Electronic Supplementary Information

Computer Vision for High-Throughput Analysis of Pickering Emulsions

Kieran D. Richards[a†*], *Ella Comish*[a] *and Rachel C. Evans*[a]*

[a] Department of Materials Science & Metallurgy, University of Cambridge, 27 Charles Babbage Road, CB3 0FS, U.K.

[†] Current address: Department of Chemistry, Swansea University, Singleton Park, Swansea, SA28PP, U.K.

* Corresponding author: Prof. Rachel C. Evans (rce26@cam.ac.uk); Dr Kieran D. Richards (k.d.richards@swansea.ac.uk)

TABLE OF CONTENTS

1 Synthetic images

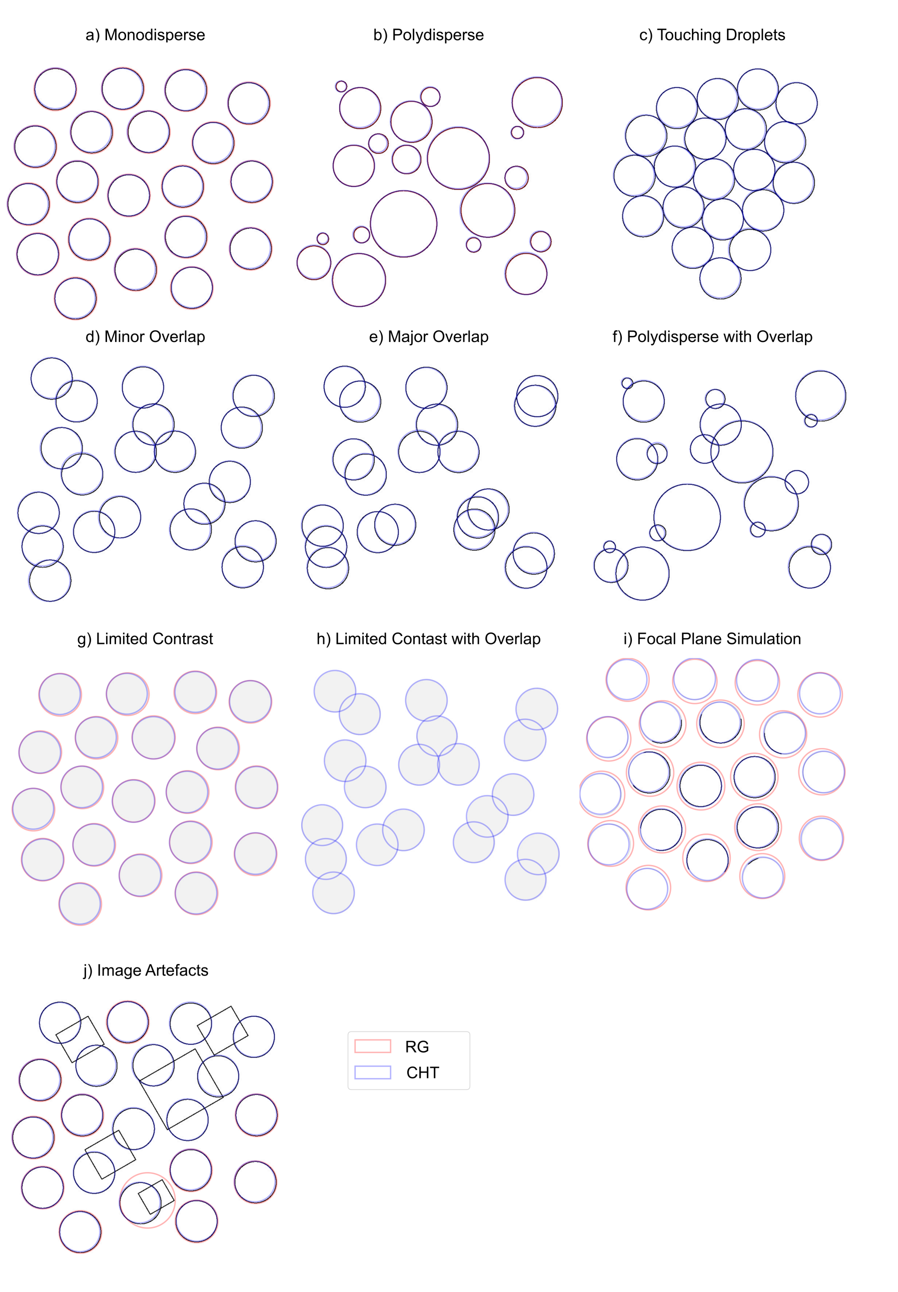

Figure S1: Synthetic images (800 × 800 pixels) representing different Pickering emulsion characteristics. (a) – (f) differ in polydispersity, overlap and spatial configuration. (g) and (h) are referred to as "limited contrast emulsions" and represent examples where emulsions do not appear as dark, well defined lines. (i) shows simulation of a focal plane where droplets appear less defined toward edges of the image and (j) shows the effect of obstructive artefacts such as clumps of particle aggregate. Red lines show the position and radius of spheres identified by RG and blue lines show the spheres identified by CHT.

Image		Precision (%)		Recall (%)		Radii Difference (Pixels)	
		CHT	RG	CHT	RG	CHT	RG
(a)	Monodisperse	100	100	100	100	-1	2
