Cancer Cell Analysis

Group members:

Ben Christian, Craig Gabel, Eric Helander, Jeffrey Kropelnicki

Class:

SEIS 764-01 Artificial Intelligence, Professor Chih Lai, Fall 2019

GitHub repository:

<https://github.com/CraigStThomas/SEIS_764_Project/tree/deliverables>

The readme.md in that repository link is a great version of this file (with functional links). The provided zip file (SEIS\_764\_Project.zip) is a compressed version of the linked repository branch.

# cell\_object\_detection

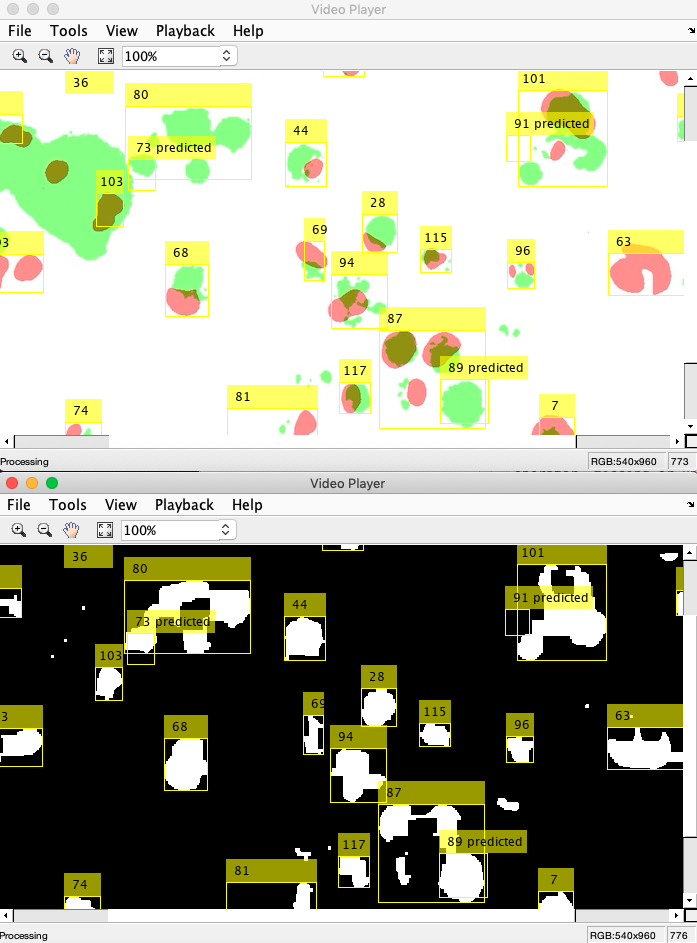
* Contains the jupyter notebook for the cell object detection
* To run, please follow the guide in the Jupyter Notebook which will instruct you how to setup the Azure VM.

# LSTM\_frame\_prediction

* dependencies for lstm.py: tensor flow, keras, numpy, matplotlib, opencv
* To run the application:
  + run with the command: python lstm.py
    - output files are the predicted images, labelled with their order
      * y\_test\_combined\_\*.png <-- these represent the results of prediction method 1 ("first method" as described in slide 15 of provided pptx presentation)
      * future\_combined\_\*.png <-- these represent the results of prediction method 2 ("second method" as described in slide 15 of provided pptx presentation)
      * for both sets of outputs, the y\_true image appears on the left, and the y\_predicted iamge appears on the right
      * you can control which sets of predictions are created by commenting out either line 138 or 139
    - the submitted code file should read the provided weights file (weights\_gp.h5) and make predictions. You can train your own model by commenting out line 136, and uncommenting line 134 (and optionally line 135)
    - important model hyperparameters:
      * model architecture: lines 16-36
        + LSTM layer count, filter count, kernel size, input shape
      * source and target data: lines 42-47
        + sequence size, sequence count, image dimensions, image directories to use (by default is only 1...you need lots of memory to use more than 1)
      * training parameters: line 134
        + epoch count, batch size, size of validation split

