Electronic Supporting Information: Motility efficiency of non-metastatic vs. metastatic breast cancer cells

Thomas M. Hermans[a], Didzis Pilans[a], Sabil Huda[a], Patrick Fuller[a], Kristiana Kandere-Grzybowska[a], Bartosz A. Grzybowski[a,*]

[a] Department of Chemical and Biological Engineering & Department of Chemistry,

Northwestern University, 2145 Sheridan Rd., Evanston, IL 60208, USA.

* Correspondence to gryzbor@northwestern.edu

Section S1: Captions for supporting movies M1–M6

Movie M1. Protrusion-Retraction analysis of a MDA-MB-231 cell. The area of protrusion (A_{pro}, shaded red) and area of retraction (A_{ret}, shaded blue) are overlaid with differential interference contrast images. The analysis of this movie (i.e., the sum of all red areas at each time, and the sum of all blue area at each time) is shown in Fig. 2A in the main text.

Movie M2. Protrusion-Retraction analysis of a MCF-7 cell. The area of protrusion (A_{pro}, shaded red) and area of retraction (A_{ret}, shaded blue) are overlaid with differential interference contrast images. The analysis of this movie (i.e., the sum of all red areas at each time, and the sum of all blue area at each time) is shown in Fig. 2B in the main text.

Movie M3. Morphodynamic analysis of a MDA-MB-231 cell (same cell as in Movie M1). The Kass snake (see main text) is represented with red (i.e., protrusion; positive boundary speed) or blue (i.e., retraction; negative boundary speed) circles. Four white circles show specific values of the normalized boundary length L, where (from large to small) the white circles represent $L = 0$, $L = 0.25$, $L = 0.5$ and $L = 0.75$. The boundary speed at each value of L at time t is shown in Fig. 5A in the main text. The yellow circle shows the cell centroid and the yellow dashed line shows the trajectory of the centroid over a 2 hour time period. The morphodynamic analysis, performed on confocal laser scanning microscopy images, is overlaid with differential interference contrast images.

Movie M4. Morphodynamic analysis of a MCF-7 cell (same cell as in Movie M2). The Kass snake (see main text) is represented with red (i.e., protrusion; positive boundary speed) or blue (i.e. retraction; negative boundary speed) circles. Four white circles show specific values of the normalized boundary length L, where (from large to small) the white circles represent $L = 0$, $L = 0.25$, $L = 0.5$ and $L = 0.75$. The boundary speed at each value of L at time t is shown in Fig. 5B in the main text. The yellow circle shows the cell centroid and the yellow dashed line shows the trajectory of the centroid over a 2 hour time period. The morphodynamic analysis, performed on confocal laser scanning microscopy images, is overlaid with differential interference contrast images.

Movie M5. Computer simulation and analysis of cell-like object translocating and deforming simultaneously. Left panel: A black circle was simulated to move in a two-dimensional plane (white background) using the Processing language (prossesing.org). Two sequential simulation steps are shown simultaneously to obtain a particle consisting of two circles. Right panel: Protrusion area (A_{pro}, shaded red) and retraction area (A_{ret}, shaded blue) for the object shown in the left panel. The efficiency of motility of the simulated object is described in SI section S2 and Fig. S1.

Movie M6. Fascin-GFP expression in MDA-MB-231 cell. A MDA-MB-231 cell was transfected using fascin-GFP and stained with cytosolic dye Celltracker Orange. Top left panel: Overlay of fascin-GFP and Celltracker orange signal. Top right panel: fascin-GFP channel (excitation wavelength 488 nm). Bottom left panel: Celltracker Orange channel (excitation wavelength 543 nm). Bottom right panel: ratio of Celltracker Orange channel and fascin-GFP channel. Fascin-GFP is observed in the fibrous structures at the rear of the cell. Celltracker Orange, which is a cytosolic dye, is not observed in these fibrous structures. This indicates that fascin-GFP is incorporated specifically in the structures and the signal is not due to non-specific cytosolic fascin-GFP.

Section S2: Definition of the motility efficiency

This section explains in more detail the definition of motility efficiency outlined in the main text. In the experiments (see main text) digital micrographs of motile cells are taken at regular time intervals. Area onto which the cell moves from one interval to another is a protrusion (with area A_{pro}), and area from which the cell retracts is a retraction (with area A_{ret}), as depicted in Fig. S1.

