# run\_workflow\_MK2

import subprocess

import os

def run\_imagej\_macro(imagej\_path, macro\_path, step\_description):

 subprocess.run([imagej\_path, "-macro", macro\_path], check=True)

 print(f"{step\_description} completed successfully.")

def run\_python\_script(script\_path, working\_directory=None, step\_description=None):

 if working\_directory:

 os.chdir(working\_directory)

 subprocess.run(["python", script\_path], check=True)

 print(f"{step\_description} completed successfully.")

# Paths to ImageJ executable and macros

imagej\_path = r"C:\Users\Victoria\fiji-win64\Fiji.app\ImageJ-win64.exe"

macros = [

 (r"C:\Users\Victoria\fiji-win64\Fiji.app\ImageAnalysis\MK2\convert\_czi\_to\_tiff\_MK2.ijm", "Convert to tif"),

 (r"C:\Users\Victoria\fiji-win64\Fiji.app\ImageAnalysis\MK2\CreateMK2nucleiMask.ijm", "Create MK2 nucleus mask"),

 (r"C:\Users\Victoria\fiji-win64\Fiji.app\ImageAnalysis\MK2\CreateMK2cytoplasmMask.ijm", "Create MK2 cell mask"),

 (r"C:\Users\Victoria\fiji-win64\Fiji.app\ImageAnalysis\MK2\CreateTTPnucleiMask.ijm", "Create TTP nucleus mask"),

 (r"C:\Users\Victoria\fiji-win64\Fiji.app\ImageAnalysis\MK2\CreateTTPcytoplasmMask.ijm", "Create TTP cell mask"),

 (r"C:\Users\Victoria\fiji-win64\Fiji.app\ImageAnalysis\MK2\measure\_intensity\_MK2\_nucleus.ijm", "Measure nucleus"),

 (r"C:\Users\Victoria\fiji-win64\Fiji.app\ImageAnalysis\MK2\measure\_intensity\_MK2\_nucleusMK2.ijm", "Measure nucleusMK2"),

 (r"C:\Users\Victoria\fiji-win64\Fiji.app\ImageAnalysis\MK2\measure\_intensity\_MK2\_cellMK2.ijm", "Measure cell MK2"),

 (r"C:\Users\Victoria\fiji-win64\Fiji.app\ImageAnalysis\MK2\measure\_intensity\_MK2\_cytoplasmMK2.ijm", "Measure cytoplasm MK2"),

 (r"C:\Users\Victoria\fiji-win64\Fiji.app\ImageAnalysis\MK2\measure\_intensity\_MK2\_nucleusTTP.ijm", "Measure nucleusTTP"),

 (r"C:\Users\Victoria\fiji-win64\Fiji.app\ImageAnalysis\MK2\measure\_intensity\_MK2\_cellTTP.ijm", "Measure cell TTP"),

 (r"C:\Users\Victoria\fiji-win64\Fiji.app\ImageAnalysis\MK2\measure\_intensity\_MK2\_cytoplasmTTP.ijm", "Measure cytoplasm TTP")

]

# Step 1: Convert to tif

run\_imagej\_macro(imagej\_path, macros[0][0], macros[0][1])

# Step 2: Create Mask in cell pose

# Activate the conda environment

subprocess.run(["conda", "activate", "cellpose\_env"], shell=True, check=True)

print("Conda environment activated.")

# Open GUI (if necessary)

subprocess.run(["python", "-m", "cellpose"], shell=True)

print("Cellpose GUI opened.")

# Run the cell pose script

cellpose\_script\_path = r"C:\Users\Victoria\ImageAnalysis\CellPose\process\_images\_MK2.py"

cellpose\_working\_directory = r"C:\Users\Victoria\ImageAnalysis\CellPose"

run\_python\_script(cellpose\_script\_path, cellpose\_working\_directory, "Cellpose script")

# Step 3: Update Masks

for macro, description in macros[1:5]:

 run\_imagej\_macro(imagej\_path, macro, description)

# Step 4: Measure MK2 in nucleus and cytoplasm

for macro, description in macros[5:9]:

 run\_imagej\_macro(imagej\_path, macro, description)

# Step 5: Measure TTP in nucleus and cytoplasm

for macro, description in macros[9:]:

 run\_imagej\_macro(imagej\_path, macro, description)

print("All steps completed successfully.")

