**CellProfiler workshop**

1. **File loading**

*Aim: To load the three test images, extract metadata, split the images into separate channels then group based on the well name.*

* Open CellProfiler
* Right click the file window and select “Browse for images”
* Add metadata extractors for “file header” and “file/folder names” (you’ll want to use the following regular expression)

Image\_(?P<Row>[A-Z]{1})(?P<Col>[0-9]{1})\_F(?P<Field>[0-9]{1}).tif

* Split images into 3 types and assign names (e.g. “Blue”, “Green” and “Red”)
* Add grouping for well number

1. **Identify primary objects**

*Aim: Load the “IdentifyPrimaryObjects” module to the pipeline and use this to pick out the nuclei as separate objects but avoid over-splitting the nuclei.*

* Add “IdentifyPrimaryObjects” module from “ObjectProcessing” to the pipeline
* Enable advanced settings
* Select the input image (“Blue”)
* Give the output objects a name (e.g. “Nuclei”)
* Vary the following parameters to optimise detection of nuclei (feel free to play around with other parameters if necessary):
  + “Typical diameter of objects, in pixel units (Min,Max)”
  + “Thresholding method”
  + “Threshold smoothing scale”
  + “Method to distinguish clumped objects” and “Method to draw dividing lines between clumped objects”
  + “Suppress local maxima that are closer than this minimum allowed distance” (must have “Automatically calculate minimum allowed distance between local maxima” to “No”)

1. **Identify secondary and tertiary objects**

*Aim: Identify cells and cytoplasm using “IdentifySecondaryObjects” and “IdentifyTertiaryObjects”.*

* Add the “IdentifySecondaryObjects” module from “ObjectProcessing”
* Set the input image (“Green”) and input objects (nuclei)
* Give the output objects a name (e.g. “Cells”)
* The cells cover the whole image, so we don’t need to apply a threshold to identify them (we assume they’re everywhere):
  + Set “Thresholding method” to “Manual”
  + Set “Manual threshold” to 0
* Change the method used for identifying secondary objects (“Select the method to identify the secondary objects”) and see what affect this has on the detection.
* It’s also good practise to remove any objects that touch the image edge, as we can’t get a complete analysis of these:
  + Enable “Discard secondary objects touching the border of the image”
* This would leave some nuclei without an associated cell, so:
  + Enable “Discard the associated primary objects” and give the new nuclei objects a name (e.g. “FilteredNuclei”)
* Add the “Identify tertiary objects” module from “ObjectProcessing”
* Select the larger and smaller objects (i.e. cells and nuclei)
* Assign a name to the output objects (e.g. “Cytoplasm”)

1. **Apply measurement-based object filters and relate objects**

*Aim: Detect bacteria, measure their size and filter objects smaller than a specified size. Finally, determine which cells bacteria are in.*

* Add the “Smooth” module from “ImageProcessing”
  + Set the input image and a name for the smoothed image
  + Select the smoothing algorithm
  + Try both automatic and manual diameter settings
* Identify bacteria in “Red” channel using a new “IdentifyPrimaryObjects” module
  + Use a similar approach to detection of nuclei
* Add the “MeasureObjectShapeSize” module from “Measurements”
  + Set it to calculate measurements for the bacteria objects
* Add the “FilterObjects” module from “ObjectProcessing”
  + Set the input objects to the bacteria objects
  + Assign a name for the filtered objects (i.e. the object collection which won’t include the filtered objects)
  + Set the measurement to “Area” and a minimum value of 50
* Add the “RelateObjects” module from “ObjectProcessing”
  + Relate bacteria (children) to cells (parents)

1. **Visualising data**

*Aim: Use “OverlayOutlines” and “DisplayDataOnImage” modules to draw outlines of nuclei, cells and bacteria (assorted colours) on an input image, then add text showing the number of bacteria per cell.*

* Add the “OverlayOutlines” module from “ImageProcessing”
  + Select the base image on which to draw overlays
  + Give a name to the output image
  + Add overlay for nuclei, setting the line colour
  + Repeat for cells and bacteria
* Add the “DrawDataOnImage” module from “ImageProcessing”
  + Select the cell objects
  + Choose the measurement corresponding to the number of bacteria children per cell
  + Select the image created by the “OverlayOutlines” module
  + Set the font size and colour
  + Assign the output image a new name

1. **Exporting data**

*Aim: Save the detected cell objects to an image (each pixel value corresponding to the ID number of that object) and save the measurements to a .csv file.*

* Add the “ConvertObjectsToImage” module from “ObjectProcessing”
  + Select the cell objects
  + Assign a name for the produced image
  + Choose an appropriate colour format
* Add the “SaveImages” module from “FileProcessing”
  + Select the image to save (i.e. the one you just created)
  + Assign a file to use as a prefix for the output name (It doesn’t matter which image you use, it just allows you to relate saved images back to the input file)
  + Select the default output folder
* Add the “ExportToSpreadsheet” module from “FileProcessing”
  + Select the output file location for your file
  + Select “Select the measurements to export” and choose a few measurements to save (it doesn’t matter which)