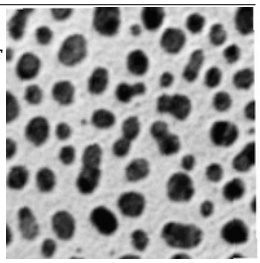
Methods for Video Analysis in Bio-medical Images (an ImageJ/Fiji based tutorial)

A few links : http://gibbs.engr.ccny.cuny.edu/technical/Tracking/RoachTrack.php for MATLAB fans *or* http://fiji.sc/Fiji or http://icy.bioimageanalysis.org/ or http://www.ipol.im/ . **I chose Fiji.**

Resources: http://www.math-info.univ-paris5.fr/~lomn/Cours/CV/BME including the lecture
(file ***TPVideo.pdf*** and software fiji-linux64.Core.tar.gz if needed otherwise install from http://fiji.sc/Fiji and repertory /***Data*** from there)

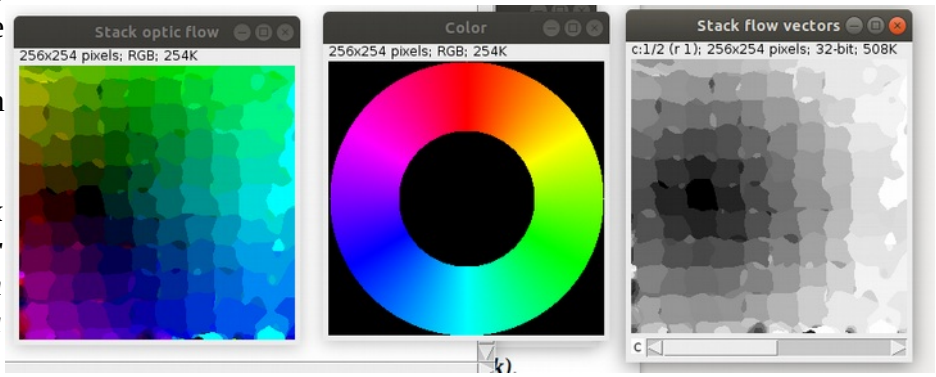
Part 1 : Optical Flow



- Start by applying a **small** rigid transform to an image like the *blobs.gif* one (all images etc. are in the Data folder). ***Menu Plugins/Transform/Interactive Rigid*** *(this menu exists in Fiji : rotate 5 degrees and translate 5 pixels if the menu is not available or Image/Transform). From there, you will have to be flexible and explore the various menus that can change from one version to another of FiJi to find the operator we need and use the internet if necessary to answer most questions.*
- Save *blobs.gif* as *blobs1.gif* and the transformed one as *blobs2.gif*.
- Open the two images and make a stack out of both of them. ***Menu Images/Stack/Images to Stack (or so)***

→ a stack is like a video sequence for us ? Can it be like a 3D image ?

- Apply the plugin ***Optical Flow/Gaussian approach*** *and check the* ***Display color map box.*** *(exists in Fiji in the menu Plugins/Optic Flow)*



→ What does it mean Gaussian here ? In addition to the color field, we get a stack of two images with (r,phi) values in polar coordinates. Interpret the output of the plugins in term of optical flow.

- Do it again after applying an ***Integral Image Filter*** like the ***Mean*** on the stacked images.

→ Are the results very different ? Do you know the difference between Mean and Median filtering ?

- From *books*, create an image sequence by importation of the folder *books* (***Images to Stack***). Then apply image processing on it for pre-processing and then use the ***Optical Flow*** menus.
- Then compute various optical flows with the ***FlowJ plugin*** *(exists in Fiji in the menu Analyze/Optic Flow)*

N.B. Lukas and Kanade which is one of the option we can choose is the extended algorithm based on the one described in the lecture (CONST_FLOW)

→ *what is PIV and explain its use in biology https://imagej.net/PIV_analyser ?*

- From two medical images from a sequence like *t420.tif* and *t421.tif (from the ISBI 2015 challenge http://celltrackingchallenge.net/ these images come from the FLUO_C3DL_MDA231 files),* find **differences**, **SIFT points of correspondences**, **flow etc. (Menu : Process/Image Calculator , Plugins/Feature Extraction/Extract SIFT corrrespondences, etc.).** Explore theses sub-menus **Feature Extraction + Landmarks + Optic Flow + Registration).**

→ what kind of biological images are *t420.tif* and *t421.tif*? These images are 3D+t. Are they coming from two 3D different slices at the same time step or are they two consecutive slices of a 3D cube of data ?