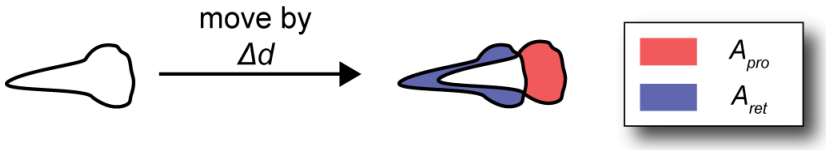

Fig. S1. Illustration of area changes during cell motility

As explained in the main text, we assume that all forces during protrusion as well as retraction activity are commensurate, and that therefore the work performed by the cell scales linearly with the protruded/retracted area. We wish to estimate what fraction of the total area changed, $\Delta A_{exp} = |A_{pro}| + |A_{ret}|$, contributes to actual translocation of the cell and what fraction is used unproductively for other purposes (e.g., membrane ruffling). This is done by comparing ΔA_{exp} to the "theoretical area" ΔA_{theo} , which is the mean area that would be changed (protruded and retracted) if a cell having the same area and circularity moved its centroid without shape deformation over the same distance Δd. With this comparison, we define motility efficiency as:

$$\eta = \frac{\Delta A_{theo}}{\Delta A_{\exp}} \cdot 100\% .$$

The definition of ΔA_{theo} itself needs more explanation. Let us first consider the translocation of a perfect circle (without any deformation) illustrated in Fig. S2 below:

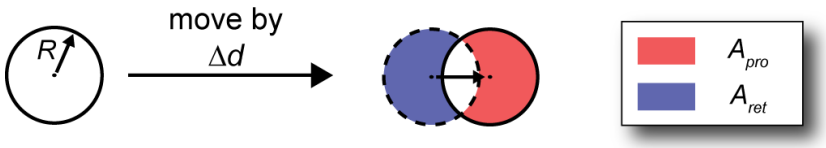

Fig. S2. Area change during translation of a circle.

In this case, total area changed $\Delta A_{theo,circ}$ can be expressed analytically and does not depend on the direction of travel. After some simple trigonometry, we have:

$$A_{theo,circ} = 2\left(\pi R^2 - \left(2R^2 \cos^{-1}\sqrt{\frac{\Delta d}{2R} - \frac{1}{2}\left((-\Delta d + 2R)(\Delta d)^2(\Delta d + R)\right)}\right)\right)$$

For non-circular shapes, calculation of ΔA_{theo} is much more complicated and the specific values depend also on the direction of travel (e.g., moving a square's centroid by Δd will give different values of ΔA_{theo} depending on whether the direction of motion is parallel to the square's side or to the square's diagonal).

Accordingly, we performed numerical simulations in which we calculated the values of ΔA_{theo} for different shapes (Fig. S3) moving – again, without deformation – along different directions but over the same distance Δd. For each of these shapes, we also calculated the circularity defined as $c = 4\pi A/p^2$, where A stands for area and p is the circumference.

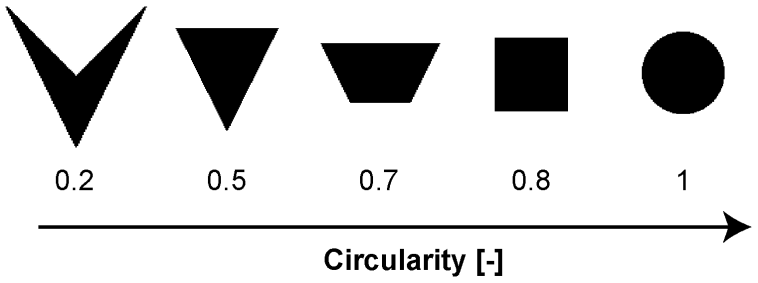

Fig. S3. Circularity changes in five different objects having the same area. See section S5 for simulation code.

The distributions of ΔA_{theo} for each object were then plotted against circularity, c. In Fig. S4, the values of ΔA_{theo} are normalized by $\Delta A_{theo,\ circ}$ – importantly, the mean values of ΔA_{theo} scale linearly with c. This proportionality is used in the definition of motility efficiency in the main text – namely, for a given instantaneous cell shape, we calculate its circularity and, based on the plot below, estimate the expected mean value of ΔA_{theo}. This value is then compared to the experimental area change ΔA_{exp}. This procedure allows us to compare the experimentally observed area changes to the area changes that would be expected if a cell of *similar shape* (i.e., same circularity) moved by pure translation with no deformation.