(b)	Polydisperse	100	100	100	100	0	1
(c)	Touching droplets	100	0	100	0	-1	N/A
(d)	Slight overlap	100	0	100	0	0	N/A
(e)	Severe overlap	100	0	100	0	0	N/A
(f)	Overlap polydisperse	100	0	100	0	0	N/A
(g)	Limited contrast	100	100	100	100	0	2
(h)	Overlap limited contrast	100	0	100	0	0	N/A
(i)	Focus	100	100	100	100	0	6
(j)	Monodisperse artefacts	100	90	100	50	0	3

Table S1. Performance of RG and CHT for each of the synthetically produced images. Three criteria, precision, recall and difference in the radii of the droplets between the actual and identified radii are given. The difference between the radii of the actual circles and detected circles are given to the nearest pixel.

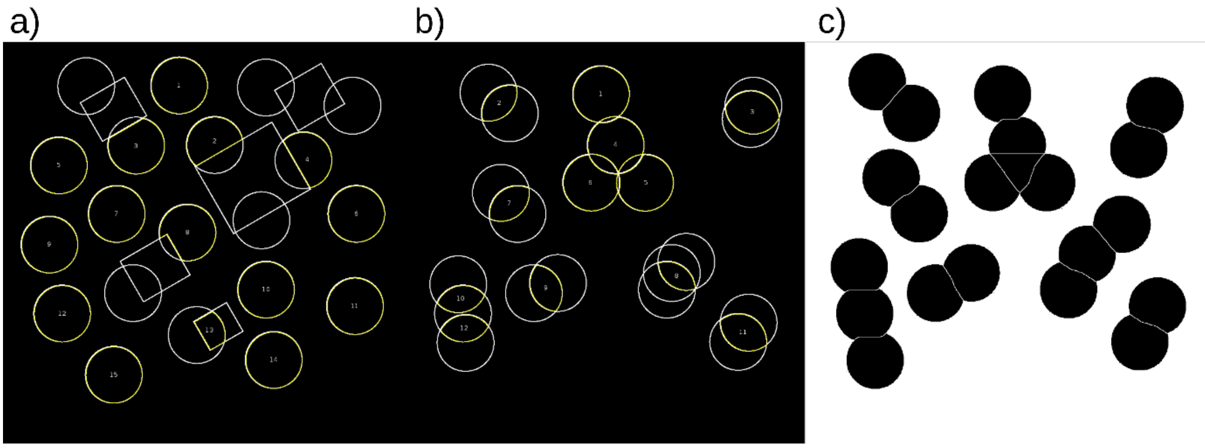

Figure S2 Some limitations of RG technique. (a) shows how square elements intersecting the circles cause mis-identification where the circularity parameter is set too low. Incomplete circles are identified at the cost of circle size accuracy. (b) shows intersecting spheres where the intersection is interpreted as an ‘island’ if the circularity parameter is set too low. (c) shows the use of watershedding (not used elsewhere in this study). Droplets are intersected by watershedding lines, but additional structures (triangular shape) may be introduced where a droplet should be.

