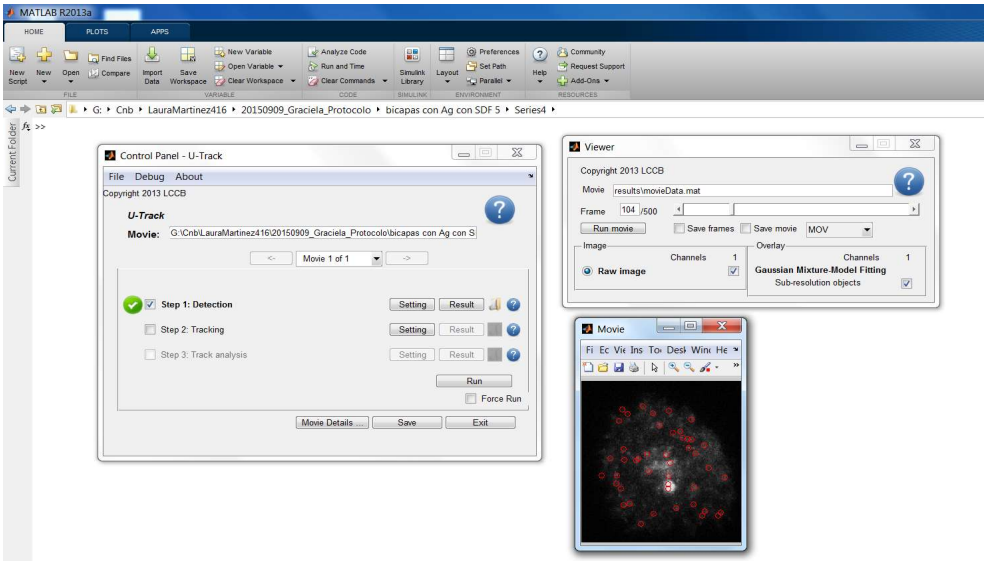

Fig. 25. Results example for the particle detection.

As shown above, the movie should show red circles on the detected particles. If no red circle is shown, then this step has not worked correctly.

20) We will now perform the identification of tracks, that is, merging the particles detected in the previous step into tracks that span multiple frames. This is the Step 2 of U-track whose settings have to be defined as shown in Fig. 26.

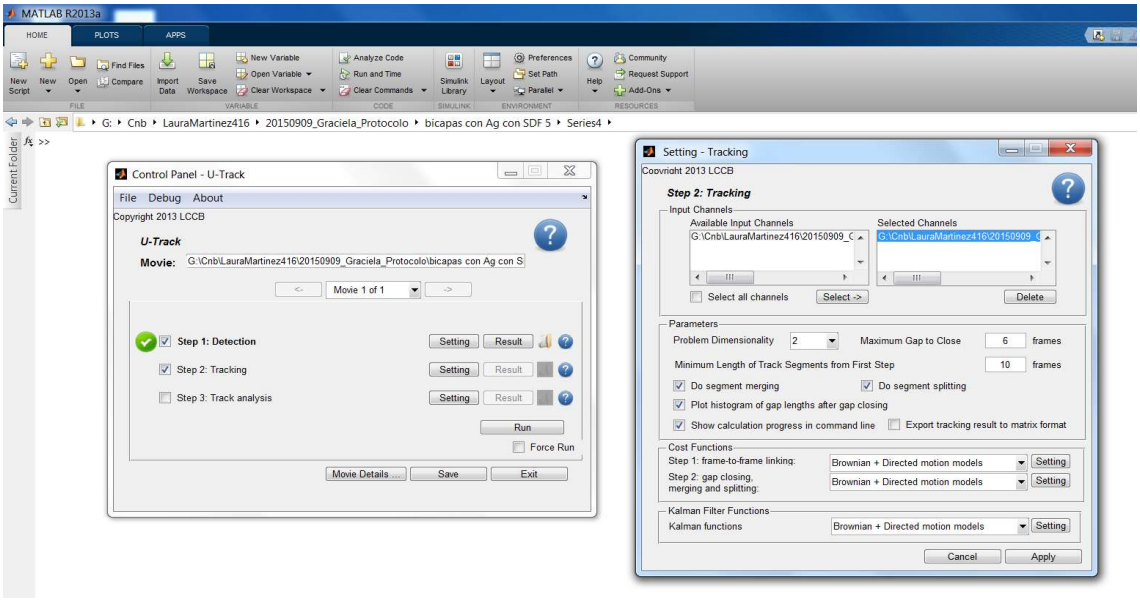

Fig. 26. Example of settings for the tracking of particles.

Important parameters at this point are the number of frames to close gaps (that is a track may span over frames in which the particle is not actually seen, but it is seen before these frames and after these frames; in our example, 6), and the number of frames that a track must span in order to be considered as a successful track (in our example, 10; this parameter is related to the camera acquisition speed and the number frames in the track must be such that it allows the calculation of the diffusion coefficient). It is also important to decide whether to merge and split segments (in our example, we have chosen these two possibilities). Then, we must also set the parameters for the Step 1 in Cost functions, this function controls how the particles are tracked along the frames. In our case we set it as shown in Fig. 27.