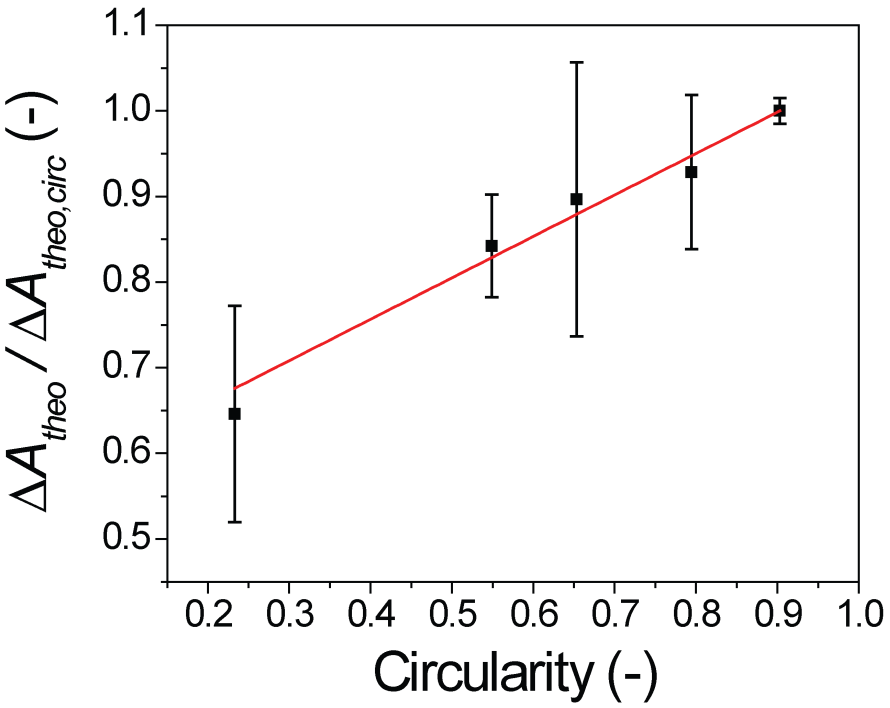

Fig. S4. Circularity vs. area change. Markers are the average values, error bars correspond to standard deviations based on different directions of motion.

We make one more comment about the efficiency of objects that are both translating and changing shape. As narrated in the main text, depending on the detailed nature of cell deformation, the instantaneous values of η can be either lower or higher than 100%. Fig. S5 below gives some examples of actual cell motions/deformations for which instantaneous η can peak above 100%. On the other hand, over many time intervals, these kinds of events average out to < 100% as illustrated in Fig. S5E.