## # Step 1: Convert to tif

// Macro to batch convert CZI files to TIFF using Bio-Formats

// Put this macro in a file named convert\_czi\_to\_tiff.ijm

// Set input and output folder paths

inputDir = "C:\\Users\\Victoria\\ImageAnalysis\\1\_Data\\MK2\\";

outputDir = "C:\\Users\\Victoria\\ImageAnalysis\\2\_TifConversion\\MK2\\";

// Get list of CZI files in the input directory

list = getFileList(inputDir);

// Check if the list is empty

if (list.length == 0) {

 print("No CZI files found in the input directory.");

} else {

 // Disable display updates for faster processing

 setBatchMode(true);

 // Iterate over each CZI file

 for (i = 0; i < list.length; i++) {

 // Print the name of the current file

 print("Processing file: " + list[i]);

 // Open CZI file using Bio-Formats importer

 run("Bio-Formats Importer", "open=[" + inputDir + list[i] + "] autoscale color\_mode=Default rois\_import=[ROI manager] view=Hyperstack stack\_order=XYCZT");

 // Wait for the image to open (adjust the delay if needed)

 wait(1000);

 // Extract filename without extension

 fileName = File.getNameWithoutExtension(list[i]);

 // Construct output file path

 outputFile = outputDir + fileName + ".tif";

 // Save the currently open image as TIFF

 saveAs("Tiff", outputFile);

 // Close the currently open image

 close();

 print("Converted " + list[i] + " to TIFF");

 }

 // Re-enable display updates

 setBatchMode(false);

}

// Close ImageJ

run("Quit");

## # Step 2: Create Mask in cell pose

// Set input and output folder paths

inputFolder = "C:\\Users\\Victoria\\ImageAnalysis\\2\_TifConversion\\MK2\\";

nucleusMaskFolder = "C:\\Users\\Victoria\\ImageAnalysis\\4\_CellposeTag\\MK2\\nucleus\\";

cellMaskFolder = "C:\\Users\\Victoria\\ImageAnalysis\\4\_CellposeTag\\MK2\\cellMK2\\";

outputFolder = "C:\\Users\\Victoria\\ImageAnalysis\\5\_UpdateTag\\MK2\\cytoplasmMK2\\";

// Get a list of image files in the input folder

imageList = getFileList(inputFolder);

// Loop through each image file

for (i = 0; i < imageList.length; i++) {

 // Check if the file is an image

 if (endsWith(imageList[i], ".tif") || endsWith(imageList[i], ".jpg") || endsWith(imageList[i], ".png")) {

 // Extract image name without extension

 imageName = File.getNameWithoutExtension(imageList[i]);

 // Open the nucleus mask (from nucleusMaskFolder)

 open(nucleusMaskFolder + imageName + "\_cp\_masks\_nucleus.tif");

 selectWindow(imageName + "\_cp\_masks\_nucleus.tif");

 // Run ROI generation for nucleus mask

 getStatistics(area, mean, min, max, std, histogram);

 for (j = 1; j <= max; j++) { // Loop through threshold values

 setThreshold(j, j);

 run("Create Selection");

 if (selectionType() != -1) { // Check if a selection is created

 roiManager("add");

 print("Added ROI for threshold: " + j);

 } else {

 print("No ROI created for threshold: " + j);

 }

 resetThreshold(); // Reset threshold after each iteration

 }

 // Save ROIs from mask

 roiManager("Deselect");

 roiManager("save", nucleusMaskFolder + imageName + "\_rois.zip");

 roiManager("reset");