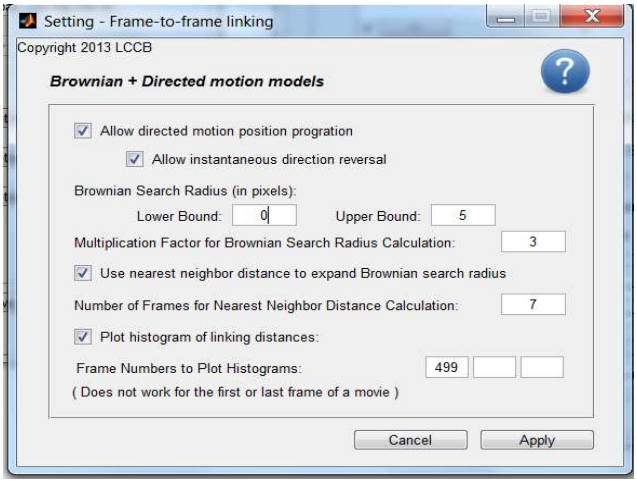

Fig. 27. Example of settings for the cost function of Step 1.

The most important parameters at this point is the selection of the Brownian search radius, which control how far each spot is expected to be in the next frame. In our example we chose 0 and 5 as lower and upper bounds, respectively.

The Step 2 cost function settings must be set as shown in Fig. 28.

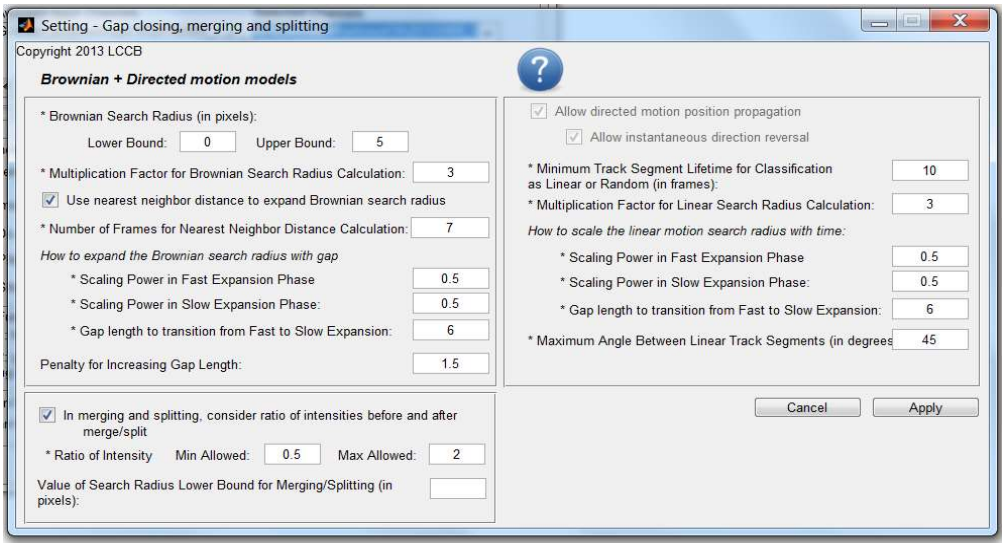

Fig. 28. Example of settings for the cost function of Step 2.

Note that these parameters are very specific to the nature of the particles being tracked, and that they may have to be tuned in each specific case. Particularly important are the scaling power in the Brownian and Linear search radii. These scaling powers depend on the kind of movement of the particles (free or confined diffusion). We chose the values (0.5, 0.5) corresponding to free diffusion. For the specific documentation of these parameters, see [Jaqaman2008].

After setting the parameters for Step 2, we press “Run” in the “Control panel” and only Step 2 should be run. It should take a few seconds, and after running it shows a histogram of the number of gaps filled in the tracks detected. This histogram should be decreasing so that there are fewer long gaps than short gaps (as in Fig. 29).

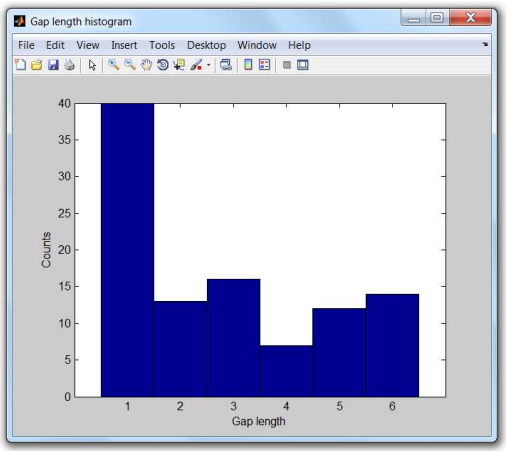

Fig. 29. Example of the gap length histogram.

21) Now we perform Step 3. The settings should be defined as shown in Fig. 30.

