Quantitative Imaging of Cytoskeletal Topology

Alberto Scarampi del Cairo

1 Introduction

Cytoskeletal filaments of tubulin form protein networks that control spatial organization in cells. Tubulin is also an important factor controlling cell replication, and as such its dynamical organization is often perturbed in cancer cells. For this reason, the ability to obtain quantifiable data about the topology of microtubules in cancer cells could allow quantification of their dependency on different stresses, drugs, mechanical stimuli, etc. (Li et al. 2012). This framework recognise cell geometry and cytoskeletal filament patterns, and as such could be useful in high-throughput screening and early detection of cancerous cells. In particular, it would be interesting to experimentally characterise cytoskeletal-nuclear surface ratios in healthy and cancerous cells and use this script to quantify differences.

2 Methods

The script detects edges in fluorescence tubulin images anf their corresponding DNA DAPI images. Edges are detected using the edge function, using ta discrete differentiation operator (sobel parameter) . The two edged images are then superimposed in a single image. The true values of the logical matrices (where edge are detected) are then plotted on a 2-D coordinate system using the find function. This conversion into a coordinate system allows to easily analyse cell geometry, for example by using the border function, which calculates the coordinate of the points at the borders of the image and measures the area circumscribed by that border. Borders and areas are calculated for both protein and DNA signals. This allows to quantify the ratio between cytosolic area and nuclear area. The area in pixels is then converted into micrometer squared, using the pixel to micrometer conversion reported in the original paper (Roques & Murphy 2002).

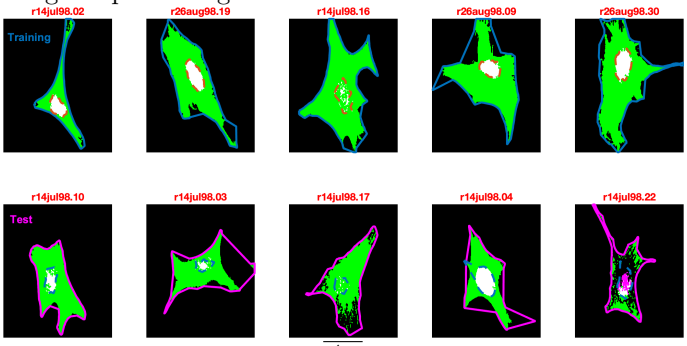
3 Results

Image	Cell Area (μm^2)	Ap/An
r14jul98.02	180.37 ± 32.55	4.85 ± 0.91
r26aug98.19	462.30 ± 32.55	9.14 ± 0.91
r14jul98.16	546.19 ± 32.55	10.84 ± 0.91
r26aug98.09	551.27 ± 32.55	14.06 ± 0.91
r26aug98.30	569.84 ± 32.55	2.78 ± 0.91
r14jul98.10	316.74 ± 11	8.71 ± 1.18
r14jul98.03	401.70 ± 11	18.70 ± 1.18
r14jul98.03	353.35 ± 11	13.86 ± 1.18
r14jul98.17	364.43 ± 11	4.31 ± 1.18
r14jul98.04	256.30 ± 11	6.04 ± 1.18

For each image, the ratio between the protein and DNA signals $\frac{A_P}{A_N}$ and whole cell surface $\overline{A_T}$ are calculated. Results are reported in the table above, with standard errors ($\sigma_x = \frac{\sigma}{\sqrt{n}}$). For the training images (first 5 rows) the calculated average area delimited by outer border, $\overline{A_T} = 461.99\mu m^2$. The average ratio of protein to nuclear area $\overline{\frac{A_P}{A_N}}$ ratio of 8.33. For test images the corresponding values are measured as $\overline{A_T} = 338.5\mu m^2$ and $\overline{\frac{A_P}{A_N}} = 10.32$.

4 Discussion

As shown in the figure below, the script can accurately trace the cytoplasmic borders (blue for training and magenta for test set) and the nuclear edges (red dashed line for training and light blue for test set) onto the segmented images representing the circumscribed surfaces.



The training set shows an $\overline{\frac{A_P}{A_N}}$ ratio of 8.33, that means that the cytosolic area is 8.33 bigger than the nuclear. This is equivalent to a 12.5 % nuclear occupancy relative to the tubulin surface. Both this protein to DNA ratio and the computed average cell area $\overline{A_T} = 461.99\mu m^2$ are in accordance to literature data (Su Lim et al. 2015). The same features calculated for the test data have similar values of $\overline{A_T} = 338.5\mu m^2$ and $\overline{\frac{A_P}{A_N}} = 10.32$ but larger errors. Indeed, as observed in some segmented images above (fig. 2 and 5 of training set), sometimes the borders do not overlap with the segmentation, accounting for the reported errors. The algorithm could be improved by improving the noise filtering step, which is currently achieved by a simple mask with cutoff value n = 12. In addition, it would be interesting to implement a function that calculates the average length of tubulin filaments from the edged images.

References

Li, J., Shariff, A., Wiking, M., Lundberg, E., Rohde, G. K. & Murphy, R. F. (2012), 'Estimating microtubule distributions from 2d immunofluorescence microscopy images reveals differences among human cultured cell lines', *PLOS ONE* **7**(11), 1–11.