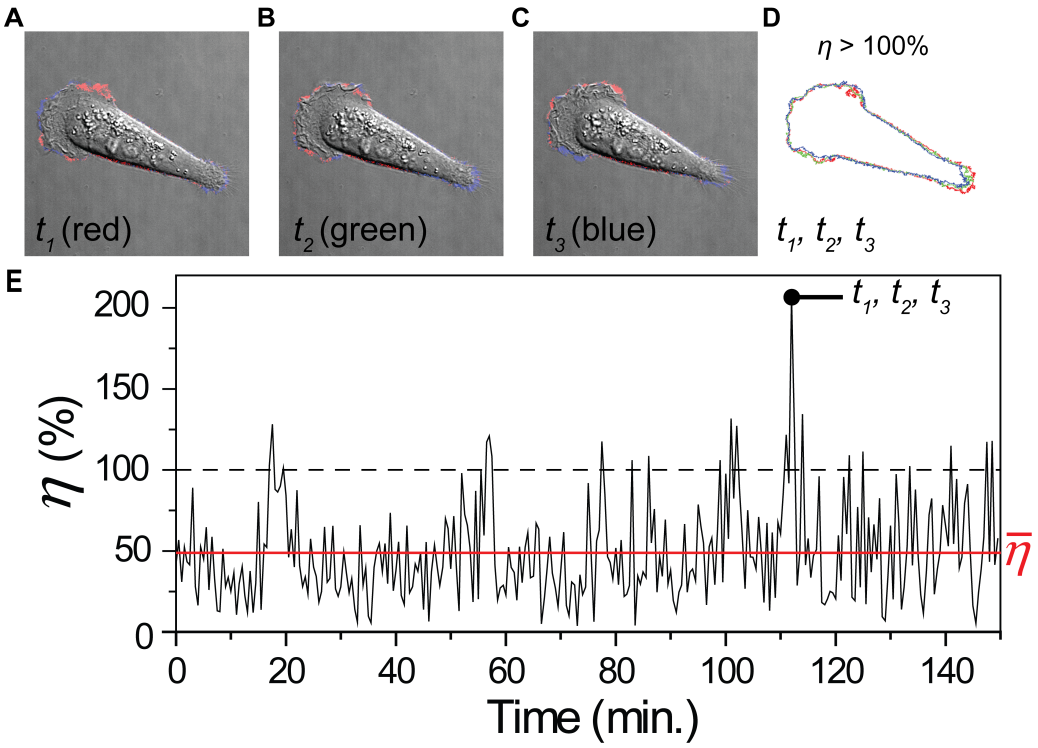

Fig. S5. Fluctuations in instantaneous values of efficiency can exceed 100%. (A–C) Experimental images of MDA-MB-231 cell (protrusions shaded red, retractions shaded blue) at times t_1 = 111.5, t_2 = 112 and t_3 = 112.5 min. (D) Outlines of cell at times t_1 (red), t_2 (green) and t_3 (blue), show that the cell's rear (bottom right end) is retracting. This particular shape change of the cell results in instantaneous values of η > 100. (E) Motility efficiency of the cell shown in panels A–D over a longer time interval. At several instances, the instantaneous values of η exceed 100% (i.e., above the dashed horizontal line). Still, the average efficiency (horizontal red line) is well below 100%.

Note: the method described here might appear unwieldy at first glance. An alternative method to correct for the circularity might consist of copying the shape of the cell at time t and translocate it along the centroid trajectory (t → t+1), and in this way calculate ΔA_{theo}. After evaluating a range of simulations (one of which is shown in Movie M5) the method described in the main text (and above) was found to produce smaller fluctuations in the instantaneous value of efficiency, and therefore lead to smaller standard deviations allowing for a better discrimination between MDA-MB-231 and MCF-7 cells.

Section S3: additional Figures

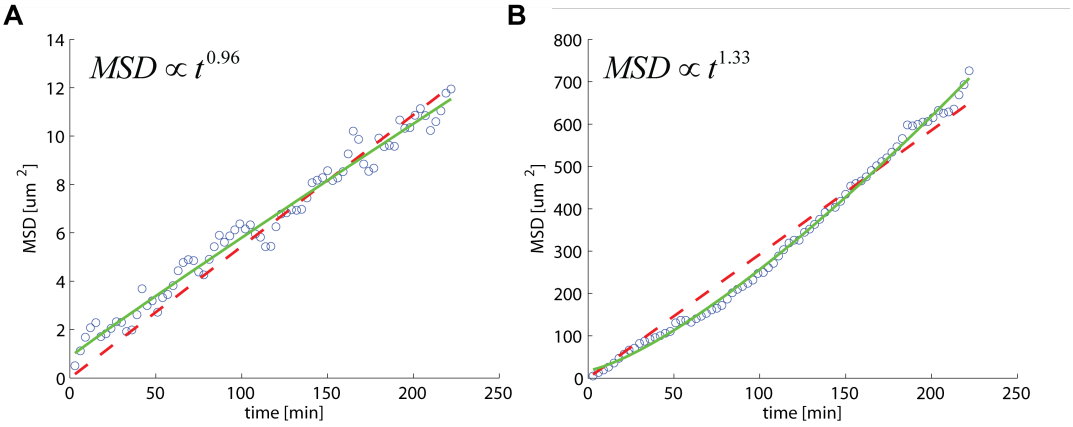

Fig. S6. Mean squared displacement (MSD) of the cell centroid versus time (using 57 cells for each of the cell lines at 3 min. time intervals) to determine diffusivity. (A) MCF-7 cells show a linear scaling with a fitted exponent (green solid line) of the power law of 0.97 ± 0.05 (Brownian diffusion = 1.00). (B) MDA-MB-231 cells show a non-linear relation with an exponent of 1.33 ± 0.02, indicating super-diffusive behavior. The red dashed line in both panel shows a linear fit (i.e., Brownian diffusion) as a reference.