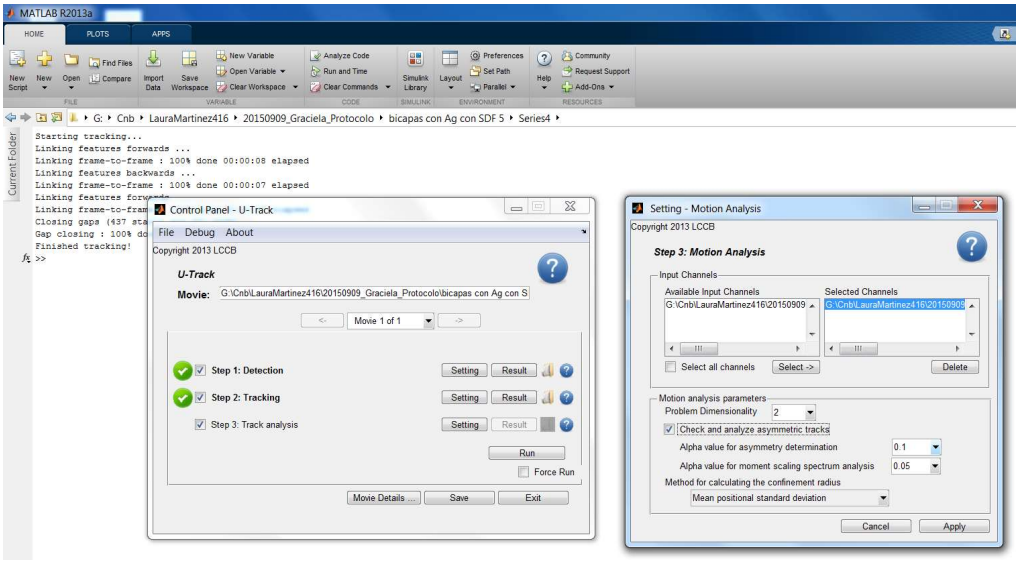

Fig. 30. Example of settings for the track analysis.

Then, press “Apply” and “Run”. This step should take a few seconds.

22) At this point we verify with the “Result” button that the process has correctly identified all the tracks. For doing so, we click on “Show track number” of the “Movie options” window, and check frame by frame that each track has been correctly identified (Fig. 31). We manually annotate those particles that are not true particles. If this manual selection is not done, a weaker automatic selection can be performed later when the diffusion coefficient is calculated (see Step 25).

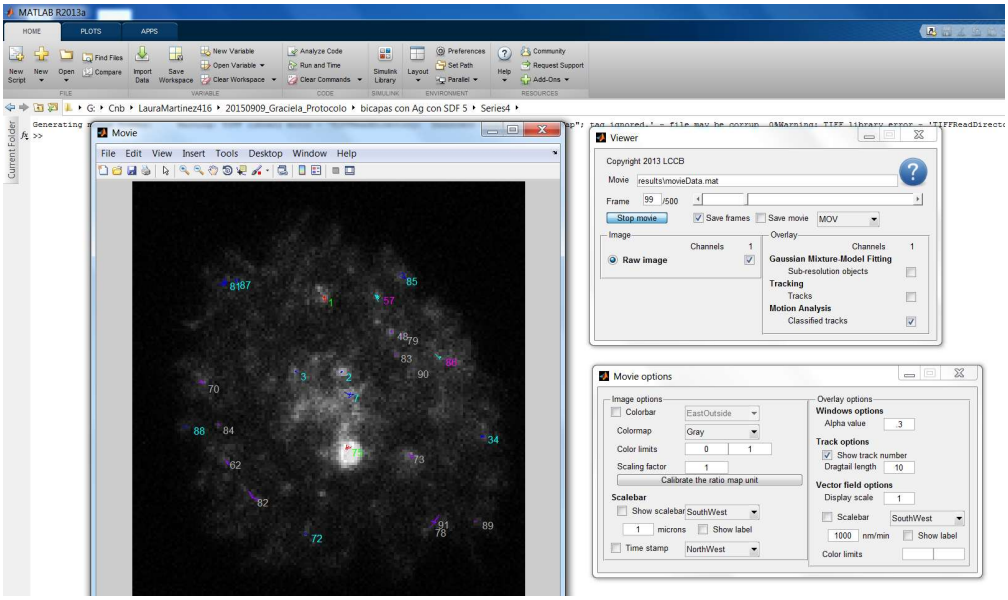

Fig. 31. Example of the verification of the track analysis.

Calculation of the diffusion coefficients and classification of trajectories

23) We now read all the trajectories to compute the diffusion coefficients. We do so by issuing in the Matlab console the command

```
trajectories=readTrajectories(0.098);
```

where the 0.098 is the time in seconds between two consecutive frames. If some trajectories have to be removed (because the corresponding spots were incorrectly identified), this is the moment to delete them. You may do so by giving a list of the spots to exclude. For instance to exclude spots 4, 5 and 28 you type

```
trajectories=readTrajectories(0.098, [4, 5, 28]);
```

24) We now calculate the diffusion coefficients for each one of the tracks of this cell. We calculate the diffusion coefficient for a time lag = 4, it is called $D_{1\text{-}4}$. For doing so, we run in the Matlab console the command

```
D=calculateDiffusion(trajectories, 113.88e-3, 0.0015, 'alpha');
```

where trajectories are the trajectories read in Step 23, 113.88e-3 is the pixel size of the acquired images in microns, 0.0015 is an upper bound for the diffusion coefficients of immobile particles measured in $\mu m^2/s$, and 'alpha' is the fitted model as explained below. The upper bound of the diffusion coefficient of immobile particles can be known from previous experiments or by running the calculateDiffusion function on a separate project with immobile particles and seeing the diffusion coefficients reported. This function takes two extra parameters outputSuffix, which is added to the output filenames and by default takes the empty value, and the plotlength, which by default is 13 and is the number frames in which the diffusion calculation is performed. If you have a faster camera and need more frames to calculate the diffusion parameter you may increase it, e.g. to 20, by

```
D=calculateDiffusion(trajectories, 113.88e-3, 0.0015, 'alpha', '', 20);
```