 // Open the cell mask (from cellMaskFolder)

 open(cellMaskFolder + imageName + "\_cp\_masks\_cellMK2.tif");

 selectWindow(imageName + "\_cp\_masks\_cellMK2.tif");

 // Show all ROIs

 roiManager("Open", nucleusMaskFolder + imageName + "\_rois.zip");

 roiManager("Show All without labels");

 // Add and fill ROIs

 //roiManager("Deselect");

 //roiManager("Select All");

 roiManager("Fill");

 // Save new image with extension \_cp\_masks\_cytoplasmMK2

 saveAs("Tiff", outputFolder + imageName + "\_cp\_masks\_cytoplasmMK2.tif");

 // Close all images to start fresh for the next iteration

 run("Close All");

 }

}

// Close ImageJ

run("Quit");

## # Step 3: Update Masks

// Set input and output folder paths

inputFolder = "C:\\Users\\Victoria\\ImageAnalysis\\2\_TifConversion\\MK2\\";

nucleusMaskFolder = "C:\\Users\\Victoria\\ImageAnalysis\\4\_CellposeTag\\MK2\\nucleus\\";

cellMaskFolder = "C:\\Users\\Victoria\\ImageAnalysis\\4\_CellposeTag\\MK2\\cellMK2\\";

outputFolder = "C:\\Users\\Victoria\\ImageAnalysis\\5\_UpdateTag\\MK2\\nucleusMK2";

// Get a list of image files in the input folder

imageList = getFileList(inputFolder);

// Loop through each image file

for (i = 0; i < imageList.length; i++) {

 // Check if the file is an image

 if (endsWith(imageList[i], ".tif") || endsWith(imageList[i], ".jpg") || endsWith(imageList[i], ".png")) {

 // Extract image name without extension

 imageName = File.getNameWithoutExtension(imageList[i]);

 // Open the nucleus mask (from nucleusMaskFolder)

 open(nucleusMaskFolder + imageName + "\_cp\_masks\_nucleus.tif");

 selectWindow(imageName + "\_cp\_masks\_nucleus.tif");

 run("Rename...", "title=nucleus");

 // Open the cell mask (from cellMaskFolder)

 open(cellMaskFolder + imageName + "\_cp\_masks\_cellMK2.tif");

 selectWindow(imageName + "\_cp\_masks\_cellMK2.tif");

 run("Rename...", "title=cell");

 // Process Image calculator and use AND function on both images

 run("Image Calculator...", "image1=nucleus operation=AND image2=cell create");

 // Select newly generated image

 selectWindow("Result of nucleus");

 // Save newly generated image in output folder with extension \_nucleusMK2

 saveAs("Tiff", outputFolder + "\\" + imageName + "\_cp\_masks\_nucleusMK2.tif");

 // Close all images to start fresh for the next iteration

 run("Close All");

 }

}

// close imageJ

run("Quit");

## # Step 4: Measure MK2 in nucleus and cytoplasm

// Set input and output folder paths

inputFolder = "C:\\Users\\Victoria\\ImageAnalysis\\2\_TifConversion\\MK2\\";

MaskFolder = "C:\\Users\\Victoria\\ImageAnalysis\\5\_UpdateTag\\MK2\\cytoplasmMK2\\";

outputFolder = "C:\\Users\\Victoria\\ImageAnalysis\\6\_Measurement\\MK2\\cytoplasmMK2\\";

// Get a list of image files in the input folder

imageList = getFileList(inputFolder);

// Loop through each image file

for (i = 0; i < imageList.length; i++) {

 // Check if the file is an image

 if (endsWith(imageList[i], ".tif") || endsWith(imageList[i], ".jpg") || endsWith(imageList[i], ".png")) {

 // Extract image name without extension

 imageName = File.getNameWithoutExtension(imageList[i]);

 // Open the mask (from MaskFolder)

 open(MaskFolder + imageName + "\_cp\_masks\_cytoplasmMK2.tif");

 selectImage(imageName + "\_cp\_masks\_cytoplasmMK2.tif");