- With *t026.tif* and *t027.tif* that come from another dataset from these challenges, let's create a stack, binarize by threshold adjustment, remove outliers, filter with a mean and process the Gaussian optical flow. Do the same analysis as before.
- On ***book.gif*** or ***book.zip***, try the ***Lucas-Kanade*** one in ***Analyze/Optic Flow.*** In other words, try to become fluent in Fiji and Motion analysis :-)

→ *is it adapted to this sequence ?*

N.B. This sequence can be downloaded from Fluo-C2DL-MSC.zip files (careful big data).

→ *what kind of biological images are they?*

Keep on exploring the various datasets at this URL including the annotation template :
https://public.celltrackingchallenge.net/documents/Naming%20and%20file%20content%20conventions.pdf

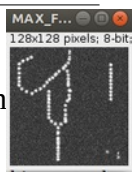
---

Part 2 (optional to do at home if you like) : ffmepg to manipulate video sequence (for engineers audience but physicians may give it a try). It is a tool very useful for video editing (linked to the codec library lib-av). Find below examples :

ffmpeg -i input.avi -c:a aac -b:a 128k -c:v libx264 -crf 23 output.mp4; ffprobe -show_streams -i "file.mp4";
mediainfo Dream.House.sample.mkv ; avprobe -show_streams file.mp4;
ffmpeg -i slow.mp4 -s 320x240 -c:a copy smallslow.mp4 ;
ffmpeg -i video.avi -an -vcodec rawvideo -y video2.avi ;
ffmpeg -i redcar.mp4 -i redcareverse.mp4 -filter_complex "blend=all_mode='overlay':all_opacity=1.0" output3.mp4

You need to distinguish between the container mp4 and the coding scheme like h264 bitstream see an example from **https://stackoverflow.com/questions/7333232/concatenate-two-mp4-files-using-ffmpeg)** (*ffmpeg -i input1.mp4 -c copy -bsf:v h264_mp4toannexb -f mpegts input1.ts; ffmpeg -i input2.mp4 -c copy -bsf:v h264_mp4toannexb -f mpegts input2.ts; ffmpeg -i "concat:input1.ts|input2.ts" -c copy output.mp4*

- Explain why some *avi* videos in the tutorial folders http://helios.mi.parisdescartes.fr/~lomn/Cours/CV/BME/Python/samples/ are possibly unreadable by Fiji.
- Find a solution and then create the videos *redcaroverlay.mp4* video.



Part 3 : Tracking (you can jump to the Python part if you prefer next page)

- Open the image *Track_for_TrakMate* from the samples menu (or *FakeTracks.gif* on my tutorial web site Data folder. Play the video.
- Project the stacks on one image with max intensity (Menu ***Image/Stack/Zproject***). What do we obtain ? Threshold the result with Adjust threshold. Comment.
- Try to use the TrackMate plugin in the Tracking chapter (https://imagej.net/Getting_started_with_TrackMate and https://imagej.net/TrackMate)
- Then launch the tracker ***MTrack2*** *(http://fiji.sc/MTrack2 and http://www.imagescience.org/meijering/software/mtrackj/)*
- Redo it with another threshold to remove or add noise. Which tracks to keep ?

To go beyond (Fiji is always evolving by plugin add-ons in labs : your own one soon maybe, it is implemented in Java but API for Java/Python exist :-)) :

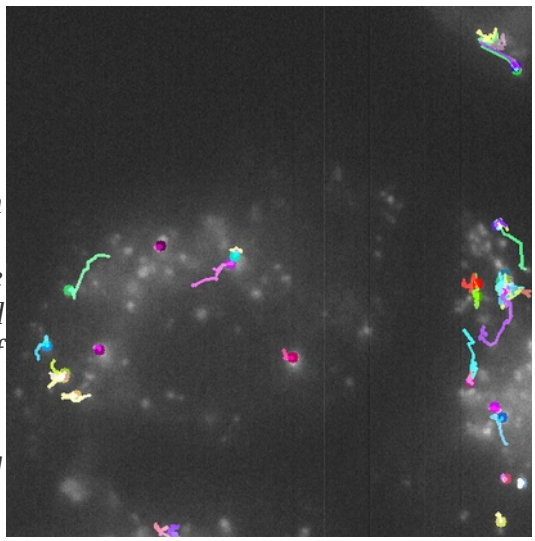
The **ToAST** plugin (Tool for Automated Sporozoite Tracking or more generally to adress the questions of motility directionality http://fiji.sc/ToAST) with image *Malaria Sporozoites.* The MOSAIC plugin[1] to efficiently track multiple targets.