In a simplified manner, the fitting mode 'alpha' implies an adjustment of the MSD to the curve

$$MSD = MSD_0 + 4Dt^{\alpha}$$

that is, an offset (MSD_0) and a power function of the time lag. The exponent of this power function, α, determines whether the movement is confined ($0 < \alpha < 0.6$), anomalous ($0.6 < \alpha < 0.9$), free ($0.9 < \alpha < 1.1$), or directed ($\alpha > 1.1$). For a review of this kind of analysis the reader is referred to [Manzo2015].

The calculation of the diffusion coefficients [Bakker2012] produces two output figures (see Fig. 32). The first one shows a histogram of the diffusion coefficients calculated (note that at this point, there is an independent diffusion coefficient for each trajectory). The mean and 95% percentile of these coefficients are shown in the Matlab console (in our example 0.011723 and 0.04969 $\mu m^2/s$). The second plot shows the Mean Squared Displacement (MSD, red curve) and the number of steps considered to calculate it (green dashed curve) as a function of the time lag for those trajectories considered to be mobile (those whose diffusion coefficient is above the threshold given

in the command line). The average and standard deviation of the mobile trajectories is computed (in our example the mean is 0.043 and the standard deviation 0.0004, remind that these are the values for the trajectories in a single cell).

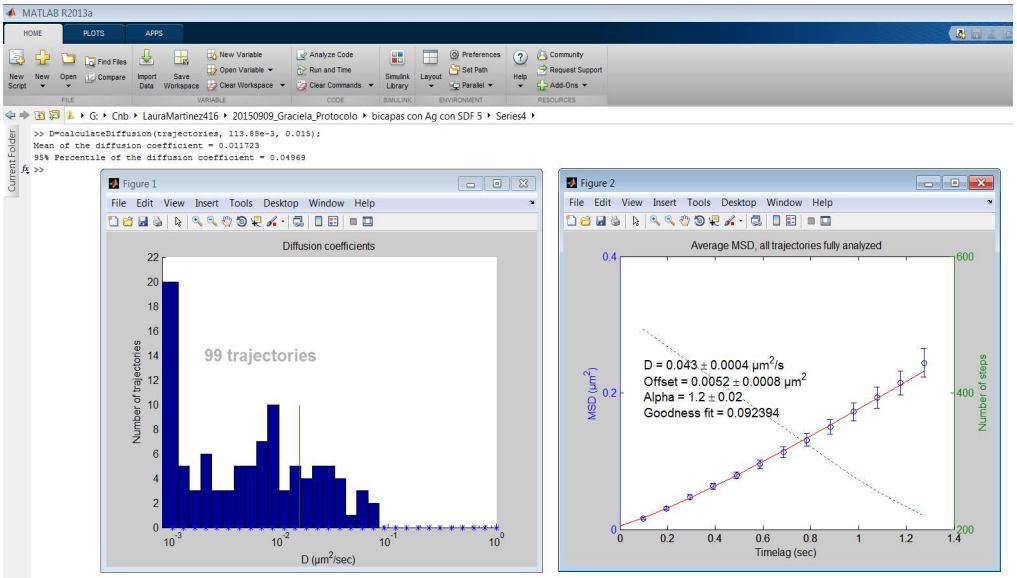

Fig. 32. Results example of the calculation of the diffusion coefficients.

In our case, the exponent is 1.2 meaning that the movement is directed. For this curve fitting, the program reports the uncertainty associated to each one of the parameters (shown as the standard deviation of each one) and the goodness of fit (a perfect fit would reach zero).

At this point and after checking the value of alpha we may decide to fit the MSD with a different function:

$$MSD = MSD_0 + < r^2 > \left(1 - Ae^{-\frac{4BDt}{<r^2>}}\right) \quad \text{if } 0 < \alpha < 0.6 \text{ (confined)}$$

$$MSD = MSD_0 + 4Dt \quad \text{if } 0.9 < \alpha < 1.1 \text{ (free)}$$

$$MSD = MSD_0 + 4Dt + (vt)^2 \quad \text{if } \alpha > 1.1 \text{ (directed)}$$

Note that anomalous trajectories cannot be fitted with a different function. In case of confined particles you may calculate the confinement size (in microns) as [Destainville2006]

$$L = \sqrt{3(< r^2 > + MSD_0)}$$

We can fit these functions by calling the `calculateDiffusion` function again with a different fitting mode ('confined', 'free', or 'directed'). In our case, we call it with 'directed'

```
D=calculateDiffusion(trajectories, 113.88e-3, 0.0015, 'directed');
```

Obtaining the fitting results for the directed model, as shown in Fig. 33.