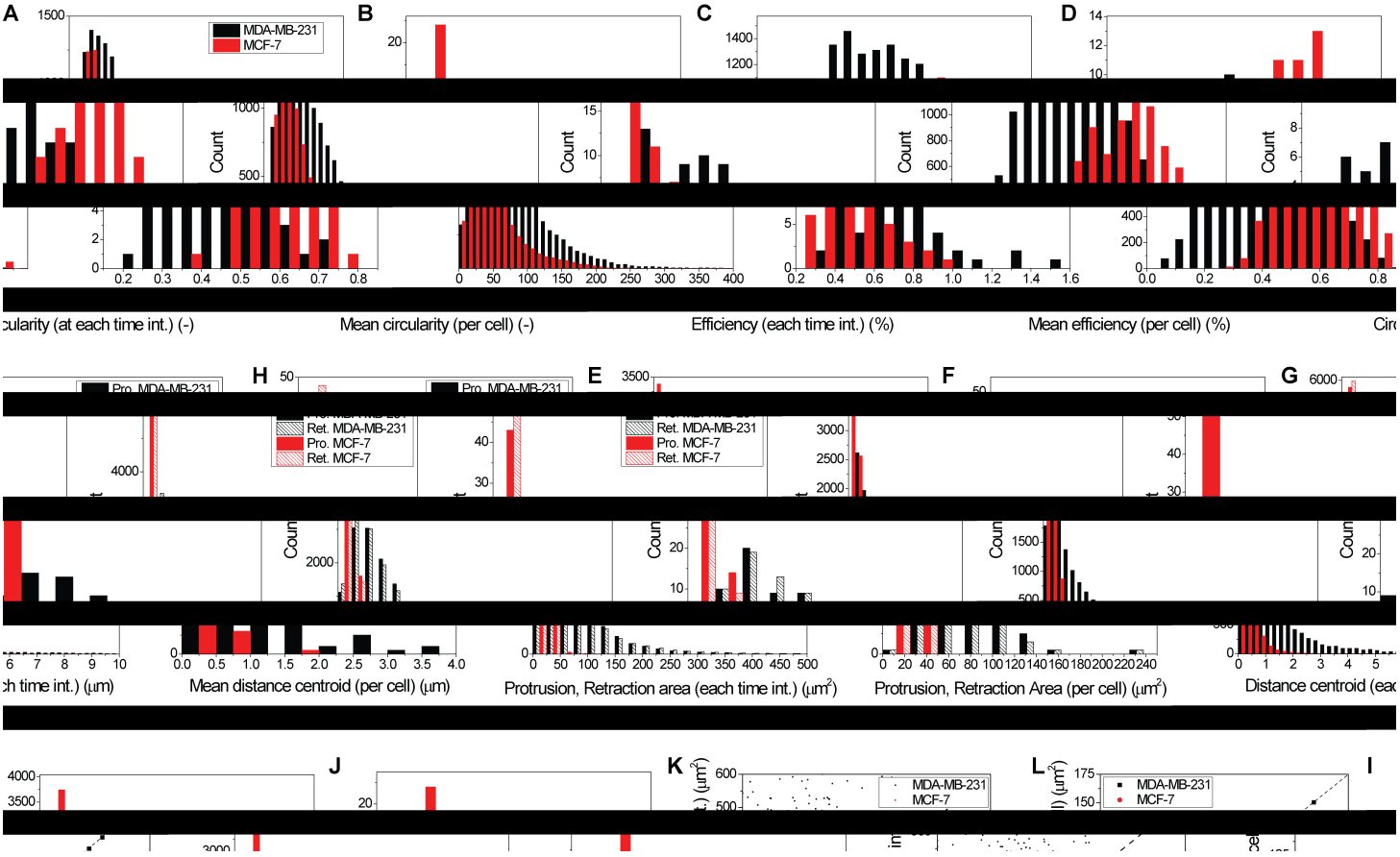

Fig. S7. Histograms and scatter plots of motility efficiency (A, B), circularity (C, D), distance traveled by the centroid (E, F), protrusion and retraction area (G, H), total cell area (I, J), and protrusion versus retraction area

(K, L). Plots in each pair have (i) data for all time intervals and (ii) mean values averaged over all time intervals per each individual cell. Note: MDA-MB-231 and MCF-7 cells keep their protrusions and retractions balanced, i.e. they protrude and retract equal areas on a per cell basis and are therefore *on average* not spreading or shrinking, explaining why a line with slope unity results when plotting $\overline{A_{pro}}$ versus $\overline{A_{ret}}$ for each cell, see Fig. S6L.