 // Run ROI generation for nucleus mask

 getStatistics(area, mean, min, max, std, histogram);

 for (j = 1; j <= max; j++) { // Loop through threshold values

 setThreshold(j, j);

 run("Create Selection");

 if (selectionType() != -1) { // Check if a selection is created

 roiManager("add");

 print("Added ROI for threshold: " + j);

 } else {

 print("No ROI created for threshold: " + j);

 }

 resetThreshold(); // Reset threshold after each iteration

 }

 // Save ROIs from mask

 roiManager("Deselect");

 roiManager("save", MaskFolder + imageName + "\_rois.zip");

 roiManager("reset");

 // Open the input image (multichannel)

 open(inputFolder + imageList[i]);

 selectImage(imageList[i]);

 // Select channel 1

 // Load and show ROIs for nucleus

 roiManager("open", MaskFolder + imageName + "\_rois.zip");

 roiManager("Show All");

 // Set measurement options to include more parameters

 run("Set Measurements...", "area mean min max integrated std perimeter shape display redirect=None decimal=3");

 // Measure ROIs

 selectImage(imageList[i]);

 roiManager("Measure");

 saveAs("Results", outputFolder + imageName + "\_Results.csv");

 roiManager("reset");

 run("Clear Results");

 // Close all images to start fresh for the next iteration

 close();

 }

}

// Close ImageJ

run("Quit");

// Set input and output folder paths

inputFolder = "C:\\Users\\Victoria\\ImageAnalysis\\2\_TifConversion\\MK2\\";

MaskFolder = "C:\\Users\\Victoria\\ImageAnalysis\\5\_UpdateTag\\MK2\\nucleusMK2\\";

outputFolder = "C:\\Users\\Victoria\\ImageAnalysis\\6\_Measurement\\MK2\\nucleusMK2\\";

// Get a list of image files in the input folder

imageList = getFileList(inputFolder);

// Loop through each image file

for (i = 0; i < imageList.length; i++) {

 // Check if the file is an image

 if (endsWith(imageList[i], ".tif") || endsWith(imageList[i], ".jpg") || endsWith(imageList[i], ".png")) {

 // Extract image name without extension

 imageName = File.getNameWithoutExtension(imageList[i]);

 // Open the mask (from MaskFolder)

 open(MaskFolder + imageName + "\_cp\_masks\_nucleusMK2.tif");

 selectImage(imageName + "\_cp\_masks\_nucleusMK2.tif");

 // Run ROI generation for nucleus mask

 getStatistics(area, mean, min, max, std, histogram);

 for (j = 1; j <= max; j++) { // Loop through threshold values

 setThreshold(j, j);

 run("Create Selection");

 if (selectionType() != -1) { // Check if a selection is created

 roiManager("add");

 print("Added ROI for threshold: " + j);

 } else {

 print("No ROI created for threshold: " + j);

 }

 resetThreshold(); // Reset threshold after each iteration

 }

 // Save ROIs from mask

 roiManager("Deselect");

 roiManager("save", MaskFolder + imageName + "\_rois.zip");

 roiManager("reset");

 // Open the input image (multichannel)

 open(inputFolder + imageList[i]);

 selectImage(imageList[i]);

 // Select channel 1

 // Load and show ROIs for nucleus

 roiManager("open", MaskFolder + imageName + "\_rois.zip");

 roiManager("Show All");

 // Set measurement options to include more parameters

 run("Set Measurements...", "area mean min max integrated std perimeter shape display redirect=None decimal=3");

 // Measure ROIs

 selectImage(imageList[i]);

 roiManager("Measure");

 saveAs("Results", outputFolder + imageName + "\_Results.csv");

 roiManager("reset");

 run("Clear Results");

 // Close all images to start fresh for the next iteration

 close();

 }

}

// Close ImageJ

run("Quit");