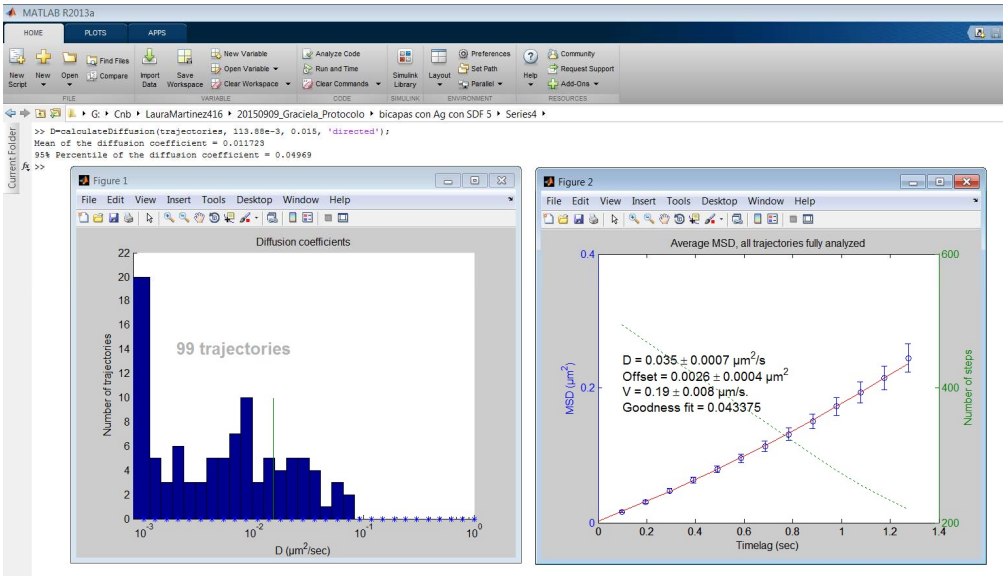

Fig. 33. Results example of the calculation of the diffusion coefficients of the directed trajectories.

A good indicator of a correct fitting is that the goodness-of-fit should decrease from the alpha fitting to the final fitting (in our example, the goodness fit falls from 0.092394 to 0.042375).

`calculateDiffusion` creates two files in the "results\TrackingPackage\tracks" inside the series directory called "diffusionCoefficients.txt" and "diffusionCoefficientsMobile.txt". These files contain three columns: 1) the index of each input trajectory, 2) their corresponding diffusion coefficient, 3) the majoritarian region in the mask where this track belongs to (the regions are obtained from the file "mask.tif" that we generated in Step 13; if this file does not exist, then this column is not present). You may use this file and the analogous files generated for other cells to analyze the distribution of the diffusion coefficient for a set of cells under similar experimental conditions.

25) Now we may decompose the trajectories into short and long trajectories. We do so by the command

```
[shortTrajectories, longTrajectories]=...
    separateTrajectoriesByLength(trajectories,50);
```

where 50 (in our example) is the minimum length in frames of a trajectory to be considered long.

Short trajectories can be studied using the same fitting procedure described in Step 24.

```
D=calculateDiffusion(shortTrajectories, 113.88e-3, 0.0015, 'directed',
'Short');
```

The last parameter 'Short' is a suffix added to the output filename so that you may analyze different subsets of trajectories without overwriting the diffusionCoefficients.txt files. The actual name of the output files are "diffusionCoefficients<Suffix>.txt" and "diffusionCoefficientsMobile<Suffix>.txt". If no suffix is given, as in the general analysis performed above, then an empty suffix is assumed.

As shown in Fig. 34 the model parameters (D, v, and MSD_0) may differ from those fitted to all trajectories. Most remarkably the standard deviation of the parameters as well as the Goodness fit normally increase. The reason is that short trajectories are more unstable and a reliable fitting is more difficult.

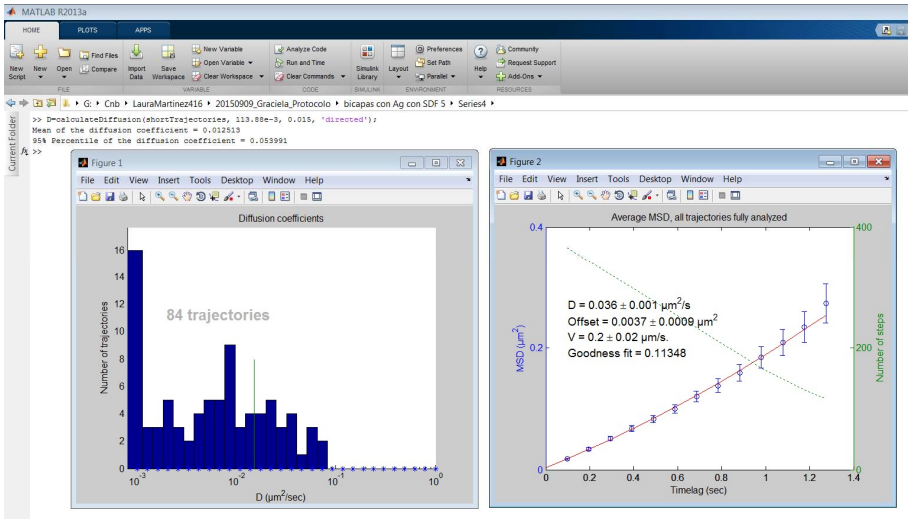

Fig. 34. Results example of the calculation of the diffusion coefficient of short trajectories.