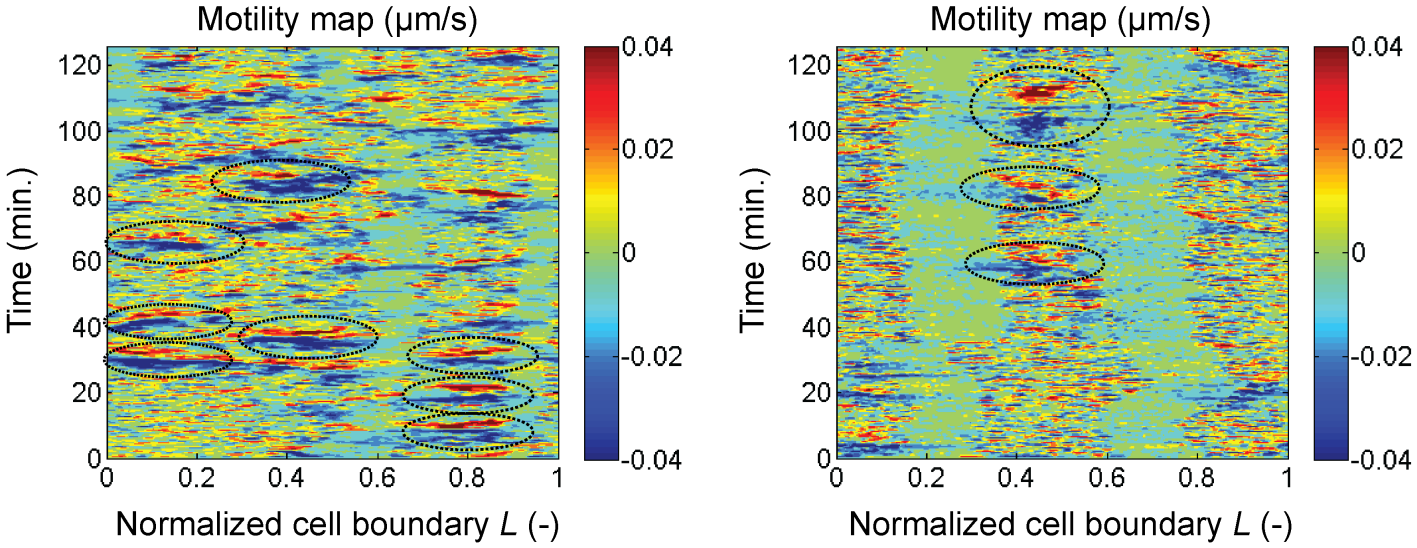

Fig. S8. Additional boundary speed maps for MDA-MB-231 cells measured at 30s intervals. Only two hours of data is shown for clarity. Retraction–protrusion pairs are indicated with dotted ellipses.