# matlabMotionBasedMultipleObjectTracking

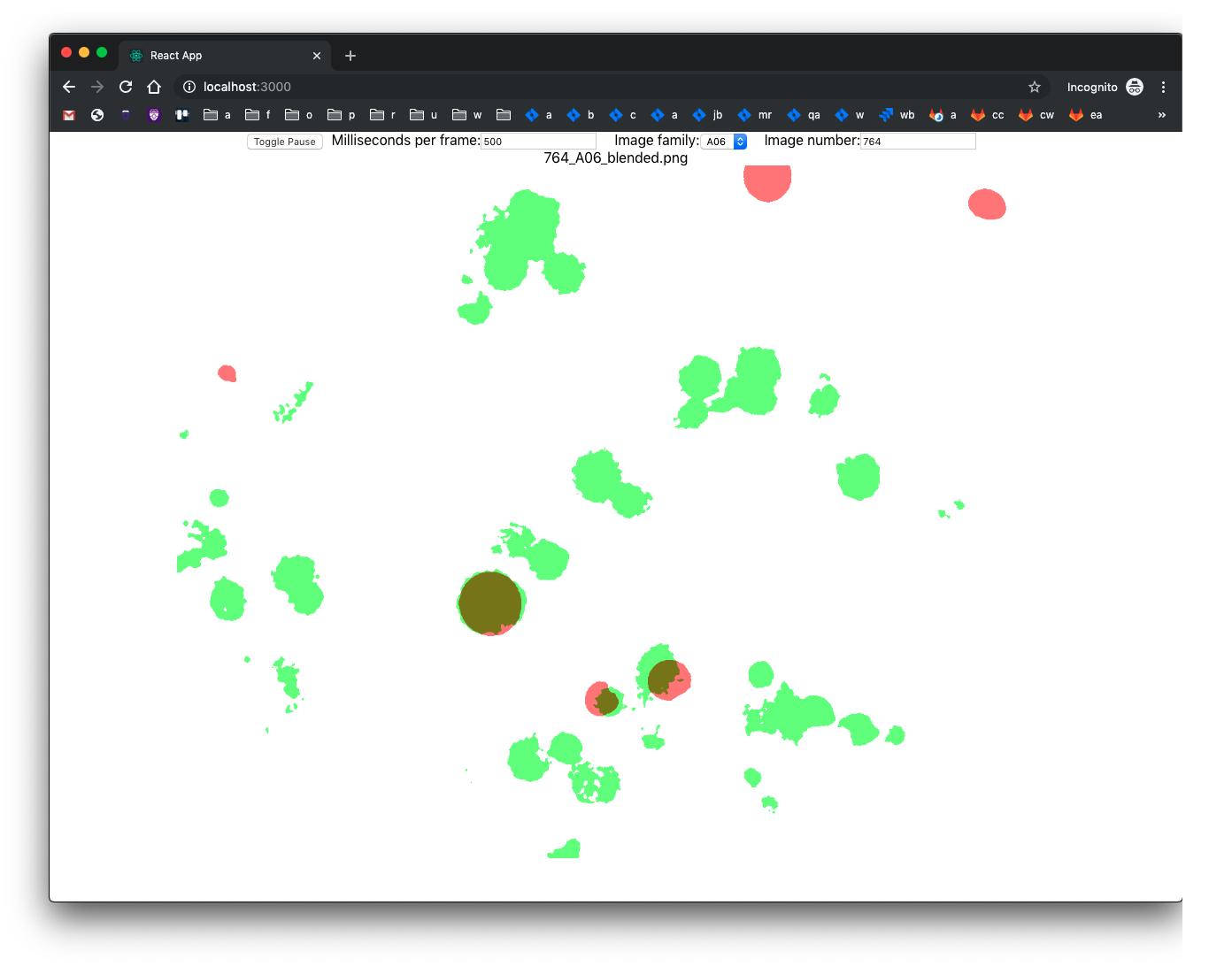
* The MatLab motionBasedMultipleObjectTracking detects and tracks object motion in a video file. The following screenshot is from the A01 set of images:

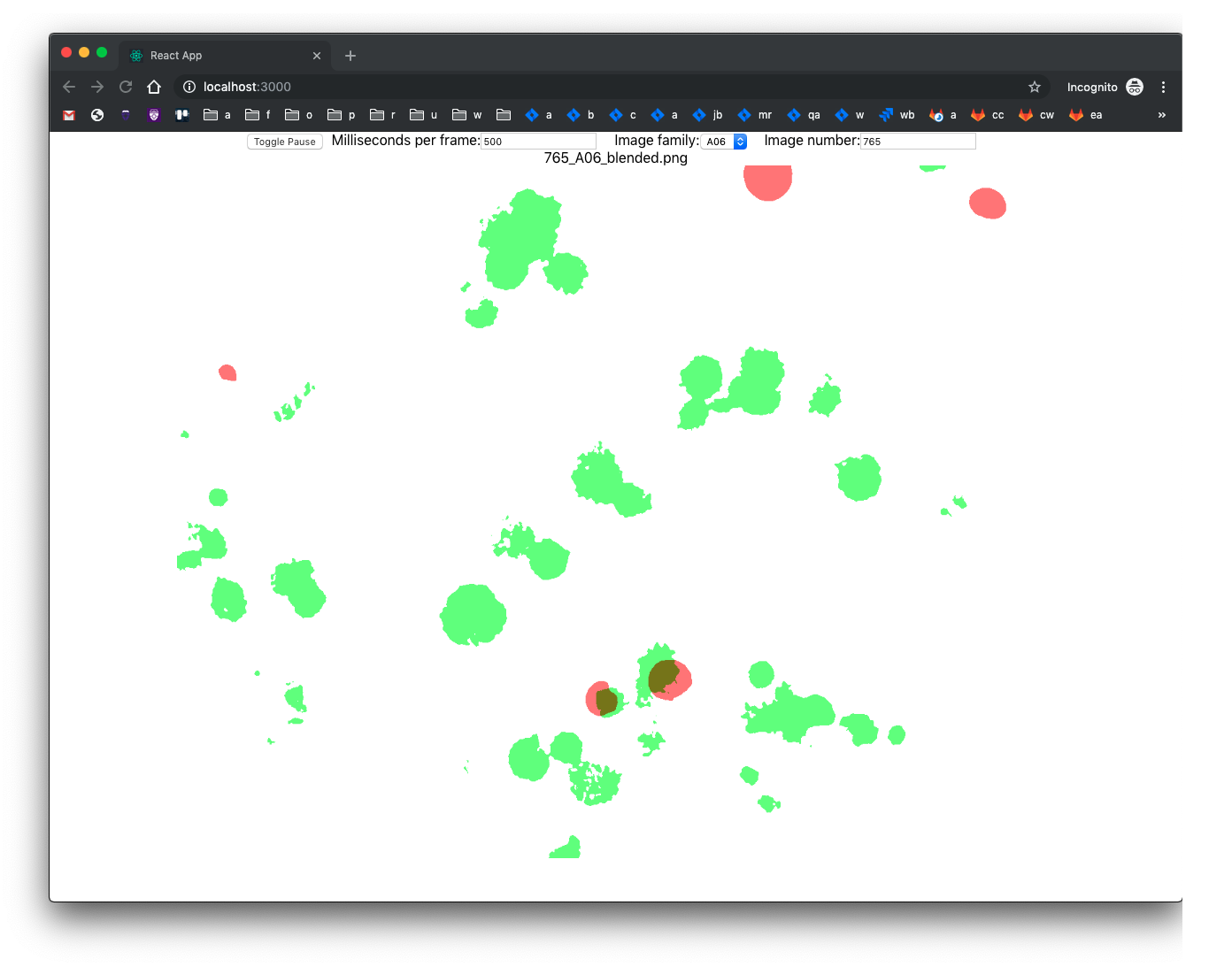


* To run the application:
  + Open MatLab and run matlabMotionBasedMultipleObjectTracking/motionBasedMultipleObjectTracking.m.
    - The file currently points to matlabMotionBasedMultipleObjectTracking/A\_01.mp4. To use a different video file, simply change the VIDEO\_FILE path.

# reactPhotoViewer

The browser-based React Photo Viewer application enables a user to view images sequentially, with full control over the number of milliseconds per frame. This is particularly useful for observing changes between frames, such as the phase change observed in image family A06 between images 764 and 765.





* Prerequisites for running the application:
  + Install Node.js
  + Install yarn
* To run the application:
  + cd into the reactPhotoViewer directory.
  + Run yarn to install the application's dependencies.
  + Run yarn start to start the application.
  + Navigate to <http://localhost:3000/>

# images

* Contains the images used by the cell\_object\_detection and the LSTM\_frame\_predictions
* The original images were divided into 6 different captures with each of them containing 3 different color channels (green, red, and colorless) The color channels are determined by the phase-dependent nature of replication licensing factors Cdt1 and Geminin. Also referenced as the Fluorescent Ubiquitination-based Cell Cycle Indicator (FUCCI)
* Programs:
  + generate\_cell\_detection\_dataset.py
    - Program to create the annotations used by the cell object detection
    - To run, simply start program and it will find the images based on their relative path
    - Reads from the ./images/raw folder and outputs the annotations and the annotation images to ./images/raw/cell\_obj\_detection\_images
  + blend\_raw\_images.py
    - Program to create the blended images used by the LSTM frame prediction
    - To run, simply start program and it will find the images based on their relative path
    - Reads from the ./images/raw folder and outputs the blended images to the ./images/{image set}/
  + compress.sh
    - used to resize the images to different sizes
      * source folders = A0x
      * destination folders = A0x\_compressed
* Directories
  + A01, A02, A03, A04, A05, A06: 3 channel color images (high-resolution)
  + A0x\_compressed: these images are compressed for use with memory and speed limited computing resources. They currently hold resized versions of the original images (original: 1328 x 1048, resized: 120x95).