Similarly, short and anomalous trajectories can be analyzed with the command

```
D=calculateDiffusion(shortTrajectories, 113.88e-3, 0.0015,
'alpha', 'Short').
```

Long trajectories are studied through their Moment Scaling Spectrum [Sbalzarini2005]. Trajectories are classified into confined (1), free (2), or directed (3) according to their first moment and its location with respect to the 2.5% and 97.5% percentiles of the first moment of 500 random paths with Brownian motion, the same diffusion coefficient and length as the trajectory being analyzed and simulated by Monte Carlo, if the first moment of the path being analyzed is below the 2.5% of the first moments observed in the simulations, the path being studied is classified as confined; if it is above the 97.5% of the simulated first moments, it is classified as directed; otherwise, the path is classified as Brownian. The command

```
trajectoriesClassification=classifyLongTrajectories(longTrajecto
ries,113.88e-3,0.0015,'Long')
```

shows in screen and generates a file called "trajectoryClassification<Suffix>.txt" in the directory "results\TrackingPackage\tracks". 113.88e-3 is the pixel size in μm, 0.0015 is the upper bound of the diffusion coefficient of immobile spots measured in $\mu m^2/s$, and 'Long' is the suffix for the output filename. The first column of this file is the trajectory number, the second is its classification (1=confined, 2=free, 3=directed), and the third is the trajectory first moment.

Calculation of cluster size through the particle intensity

26) We now analyze the intensity of each particle along their trajectory. For doing so, we need to invoke the script `analyzeSpotIntensities` that takes as input the trajectories calculated by U-track in the first section. This script is highly flexible and allows tracking the fluorescence intensities in many different ways (each one well suited for different situations like tracking immobile spots of a control experiment or tracking highly movable spots over a cell with variable fluorescent background). In its most basic form, you simply need to call the script without any argument from the directory of the video being analyzed (in our example, VideoName/Series4)

```
analyzeSpotIntensities()
```

The script will analyze all trajectories along all frames. For each spot at each frame it will measure the intensity of the pixels around the spot (it analyzes a square patch around the spot, by default of a size 3x3 pixels), estimate the spot background (by a method that will be described below) and calculate the fluorescence difference between the spot and the background. The percentage of photobleaching particles and the amount of fluorescent particles in a single cluster, seen as a single spot, can be estimated in this way.

The script will print on screen all the information analyzed for each spot and each frame. An example of the kind of output produced is

```
spot=43 frame=184
       x=78.0397
       y=72.5395
       k0=1571
       spotRawIntensity=5550.1111
       spotCorrectedIntensity=3979.1111
       maxCorrectedSpotIntensity=6243
       maskRegion=2
```

For each spot and frame, the script shows the coordinate of the spot in that frame (x,y) in pixel units, the estimate of the fluorescence of the background (k0), the raw spot intensity in the 3x3 patch, the corrected intensity (calculated as the raw intensity minus its background), the maximum value of intensity observed in the patch, and the region number within the mask where this spot is located at this frame. All this information is stored in a log file (`results/TrackingPackage/tracks/log.txt`), and a table that can be read from Excel (`results/TrackingPackage/tracks/spotIntensitiesByFrame.txt`).

After analyzing each spot, the script prints the average intensity along the trajectory and the majoritarian region within the mask

```
spot=43
      meanCorrectedSpotIntensity along frames=4762.303
      majoritarian region=3
```

This information is stored in the log file above and a table that can be read from Excel (`results/TrackingPackage/tracks/meanSpotIntensities.txt`). We may now use this information to roughly estimate the size of the fluorescent cluster. If a spot's mean corrected intensity is around 4700 units, and the fluorescence of a monomer (measured through a similar but independent experiment) is 800, the cluster is formed by 6 monomers approximately.

This basic behavior can be configured in many different ways by providing arguments to the script as in

```
analyzeSpotIntensities('Arg1', Value1, 'Arg2', Value2, …)
```

Valid arguments are:

- 'spotRadius' (by default, 1 pixel). The fluorescence intensities are analyzed in a patch of size (2*spotRadius+1)x(2*spotRadius+1). Choosing a spotRadius of 1 results in a patch of 3x3 centered at the spot, a spotRadius of 2 results in a patch of 5x5 centered at the spot, etc.
- 'onlyInitialTrajectories' (by default 0, i.e., false). If this argument is given, then the script will only analyze the trajectories that start in the first frame of the video. This is useful to analyze control images.
- 'trackTrajectory' (by default 1, i.e., true). If this argument is set to false, then the coordinate of the spot in its first frame will be kept for all frames (this is useful for immobile spots). If the argument is set to true, then the spot is tracked along the video following the coordinates calculated by U-track.
- 'excludeTrajectories' (by default none). If you want to exclude some trajectories, as was done in Step 23, you may provide their index at this point (e.g., `'excludeTrajectories',[4,5,28]`)
- 'extendTrajectory' (by default 0, i.e., false). If this argument is set to true, then the intensity in the patch is analyzed to the end of the video (even if the trajectory stops earlier). The coordinate of the spot is either the last coordinate in the trajectory (if trackTrajectory is true) or the first coordinate in the trajectory (if trackTrajectory is false).
- 'subtractBackground' (by default 1, i.e., true). If this parameter is set, then the raw fluorescence measured at each spot is corrected by the estimate of the background fluorescence for that spot (see below).
- 'meanLength'(by default, full length). If this parameter is set, then the mean intensity spot is measured at the length indicated. For example, to measure the spot intensity at the first 30 frames you set, `'meanLength', 30`. If the argument is not set, then the spot intensity is calculated at the whole trajectory.