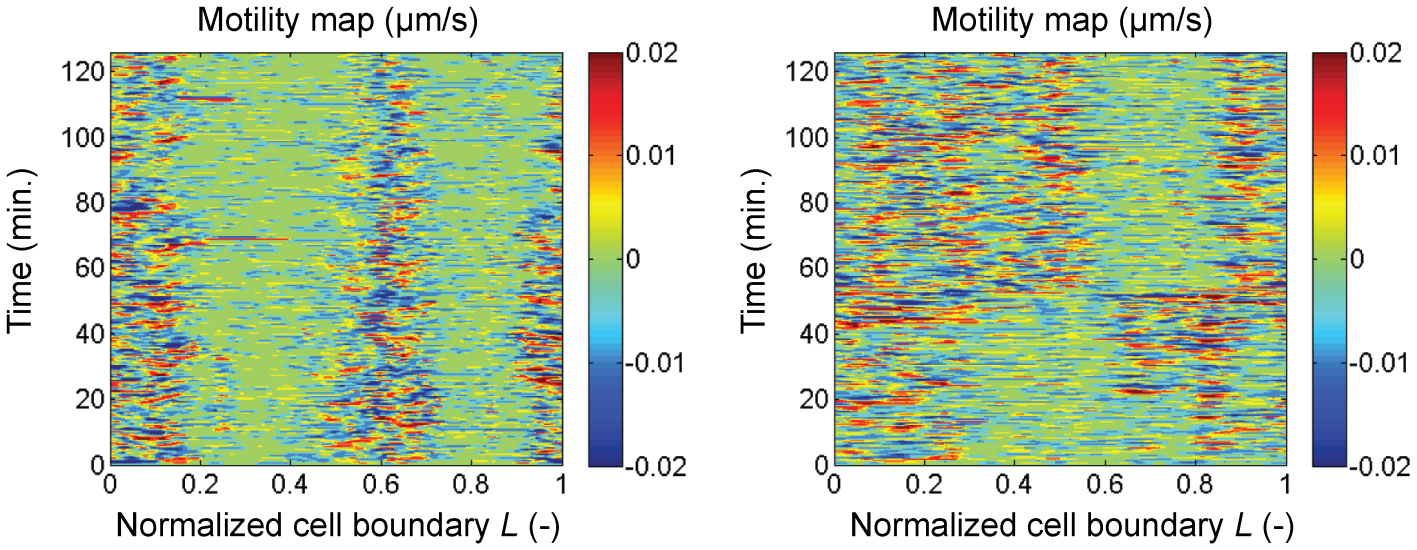

Fig. S9. Additional boundary speed maps for MCF-7 cells measured at 30s intervals. Only two hours of data is shown for clarity. No large retraction-protrusion pairs were observed for MCF-7 cells.

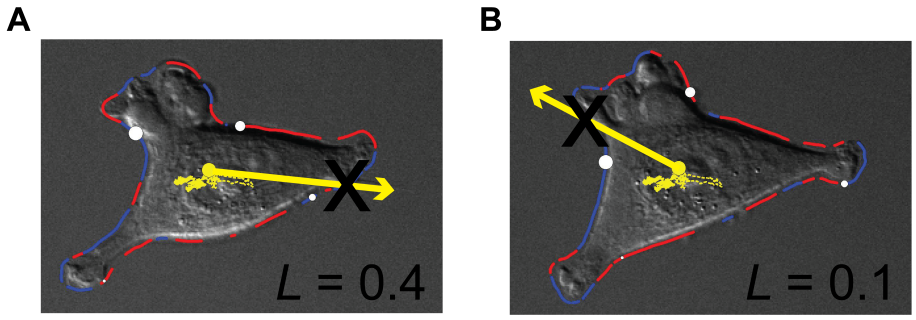

Fig. S10. Schematic representation of vector originating at the cell centroid and pointing along the direction of net centroid displacement (yellow arrow). The vector intersects the cell boundary at a certain location marked with a black cross. (A) Cell shown in Fig. 5B (main text) at time $t = 0$ min. The vector (yellow arrow) intersects the boundary at $L = 0.4$, which is indicated in Fig. 5B with the dashed black arrow; (B) the same cell at $t = 70$ min with an intersection at $L = 0.1$. Short-lived changes in the direction of the centroid movement were discarded for clarity. The dashed arrows in Fig. 5 of the main text therefore indicate the overall direction of centroid movement.

Section S4: Protein–protein interactions leading to protrusions and retractions.