Roques, E. J. S. & Murphy, R. F. (2002), 'Objective evaluation of differences in protein subcellular distribution', *Traffic* **3**(1), 61–65.

Su Lim, C., Sun Kim, E., Yeon Kim, J., Taek Hong, S., Jai Chun, H., Eun Kang, D. & Rae Cho, B. (2015), 'Measurement of the nucleus area and nucleus/cytoplasm and mitochondria/nucleus ratios in human colon tissues by dual-colour two-photon microscopy imaging', *Scientific Reports* **5**.

```
function cytoskimager()
    %% DATA Inputs
    % load Protein images
    P1 = imread ('r14jul98.tubul.02——1———2.tif');
    P2 = imread ('r26aug98.tubul.19——1———2.tif');
    P3 = imread ('r14jul98.tubul.16——1———2.tif');
    P4 = imread ('r26aug98.tubul.09——1———2.tif');
    P5 = imread ('r26aug98.tubul.30——1———2.tif');
    % load DNA images
    D1 = imread ('r14jul98.tudap.02——1———2.tif');
    D2 = imread ('r26aug98.tudap.19——1———2.tif');
    D3 = imread ('r14jul98.tudap.16——1———2.tif');
    D4 = imread ('r26aug98.tudap.09——1———2.tif');
    D5 = imread ('r26aug98.tudap.30——1———2.tif');
    % noise filter by cutoff value n
    n = 12;
    Pn1 = P1 > n; % protein
    Pn2 = P2 > n;
    Pn3 = P3 > n;
    Pn4 = P4 > n;
    Pn5 = P5 > n;
    Dn1 = D1 > n; % dna
    Dn2 = D2 > n;
    Dn3 = D3 > n;
    Dn4 = D4 > n;
    Dn5 = D5 > n;
    %% EDGE DETECTION
    % calculate the threshold for edge detection in images
    [¬, thresholdP1] = edge(Pn1, 'sobel'); % protein
    [¬, thresholdP2] = edge(Pn2, 'sobel');
    [¬, thresholdP3] = edge(Pn3, 'sobel');
    [¬, thresholdP4] = edge(Pn4, 'sobel');
    [¬, thresholdP5] = edge(Pn5, 'sobel');
    [¬, thresholdD1] = edge(Dn1, 'sobel'); % DNA
    [¬, thresholdD2] = edge(Dn2, 'sobel');
    [¬, thresholdD3] = edge(Dn3, 'sobel');
    [¬, thresholdD4] = edge(Dn4, 'sobel');
    [¬, thresholdD5] = edge(Dn5, 'sobel');
    % define resolution of edge detection
    resolution = .7;
    % find edges (using threshold and resolution)
    EP1 = edge(Pn1,'sobel', thresholdP1 * resolution);%protein
    EP2 = edge(Pn2,'sobel', thresholdP2 * resolution);
    EP3 = edge(Pn3,'sobel', thresholdP3 * resolution);
    EP4 = edge(Pn4,'sobel', thresholdP4 * resolution);
    EP5 = edge(Pn5,'sobel', thresholdP5 * resolution);
    ED1 = edge(Dn1,'sobel', thresholdD1 * resolution);%DNA
    ED2 = edge(Dn2,'sobel', thresholdD2 * resolution);
    ED3 = edge(Dn3,'sobel', thresholdD3 * resolution);
    ED4 = edge(Dn4,'sobel', thresholdD4 * resolution);
    ED5 = edge(Dn5,'sobel', thresholdD5 * resolution);
    %% SEGMENT IMAGES
    se90 = strel('line', 3, 90); %strel finds true pixels with 'line' geometry in neighbors of 3
    se0 = strel('line', 3, 0); %perpendicular direction % dilate images based on strel (3 pixels around)
    EP1dil = imdilate(EP1, [se90 se0]); %protein
    EP2dil = imdilate(EP2, [se90 se0]);
    EP3dil = imdilate(EP3, [se90 se0]);
    EP4dil = imdilate(EP4, [se90 se0]);
    EP5dil = imdilate(EP5, [se90 se0]);
    ED1dil = imdilate(ED1, [se90 se0]); %DNA
    ED2dil = imdilate(ED2, [se90 se0]);
    ED3dil = imdilate(ED3, [se90 se0]);
    ED4dil = imdilate(ED4, [se90 se0]);
    ED5dil = imdilate(ED5, [se90 se0]);
    % fill holes
    EP1fill = imfill(EP1dil, 'holes'); %protein
    EP2fill = imfill(EP2dil, 'holes');
    EP3fill = imfill(EP3dil, 'holes');
    EP4fill = imfill(EP4dil, 'holes');
    EP5fill = imfill(EP5dil, 'holes');
    ED1fill = imfill(ED1dil, 'holes'); %DNA
    ED2fill = imfill(ED2dil, 'holes');
    ED3fill = imfill(ED3dil, 'holes');
    ED4fill = imfill(ED4dil, 'holes');
    ED5fill = imfill(ED5dil, 'holes');
    % polish borders
    seD = strel('diamond',1);
    EP1final = imerode(EP1fill,seD); % polish based on strel diamond (it's smoother)