- 'showIntensityProfiles' (by default 0, i.e., false). If this argument is given, then the intensity profile along the different frames as well as their background are plotted. These plots are very useful to identify photobleaching as shown in the Fig. 35.

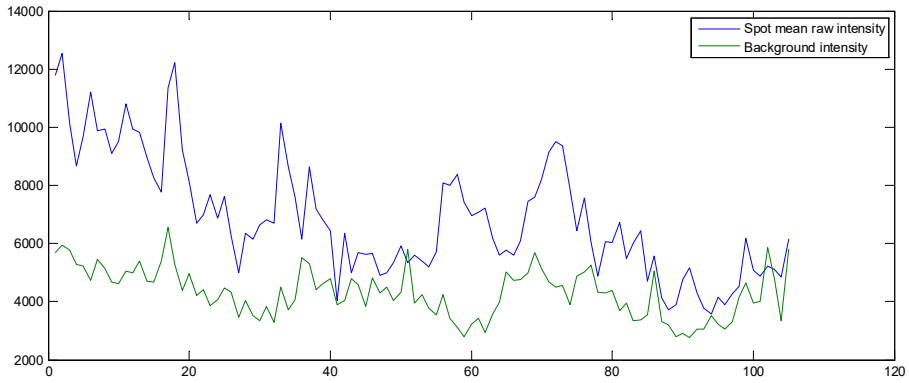

Fig. 35. Example of spot and background intensities.

For every path, the routine automatically analyzes if it is possible that there has been phtobleaching. This is done by comparing the intensity values with a Student's t in the first N and last N frames along the path. By default, N is 10, but this value can be modified through the argument 'Nbleach'.

One of the most flexible features is how to determine the background of each spot. This can be done in several ways, and which one to use can be selected through the argument 'backgroundMethod' which can take the values:

- 0: Manual identification for the whole video. The user is allowed to select 8 points in the first frame of the video. A patch around these points is analyzed along the whole video, and the 95% quantile of all these intensities is chosen as the background intensity for all spots.
- 1: Manual identification for each frame. The user can choose the 8 points for every spot and every frame. This is a time consuming task but it gives a lot of control to the user. The 95% quantile of the intensities in these patches is chosen as the background intensity for this spot in this frame.
- 2: The background of each spot is estimated from 8 points located in a circle around the spot with a radius controlled by the argument 'backgroundRadius' (by default, 4*spotRadius).
- 3: The background for each frame is calculated by first locating the cell in the video and then analyzing the intensities of the cell in each frame (Fig. 36). The background is chosen as the gray value at a given quantile of this distribution (by default 0.5 (=50%), although this parameter can be controlled through the argument 'backgroundPercentile', this value can be set higher, for instance, 0.9 (=90%) if you want that most of the cell is considered as background). To help in the identification

of the cell, you may indicate which is the maximum background value expected along the frames (for instance, in our videos, the background value normally never goes beyond 6000). By default, this option ('maxBackground') is set to 0, meaning that this help is not used by default. You may see which is the cell detection and the area selected for the background estimation by setting the argument 'showImages' to 1 (you may stop the execution at any time by pressing CTRL-C).

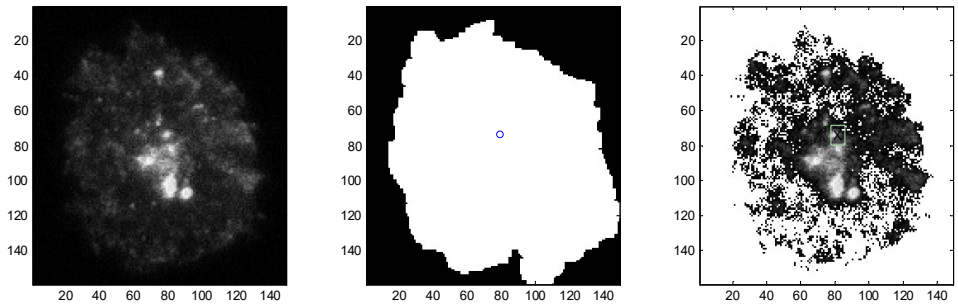

Fig. 36. Left: the frame being analyzed. Middle: the cell detected and the spot being analyzed within this frame. Right: white pixels correspond to those points that are considered to belong to the background. Only those within the cell mask are used for the estimate of the background.

27) We may gather the diffusion and intensity information of the previous section with the intensity information calculated in this section

```
gatherDiffusionAndIntensity()
```

28) Finally, we may also gather the diffusion and moment scaling spectrum information with the intensity. For doing so, we need the suffixes used in Steps 24 and 25. For instance, for gathering the diffusion and intensity information we may do

```
gatherDiffusionAndIntensity('Short')
```

where 'Short' was the suffix used in Step 25. This command will create two files one called `diffusionCoefficientsAndIntensitiesShort.txt` and another `log_diffusionCoefficientsAndIntensitiesShort.txt` in the directory `results/TrackingPackage/tracks`. The first one contains 1) the spot index, 2) the diffusion coefficient, 3) the intensity, and 4) the region number within the mask. This file can be read from Excel. The log file contains this information in a more human readable way.