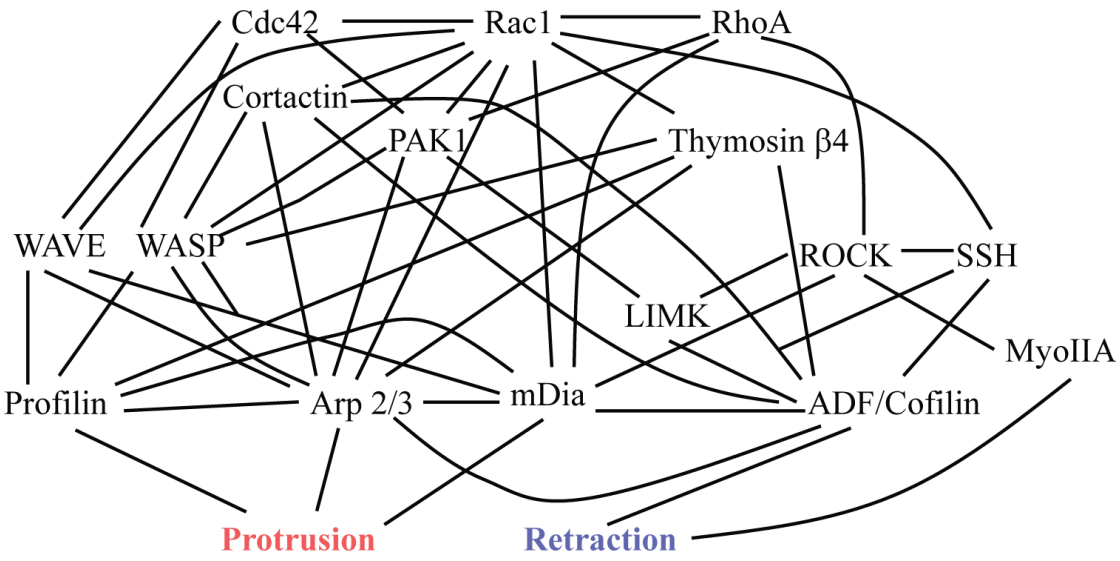

Fig. S11. Schematic representation of a network of protein-protein interactions involved in cell protrusion and retraction activity. The network was hand-curated from available literature – its construction and analysis will be described in a separate communication.

Section S5: Python 1.5.1 code for simulation of translocating shapes in Fig. S3

```
from PIL import Image, ImageDraw
from numpy.random import uniform, normal, pareto
from numpy import cumsum, savetxt, array, absolute
from math import pi, sin, cos

# Distribution parameters
distribution = "Gaussian"   # Can be "Pareto", "Gaussian", or "Ballistic"
alpha = 1.5                 # Alpha of Pareto distribution
st_dev = 12                 # Sigma of Gaussian distribution

# Shape parameters
shape = "Circle" # Can be "Circle", "Square", "Triangle", "Trapezoid", "Arrow"
r = 200

# Iteration parameters
timesteps_per_image = 1
number_of_images = 500
image_size = 4096
path_image = "path.png"

# Starting point and size of circle
start = (image_size/2, image_size/2)

# First, run actual simulation using floating points
if distribution == "Gaussian":
    random = normal(scale=st_dev, size=(number_of_images, 2))
elif distribution == "Ballistic":
    mag = absolute(normal(scale=st_dev, size=(number_of_images, 1)))
    rot = uniform(high=2*pi)
    random = array([[d*cos(rot), d*sin(rot)] for d in mag])
elif distribution == "Pareto":
    mag = pareto(alpha, size=(number_of_images, 1))
    rot = uniform(high=2*pi, size=(number_of_images, 1))
    random = array([[d*cos(a), d*sin(a)] for d, a in zip(mag, rot)])
cumulative = cumsum(random, dtype=float, axis=0)

# Save text file
savetxt("output.csv", cumulative, delimiter=",")

# Draw an image of the path on top of old image
try:
    image = Image.open(path_image)
    print("Appending old path image...")
except:
    print("Creating new path image...")
    image = Image.new('RGBA', (image_size, image_size), (255,)*4)
draw = ImageDraw.Draw(image)

# Drawing path as lines
x_old, y_old = start
for i in range(1, number_of_images):
    x, y = [a+b for a,b in zip(start, cumulative[i])]
    draw.line((x_old, y_old, x, y), fill=(0,0,0,128))
    x_old, y_old = x, y
    draw.ellipse((x-5, y-5, x+5, y+5), fill=(255,0,0,128))
```

```
image.save(path_image, "PNG")

# Now, draw the images corresponding to the run
for i in range(number_of_images-timesteps_per_image+1):
    # Initialize image and print some output
    print("Iteration %d \t circle at (%d, %d)" % (i+1, x, y))
    image = Image.new('L', (image_size, image_size), 255)

    # Loop through the number of desired timesteps
    for j in range(i, i+timesteps_per_image):
        x, y = [a+b for a,b in zip(start, cumulative[j])]
        if shape == "Circle":
            ImageDraw.Draw(image).ellipse((x-r, y-r, x+r, y+r), fill=0)
        elif shape == "Square":
            ImageDraw.Draw(image).rectangle((x-r, y-r, x+r, y+r), fill=0)
        elif shape == "Triangle":
            coords = [(x-r, y-r), (x, y+r), (x+r, y-r)]
            ImageDraw.Draw(image).polygon(coords, fill=0)
        elif shape == "Trapezoid":
            coords = [(x-r, y-r), (x-r/2, y), (x+r/2, y), (x+r, y-r)]
            ImageDraw.Draw(image).polygon(coords, fill=0)
        elif shape == "Arrow":
            coords = [(x-r, y-r), (x, y+r), (x+r, y-r), (x, y)]
            ImageDraw.Draw(image).polygon(coords, fill=0)
    # Save
    image.save("img_%04d.png" % (i+1), "PNG")
```