2 Emulsion composition

Table S1. Compositions of the different emulsions investigated in this study. Letters in the first column correspond to images in Figure 3 (main manuscript). Image files of all samples are freely available at: https://doi.org/10.17863/CAM.113063

Sample ID	Solid Particle	Oil	Particles mass (wt%)	Dye
1	Azo$(CH_2)_3$OH (0.6)- SiO_2[a]	Diethyl adipate	1	none
2	Azo$(CH_2)_3$OH (1.4)- SiO_2[a]	Diethyl adipate	0.5	none
3 h)	Cu (II) phthalocyanine	Mineral oil	0.2	none
4	Cu (II) phthalocyanine	Mineral oil	0.2	Nile red
5 i)	Casein	Soybean oil	1	none
6 a)	SiO_2	Mineral oil	2.5	none
7	Starch	Mineral oil	0.5	none
8	Starch	Mineral oil	0.75	none
9	Starch	Mineral oil	1	none
10 c)	Starch	Silicone oil	0.5	none
11	Starch	Silicone oil	1	none
12	Starch	Silicone oil	1	none
13 d)	Starch	Silicone oil	2	none
14	AzoOH (1.0)- SiO_2[a]	Diethyl adipate	1	none
15	AzoOH (1.5)- SiO_2[a]	Diethyl adipate	5	none
16	Azo$(CH_2)_{11}$OH (0.4)-SiO_2[a]	Diethyl adipate	0.5	none
17	Azo$(CH_2)_{11}$OH (0.4)- SiO_2[a]	Diethyl adipate	1	none
18	Azo$(CH_2)_{11}$OH (1.5)- SiO_2[a]	Diethyl adipate	5	none
19	Azo$(CH_2)_{11}$OH (1.5)- SiO_2[a]	Diethyl adipate	5	none
20	Azo$(CH_2)_{11}$OH (1.7)- SiO_2[a]	Diethyl adipate	5	none
21	Azo$(CH_2)_{11}$OH (1.7)- SiO_2[a]	Diethyl adipate	5	none
22	Azo$(CH_2)_{11}$OH (0.4)- SiO_2[a]	Diethyl adipate	2	none
23 e)	Azo$(CH_2)_{11}$OH (1.5)- SiO_2[a]	Diethyl adipate	0.5	Nile red
24	Azo$(CH_2)_{11}$OH (1.5)- SiO_2[a]	Diethyl adipate	1	none
25	Azo$(CH_2)_{11}$OH (1.5)- SiO_2[a]	Diethyl adipate	1	none
26	Azo$(CH_2)_{11}$OH (1.5)- SiO_2[a]	Diethyl adipate	1	none