To gather the Moment Spectrum Scaling and the intensity information you may use

```
gatherTrajectoryClassificationAndIntensity('Long')
```

where 'Long' was the suffix used in Step 25. This command will create two files one called `trajectoryClassificationAndIntensitiesLong.txt` and another `log_ trajectoryClassificationAndIntensitiesLong.txt` in the directory `results/TrackingPackage/tracks`. The first one contains 1) the spot index, 2) the movement type (see Step 25), 3) the spot first moment (see Step 25), 4) the intensity, and 5) $D_{1\text{-}4}$ and 6) the region number within the mask. This file can be read from Excel. The log file contains this information in a more human readable way.

Conclusions

The quantitative analysis of the diffusion coefficients and cluster size by fluorescence microscopy provides a valuable tool to objectively determine the effect of the treatments applied and manifested at the level of membrane receptors. In this article we have presented detailed protocols for the analysis of these features. The method described here not only allows single-molecule tracking detection, but also automates estimation of lateral diffusion parameters at the cell membrane, classifies the type of trajectory and allows complete analysis thus overcoming the difficulties in quantifying spot size over its entire trajectory at the cell membrane. The method is publicly available at http://i2pc.es/coss/Programs/protocolScripts.zip.

Acknowledgements

We are thankful to Carlo Manzo and Maria García Parajo for their help and source code of the diffusion coefficient analysis.

This work was supported in part by grants from the Spanish Ministry of Economy and Competitiveness (SAF 2014-53416-R) and the RETICS Program of the Instituto de salud Carlos III (RD12/0009/009 and RD16/0012/0006; RIER). LM-M and JV are supported by the COMFUTURO program of the Fundación General CSIC.

Bibliography

[Bakker2012] Bakker, G. J.; Eich, C.; Torreno-Pina, J. A.; Diez-Ahedo, R.; Perez-Samper, G.; van Zanten, T. S.; Figdor, C. G.; Cambi, A. & Garcia-Parajo, M. F. Lateral mobility of individual integrin nanoclusters orchestrates the onset for leukocyte adhesion. *Proc. Natl. Academy of Sciences of the USA*, **2012**, 109, 4869-4874

[Calebiro2013] Calebiro, D.; Rieken, F.; Wagner, J.; Sungkaworn, T.; Zabel, U.; Borzi, A.; Cocucci, E.; Zürn, A. & Lohse, M. J. Single-molecule analysis of fluorescently labeled G-protein-coupled receptors reveals complexes with distinct dynamics and organization. *Proc. Natl. Academy of Sciences of the USA*, **2013**, 110, 743-748

[Destainville2006] Destainville, N. & Salomé, L. Quantification and correction of systematic errors due to detector time-averaging in single-molecule tracking experiments. *Biophysical J*, **2006**, 90, L17-L19

[Digman2008] Digman, M. A.; Dalal, R.; Horwitz, A. F. & Gratton, E. Mapping the number of molecules and brightness in the laser scanning microscope. *Biophysical J*, **2008**, 94, 2320-2332

[Ferrari2001] Ferrari, R.; Manfroi, A. J. & Young, W. R. Strongly and weakly self-similar diffusion. *Physica D*, **2001**, 154, 111-137

[Jaqaman2008] Jaqaman, K.; Loerke, D.; Mettlen, M.; Kuwata, H.; Grinstein, S.; Schmid, S. L. & Danuser, G. Robust single-particle tracking in live-cell time-lapse sequences. *Nature methods*, **2008**, 5, 695-702

[Kusumi1993] Kusumi, A.; Sako, Y. & Yamamoto, M. Confined lateral diffusion of membrane receptors as studied by single particle tracking (nanovid microscopy). Effects of calcium-induced differentiation in cultured epithelial cells. *Biophys J*, **1993**, 65, 2021-2040

[Manzo2015] Manzo, C. & Garcia-Parajo, M. A review of progress in single particle tracking: from methods to biophysical insights. *Reports on Progress in Physics*, **2015**, 124601

[Mattheyses2010] Mattheyses, A. L.; Simon, S. M. & Rappoport, J. Z. Imaging with total internal reflection fluorescence microscopy for the cell biologist. *J Cell Science*, **2010**, 123, 3621-3628

[Sbalzarini2005] Sbalzarini, I. F. & Koumoutsakos, P. Feature point tracking and trajectory analysis for video imaging in cell biology. *J. Structural Biology*, **2005**, 151, 182-195

[Schindelin2012] Schindelin, J.; Arganda-Carreras, I.; Frise, E.; Kaynig, V.; Longair, M.; Pietzsch, T.; Preibisch, S.; Rueden, C.; Saalfeld, S.; Schmid, B. Fiji: an open-source platform for biological-image analysis. *Nature methods*, **2012**, 9, 676-682

[Schuetz1997] Schütz, G. J.; Schindler, H. & Schmidt, T. Single-molecule microscopy on model membranes reveals anomalous diffusion. *Biophysical journal*, **1997**, 73, 1073-1080

[Yu2016] Yu, J. Single-Molecule Studies in Live Cells. *Annual review of physical chemistry*, **2016**, 67, 565-585