1 http://mosaic.mpi-cbg.de/ParticleTracker/ and http://fiji.sc/Particle_Tracker
http://mosaic.mpi-cbg.de/?q=downloads/imageJ and https://depts.washington.edu/soslab/gro/tutorial/gro-tutorial-4.pdf

- We switch to Python (no more Fiji/ImagJ) in case you want to start programming something in a easy way. Explore the Python examples in the Repertory /**Python** just by running them (check if cv2 is installed ou opencv library for python). Of course, ***avi*** files are in the Data folder. N.B. : these codes are running with python2 (python2.7 prog.py to switch to it, to adapt to python 3 see the update here https://docs.opencv.org/4.8.0/d4/dee/tutorial_optical_flow.html :

$python mon_programme.py [argument1]
Synopsis of the python codes : argument1 stands for the video sequence to analyse
Example : $python background.py redcar.avi

To go further in Python



Using
$python lkOpticalFlow.py bio1.avi
outputs the result there ---------->
Can you play with the parameters of both ***GoodFeaturesToTrack***
and ***calcOpticalFlowPyrLK*** *as asked in the core of the provided code at the level of* ***feature_params*** *and* **lk_params** *to improve the quality or the number of tracked spots.*

Test and improve the tracking result for the following videos :
- sheep.mp4
- dog.mp4
- beys.mp4

```
import numpy as np
import cv2
import sys

cap = cv2.VideoCapture(sys.argv[1])

# params for ShiTomasi corner detection
#https://docs.opencv.org/3.4/d8/dd8/tutorial_good_features_to_track.html
feature_params = dict( maxCorners = 50,qualityLevel = 0.3,minDistance = 7,blockSize = 7 )

#We can play on these parameters
#what happens if minDistance = 7
#what happens if blocksize = 3
#what happens if minDistance = 1

# Parameters for lucas kanade optical flow
# https://justinshenk.github.io/posts/2018/04/optical-flow/
lk_params = dict( winSize  = (15,15),
                  maxLevel = 2,
                  criteria = (cv2.TERM_CRITERIA_EPS | cv2.TERM_CRITERIA_COUNT, 10, 0.03))

#We can play on these parameters as well
#what happens if winSize = (7,7)
#what happens if maxLevel = 0
#what happens if criteria = max number of iteration set to 2 instead of 10
#what happens if criteria = max epsilon set to 0.1 instead of 0.03

# Create some random colors
color = np.random.randint(0,255,(50,3))

# Take first frame and find corners in it
ret, old_frame = cap.read()
old_gray = cv2.cvtColor(old_frame, cv2.COLOR_BGR2GRAY)
p0 = cv2.goodFeaturesToTrack(old_gray, mask = None, **feature_params)

 # Create a mask image for drawing purposes
mask = np.zeros_like(old_frame)

while(1):
    ret,frame = cap.read()
    frame_gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

     # calculate optical flow
    p1, st, err = cv2.calcOpticalFlowPyrLK(old_gray, frame_gray, p0, None, **lk_params)

     # Select good points
    good_new = p1[st==1]
    good_old = p0[st==1]

     # draw the tracks
    for i,(new,old) in enumerate(zip(good_new,good_old)):
          a,b = new.ravel()
          c,d = old.ravel()
          mask = cv2.line(mask, (a,b),(c,d), color[i].tolist(), 2)
          frame = cv2.circle(frame,(a,b),5,color[i].tolist(),-1)

    img = cv2.add(frame,mask)

    cv2.imshow('frame',img)
    k = cv2.waitKey(300) & 0xff
    if k == 27:
          break


# Now update the previous frame and previous points
    old_gray = frame_gray.copy()
    p0 = good_new.reshape(-1,1,2)

    if cap.get(cv2.CAP_PROP_POS_FRAMES) == cap.get(cv2.CAP_PROP_FRAME_COUNT):
          break

print("Done still a few sec")
cv2.waitKey(5000)
cv2.destroyAllWindows()
cap.release()
```