    EP2final = imerode(EP2fill,seD);
    EP3final = imerode(EP3fill,seD);
    EP4final = imerode(EP4fill,seD);
    EP5final = imerode(EP5fill,seD);
    ED1final = imerode(ED1fill,seD);
    ED2final = imerode(ED2fill,seD);
```

```
    ED3final = imerode(ED3fill,seD);
    ED4final = imerode(ED4fill,seD);
    ED5final = imerode(ED5fill,seD);
    %% Fuse Segmented Protein and Nuclear Areas
    F1final = imfuse(EP1final, ED1final);
    F2final = imfuse(EP2final, ED2final);
    F3final = imfuse(EP3final, ED3final);
    F4final = imfuse(EP4final, ED4final);
    F5final = imfuse(EP5final, ED5final);
    %% Images to Coordinate System
    % find coordinates of true values (where edge is detected)
    [rowP1,colP1] = find(EP1final==1); % protein
    [rowP2,colP2] = find(EP2final==1);
    [rowP3,colP3] = find(EP3final==1);
    [rowP4,colP4] = find(EP4final==1);
    [rowP5,colP5] = find(EP5final==1);
    [rowD1,colD1] = find(ED1final==1); % DNA
    [rowD2,colD2] = find(ED2final==1);
    [rowD3,colD3] = find(ED3final==1);
    [rowD4,colD4] = find(ED4final==1);
    [rowD5,colD5] = find(ED5final==1);
    %% determine borders of the cell (k) and measure Area (A) delimited by k borders
    [kP1, AT1] = boundary(colP1,rowP1);  % k is border and AT area delimited by k border (tot cell area)
    [kP2, AT2] = boundary(colP2,rowP2);
    [kP3, AT3] = boundary(colP3,rowP3);
    [kP4, AT4] = boundary(colP4,rowP4);
    [kP5, AT5] = boundary(colP5,rowP5);
    [kD1, AD1] = boundary(colD1,rowD1); %AD is Area of nucleus
    [kD2, AD2] = boundary(colD2,rowD2);
    [kD3, AD3] = boundary(colD3,rowD3);
    [kD4, AD4] = boundary(colD4,rowD4);
    [kD5, AD5] = boundary(colD5,rowD5);
    %% Graphical Outputs
    % plot cell borders (superimpose borders onto segmented image)
    figure
    subplot(1,5,1)
    imshow(F1final); hold on % hold on until borders plotted
    plot(colP1(kP1), rowP1(kP1),'b', 'LineWidth', 2)
    plot(colD1(kD1), rowD1(kD1), '—', 'LineWidth', 2)
    title('r14jul98.02');grid on; hold off
    subplot(1,5,2)
    imshow(F2final); hold on
    plot(colP2(kP2), rowP2(kP2), 'LineWidth', 2)
    plot(colD2(kD2), rowD2(kD2), '—', 'LineWidth', 2)
    title('r26aug98.19');grid on; hold off
    subplot(1,5,3)
    imshow(F3final); hold on
    plot(colP3(kP3), rowP3(kP3), 'LineWidth', 2)
    plot(colD3(kD3), rowD3(kD3), '—', 'LineWidth', 2)
    title('r14jul98.16');grid on; hold off
    subplot(1,5,4)
    imshow(F4final); hold on
    plot(colP4(kP4), rowP4(kP4), 'LineWidth', 2)
    plot(colD4(kD4), rowD4(kD4), '—', 'LineWidth', 2)
    title('r26aug98.09');grid on; hold off
    subplot(1,5,5)
    imshow(F5final); hold on
    plot(colP5(kP5), rowP5(kP5), 'LineWidth', 2)
    plot(colD5(kD5), rowD5(kD5), '—', 'LineWidth', 2)
    title('r26aug98.30');grid on; hold off
    %% Numerical Outputs
    % cytosolic area (= total area  — nuclear)
    AP1 = AT1 — AD1; AP2 = AT2 — AD2; AP3 = AT3 — AD3; AP4 = AT4 — AD4; AP5 = AT5 — AD5;
    % ratio protein/nuclear area
    RPD = [];
    RPD(1,1) = AP1/AD1;
    RPD(2,1) = AP2/AD2;
    RPD(3,1) = AP3/AD3;
    RPD(4,1) = AP4/AD4;
    RPD(5,1) = AP1/AD5;
    RPDavg = mean(RPD); RPDsterr = std(RPD)/5; RPD  %output ratios
    % convert area from pixels to micrometers
    conversion = 98^2; % 1 pixel = 98 nm (in length) —> 1 pixel squared = 98x98 nm squared
    Aum = [];
    Aum(1,1) = conversion * AT1 * 10^—6;
    Aum(2,1) = conversion * AT2 * 10^—6;
    Aum(3,1) = conversion * AT3 * 10^—6;
    Aum(4,1) = conversion * AT4 * 10^—6;
    Aum(5,1) = conversion * AT5 * 10^—6;
    Aumavg = mean(Aum); Aumsterr = std(Aum)/5; Aum % output the ratios
end
```