27	Azo$(CH_2)_{11}$OH (1.5)- SiO_2[a]	Diethyl adipate	1	none
28	Azo$(CH_2)_{11}$OH (1.5)-SiO_2[a]	Diethyl adipate	1	none
29 g)	Azo$(CH_2)_{11}$OH (1.5)-SiO_2[a]	Silicone oil	1	none
30 f)	Azo$(CH_2)_{11}$OH (1.5)-SiO_2[a]	Silicone oil	1	Nile red
31	Azo$(CH_2)_{11}$OH (1.5)-SiO_2[a]	Diethyl adipate	1	none
32	Azo$(CH_2)_{11}$OH (1.5)-SiO_2[a]	Diethyl adipate	1	none
33	Azo$(CH_2)_{11}$OH (1.5)-SiO_2[a]	Diethyl adipate	1	none
34	Azo$(CH_2)_3$OH (1.4)-SiO_2[a]	Diethyl adipate	1	none
35	Azo$(CH_2)_3$OH (0.6)- SiO_2[a]	Diethyl adipate	0.5	none
36	Azo$(CH_2)_3$OH (0.6)-SiO_2[a]	Diethyl adipate	0.5	none
37	Azo$(CH_2)_3$OH (1.4)-SiO_2[a]	Diethyl adipate	0.5	none
38	Azo$(CH_2)_3$OH (1.4)-SiO_2[a]	Diethyl adipate	5	none
39	Azo$(CH_2)_3$OH (1.4) -SiO_2[a]	Diethyl adipate	5	none
40	Azo$(CH_2)_3$OH (1.4)-SiO_2[a]	Diethyl adipate	5	none
41	Azo$(CH_2)_3$OH (1.4)-SiO_2[a]	Diethyl adipate	5	none
42	Azo$(CH_2)_3$OH (1.7)-SiO_2[a]	Diethyl adipate	5	none
43	Azo$(CH_2)_3$OH (1.7)-SiO_2[a]	Diethyl adipate	1	none
44 b)	Azo$(CH_2)_3$OH (1.7)-SiO_2[a]	Diethyl adipate	0.5	none
45	Azo$(CH_2)_3$OH (1.7)-SiO_2[a]	Diethyl adipate	0.75	none
46	Azo$(CH_2)_3$OH (1.7)-SiO_2[a]	Diethyl adipate	5	none
47	Azo$(CH_2)_3$OH (0.6)-SiO_2[a]	Diethyl adipate	0.25	none
48	Azo$(CH_2)_3$OH (0.6)-SiO_2[a]	Diethyl adipate	2	none
49	Azo$(CH_2)_3$OH (1.7)-SiO_2[a]	Diethyl adipate	0.25	none
50	Azo$(CH_2)_3$OH (1.7)-SiO_2[a]	Diethyl adipate	0.75	none

[a] Synthesis described in reference [1].

3 Software parameters

In this work, open-source software was developed in order to carry out the semi-automated detection of droplets using the Circle Hough Transform. The software, called “Hough-scan”,[2] combines tools found in the OpenCV library[3] with a graphical user interface (GUI) that makes it easier to select the appropriate image analysis parameters (see Figure S3). The user selectable parameters provided by OpenCV are: blur, minimum distance, canny upper limit, Hough-threshold, min. radius and max. radius. A discussion of each of these parameters is given below below.

On its own, the CHT implementation in OpenCV is not suited to the detection of droplets in images where there are many circular objects, due to the associated computational burden.[4] This is because with more droplets and image detail, there are more locations for the algorithm to scan across (see below). This problem is also compounded when using high-resolution images, with many modern cameras shooting at 4K resolution. The computational requirement can also increase exponentially when the initial guess of the parameters is less accurate. To reduce this, the software breaks the image down into a series of user-defined tiles which overlap. The user can set the tiles to contain, roughly 1-10 droplets. A set of trial parameters can then be tested on a single tile, before processing the entire image. Multiprocessing as part of the Python “multiprocessing” library is also used.[5] This allows tiles to be processed concurrently, decreasing the overall processing time. The above parameters can all be selected, and the output viewed using a GUI, which was developed using the GTK3+ toolkit (see Figure S3).[6] The overall experience is that users can guess parameters for their sample, run the software (which processes on the order of seconds), check the accuracy and then either update the parameters or export their results as a list of droplet locations (x,y) and radii (in pixels, px). In this work, all images were analysed using a desktop computer equipped with a Ryzen 5 3600 processor and 16 GB of RAM. The program and all of its dependencies are open-source and available online.[2]

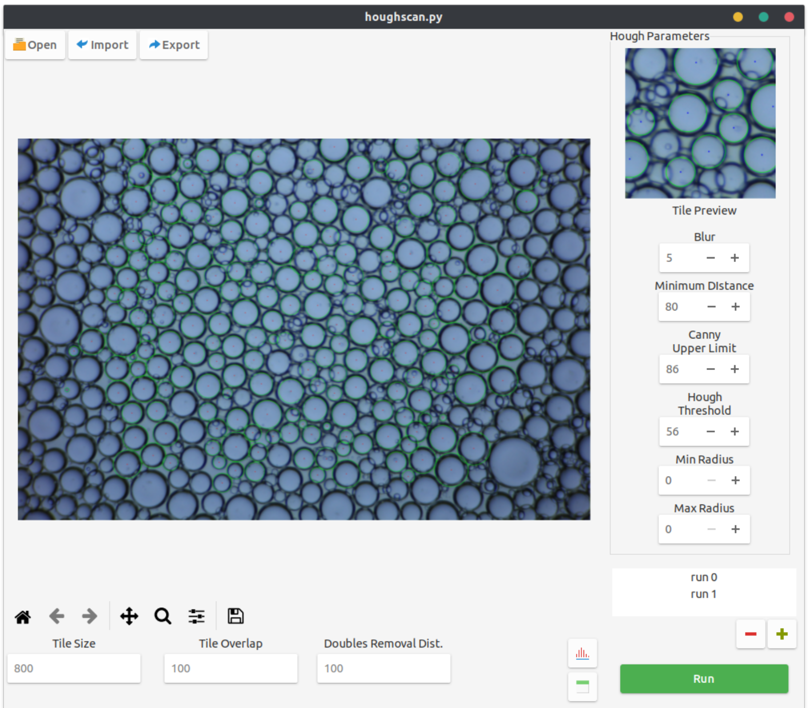

Figure S3: Example screenshot of the GUI for Hough-scan application, showing an optical microscopy image of a Pickering emulsion. User definable parameters are shown on the right and bottom of screen. A tile preview is in the top right corner and the process can be run by clicking the run button in lower right corner.

In order to analyse an image effectively, it is important to understand the origin of the user selectable paraments as part of the CHT provided by OpenCV.[3] As such, a brief explanation is given below, using Figure S4 for reference. The order of the applied operations is as follows: a) Blur; b) Sobel; c) Canny Edge Detection; d) Circle Hough Transform. The user-definable parameters are: Blur, Canny Upper Limit, Hough Threshold, Min Radius, Max Radius and Minimum Distance.

a) Blur

Blur is an example of a kernel convolution processes. For each pixel in the image (red square, Figure S4a, the neighbouring pixels (green square) are added using a weighting which is determined by the kernel (3 × 3 matrix - Greek letters in this case). The output is then often normalised. The Blur parameter adjusts the number of pixels used in blurring, i.e., the size of the matrix. A typical example is given below, where Gx,y is the kernel applied to an image:

$$G_{x,y} = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \qquad \text{Eq. 1}$$

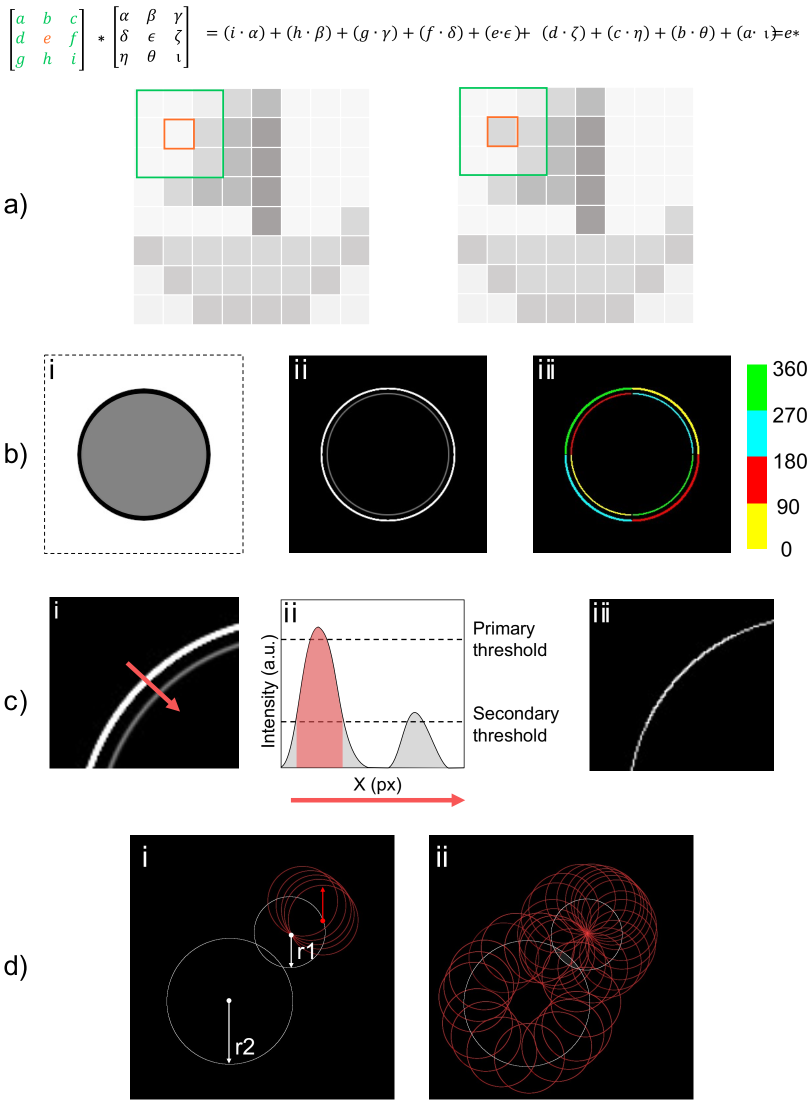

Figure S4. Diagrams of the different stages involved in the Circle Hough Transform, showing the origin of each of the parameters available in the OpenCV module. (a) Kernel convolution used for Blur and Sobel operations. An operation (Greek letters) is applied to the central pixel (red square) using the surrounding pixels (green square), as shown by the central pixel changing from light to dark grey. (b) (i) An example Sobel operation applied to a black circle which is inset with a grey circle and is on a white background. (ii) shows edge detection and (iii) shows the angular information. The coloured scale is in degrees, split at 90° increments. (c) Example Canny Edge Detection for the circle in (b). The inner arc in (i) has a lower intensity than the outer arc (white line vs. grey line). (ii) shows that only the pixels above the primary threshold or above the secondary, but connected to the primary threshold, are retained. (d) Example CHT applied to two overlapping circles of radius r1 and r2. The image is given at an instance where the CHT is scanning at radius rx, where rx is equal to r1. (ii) is the accumulator image and shows that a bright spot is formed after scanning each of the white pixels. The radius and centre of the circle can therefore be calculated.

b) Sobel Operation

The Sobel operation is another example of a kernel convolution process which is used for edge detection. It is applied in both the x and y direction independently and allows calculation of angular information as shown in Figure S4b. This information is then used in the next process - Canny Edge Detection. The kernel convolution matrix used for edge detection is given below:

$$\text{Sobel: } G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} G_y = \begin{bmatrix} -1 & -2 & 1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} G_{x,y} = \sqrt{G_x^2 + G_y^2} G_{\text{orientation}} = \text{atan}\left(\frac{G_x}{G_y}\right) \quad \text{Eq. 2}$$

c) Canny Edge Detection

Canny edge detection uses two thresholds (primary and secondary). A line is drawn in the direction of the 'edge' (see Figure S4c). Values above the primary threshold are retained and values below are removed. If, however, a value is below the primary threshold but above the secondary threshold and is also connected to a point above the primary threshold by connecting pixels (i.e., without dipping below the secondary threshold), the value is retained. The 'Canny Upper' parameter sets the primary threshold and the secondary threshold is set by OpenCV as half of the value of the upper. This operation helps to find the edges of the droplets, which is then used by the Circle Hough Transform.

d) Circle Hough Transform

At this stage the image has been refined to a set of thin white lines/circles on a black background. The CHT will scan across the image until it finds a white pixel. For each pixel, a circle of radius *r* (where *r* is an ever-increasing value upon each pass of the image and is set between two limits) is drawn using the equation for a circle:

4 References

1. K. D. Richards and R. C. Evans, *Soft Matter*, 2022, **18**, 5770–5781.
2. Hough-scan, https://github.com/KRichardsF/Hough-Scan, (accessed 28 May 2021).
3. OpenCV, https://opencv.org/, (accessed 29 September 2021).
4. S. Maaß, J. Rojahn, R. Hänsch and M. Kraume, *Comput. Chem. Eng.*, 2012, **45**, 27–37.
5. Multiprocessing, docs.python.org/3/library/multiprocessing.html, (accessed 28 August 2022).
6. GTK3+, https://www.gtk.org/, (accessed 28 September 2022).