Favorites are highlighted.

FIRST, please note if you’re using Python, you can see a pure Python implementation of optical flow at <https://github.com/scivision/pyoptflow/tree/4485da929c5567470d821c3f2a23e3bccc7f768e/src/pyoptflow>

The broadest background about deriving motion or morphology aka structure from images and movies:

* <https://users.fmrib.ox.ac.uk/~steve/review/review/node1.html#SECTION00010000000000000000>
* summarizing the main methods: feature, gradient, correlation based methods
* Deciding between feature matching optical flow: <https://stackoverflow.com/questions/47465570/opencv-feature-matching-vs-optical-flow>

On Optical flow

* <http://www.cs.cmu.edu/~16385/s15/lectures/Lecture21.pdf> (great slides, notes below come from this)
* A way to put optical flow in physics terms if people like is that we are just taking a material derivative where
* D/Dt = d/dt + v dot grad…or DI/Dt = 0, so dI/dt + v dot grad(I) = 0!
* Do I need to worry about non-cartesian coordinates? Or maybe accounted for with drizzlepac? Or adding very simple fluid constraints about divergence, curl, intensity-density correspondence?
* -Lucas-Kanade assumes a constant (aka affine) motion around a pixel i. This makes the problem overdetermimed, better locally where this may be true
* -Horn-Schunk (<https://www.cmor-faculty.rice.edu/~zhang/caam699/opt-flow/horn81.pdf> – the original paper) is applied globally with an assumption of smoothing, solving a constraint function (like lambda \* gradvx^2 + lambda\*gradvz^2). The key question is if this holds true, truly, then it can be solved via finite differences…the key application is rigid body movement applied to the Stokes equation (but typically the divergence term si dropped for some reason?). These are associated with diffusion (viscous) and pressure gradient of your fluid. This is also very similar to the Euler-Lagrange equation! The finite differences is used to solve what’s called a variational problem, aka minimizing an energy functional.
* We can show how well this works by showing areas of smooth gradient (is this effectively smoothing or convolving the image…?). For example, non-linear gradients breaks this. Large displacements break this.
* Typically this is solved by a “pyramidal structure” (a pyramid scheme?) with coarse-to-fine refinement of the flow (EX: <https://www.ipol.im/pub/art/2013/20/article.pdf>). Also see any Pythonic implementations (EX: please fill in) of the Gunnar-Farneback method
* There are also “illumination-robust” cases (<https://uwaterloo.ca/vision-image-processing-lab/sites/ca.vision-image-processing-lab/files/uploads/files/opticalflow-tip_doublecolumn_2.pdf>)

People then visualize these velocity vectors in various ways:

* Flow fields or streamlines
* Color wheels ( see <https://github.com/tomrunia/OpticalFlow_Visualization/blob/master/flow_vis/flow_vis.py>)
* Particle tracing (see for example one group called “flowtrace” which has a nature paper or else see particle image velocimetry, “PIV”)
* Maps or contours of the magnitude and angle (less common)
* A mix of methods (like colored arrows, see <https://oflibpytorch.readthedocs.io/en/2.1.1/>)

Other examples writeups on optical flow:

* <https://www.dgp.toronto.edu/~donovan/stabilization/opticalflow.pdf> or https://www.cs.toronto.edu/~fleet/research/Papers/flowChapter05.pdf (slightly more math)
* <http://ipa.math.uni-heidelberg.de/dokuwiki/lib/exe/fetch.php?media=people:spetra:of_becker_et_al_2015.pdf> or <https://www.math.sk/mpm/wp-content/uploads/2018/06/AUTOREFERAT.pdf> (…more math)
* -optical flow involves some assumptions, the aperture problem
* -correlation based methods only produce integer values unlike the other methods (pixel scale limits typically)
* -hierarchical methods can also be used to check motions at different scales (e.g. checking against the motion at lower-res vs higher-res sampling) – see <https://github.com/pderian/PyTyphoon/blob/master/pytyphoon.py> for example (a separate script). There is also <https://github.com/theHamsta/farneback3d/tree/master/farneback3d> for example (but can’t get it working because it requires pycuda?)

(optional idea) How to setup the FITS data for use in OpenCV (open computer vision, one of the main ways people implement this with computers):

* <https://stackoverflow.com/questions/46689428/convert-np-array-of-type-float64-to-type-uint8-scaling-values?rq=1>
* <https://stackoverflow.com/questions/45913007/finding-sunspots-on-fits-image-via-opencv>
* In both, image segmentation and thresholding are recommended ways to break apart images and see many different features…
* There are also issues of getting the segments to overlap (e.g. if the segment’s or object’s motion is not contiguous frame to frame)

***Conclusions***: Optical flow can work great either for global dense flows (HS) or local sparse flows (LK). You can also histogram them for the speeds. Unsure on angles. How to do statistics of that is unclear… for more you can see. The best works on this for us are:

* <https://amt.copernicus.org/preprints/amt-2017-185/amt-2017-185-manuscript-version3.pdf>
* <https://www.mdpi.com/2073-4441/11/11/2320/htm>
* <https://www.nature.com/articles/s41598-022-16145-y>

Other example: histograms/stats for speeds within an aperture or “object”…

* <http://www.vision.jhu.edu/activityRecognition.htm> (good overview of different metrics people use)
* <https://www.researchgate.net/publication/348078287_Investigating_Memorability_of_Dynamic_Media> (some different example statistics have been used: mean, variance, along with the correlation coefficients like the spearman, Pearson, and MSE=mean square error …)
* <https://www.researchgate.net/publication/284727059_Grid-based_Histogram_of_Oriented_Optical_Flow_for_Analyzing_Movements_on_Video_Data>
* <https://projet.liris.cnrs.fr/imagine/pub/proceedings/ICPR-2016/media/files/1818.pdf> (very brief use of gaussian mixture modeling)
* <https://stackoverflow.com/questions/67777391/most-efficient-way-to-generate-histograms-of-oriented-optical-flow-hoof-in-pyt> (some general work, not bad)
* <https://www.researchgate.net/publication/284727059_Grid-based_Histogram_of_Oriented_Optical_Flow_for_Analyzing_Movements_on_Video_Data> (on false positives)
* <https://www.researchgate.net/publication/305083232_Automated_Detection_of_Similar_Human_Actions_using_Motion_Descriptors> (SVM, machine learning auto algorithm?)
* <https://www.researchgate.net/publication/277574336_Detection_of_Abnormal_Events_via_Optical_Flow_Feature_Analysis>

Aside from or in addition to optical flow, ***correlational methods*** are used, for this see:

* Astronomy/astropy = <https://image-registration.readthedocs.io/en/latest/index.html> and <https://www.aanda.org/articles/aa/pdf/2005/43/aa2188-04.pdf>
* Sci-kit image = <https://scikit-image.org/docs/stable/api/skimage.registration.html#skimage.registration.phase_cross_correlation> and <http://boutigny.free.fr/Astronomie/AstroSources/Kuglin-Hines.pdf>
* In general, seems like astropy and sci-kit image produce similar results, though astropy might be slightly better for astronomy (?). Depends on exact image, generally good to try both bc they work slightly differently.

***Conclusions***: Phase-based cross-correlations work just as well with the caveat that you can only use them if your images have a well-characterized feature, background, noise, threshold, mask, etc. For masks, look into watershed segmentation, image erosion and dilation, and image skeletons.

Further reading in literature…

* General ideas at AGU (for more background):
  + <https://cpaess.ucar.edu/sites/default/files/meetings/2021-punch-2/gallardo-aug2021.pdf>
  + <https://punch.space.swri.edu/files/Colaninno_AGU_2020_Poster_OpticalFlow.pdf>
  + and <https://agupubs.onlinelibrary.wiley.com/doi/epdf/10.1029/2019JE005974> (saturn clouds)
  + on ryugu’s rotation <https://arxiv.org/ftp/arxiv/papers/2112/2112.09404.pdf>
* Physics/math developments to make sense of math
  + Another way is by considering flows to be potential and incompressible (e.g. pg 79 of <http://roysouvik2.github.io/souvik_mphil.pdf>)
  + Another way is by adding a source/sink term: <https://agupubs.onlinelibrary.wiley.com/doi/full/10.1029/2019JE005974>
  + And yet another github with matlab and tif files from WMU: <https://github.com/Tianshu-Liu/OpenOpticalFlow>
* Smoke and methane plumes (most similar/advanced):
  + <https://arxiv.org/pdf/1609.02001.pdf> and <https://purehost.bath.ac.uk/ws/portalfiles/portal/187951477/Thesis_Da_Chen.pdf>
  + Mapping using sigma clip masks <https://amt.copernicus.org/articles/15/1657/2022/> ‘
  + “Machine vision” or computer vision + machine learning = <https://arxiv.org/pdf/1904.08500.pdf> and <https://mountainscholar.org/bitstream/handle/10217/208462/Martinez_colostate_0053N_15971.pdf?sequence=1&isAllowed=n>
  + [**https://amt.copernicus.org/preprints/amt-2017-185/amt-2017-185-manuscript-version3.pdf**](https://amt.copernicus.org/preprints/amt-2017-185/amt-2017-185-manuscript-version3.pdf)
* Bubbles (more taking advantage of generally uniform shapes and morphological based pre-processing…but relates to the smoke and methane plumes above)
  + “An integrative image measurement technique for dense bubbly flows with a wide size distribution” by Karn 2015 <https://www.sciencedirect.com/science/article/abs/pii/S0009250914005405>
  + <https://www.mdpi.com/2073-4441/11/11/2320/htm>
  + <https://www.nature.com/articles/s41598-022-16145-y>
* Astronomy/planetary (mostly for solar astronomy, sunspots):
  + Smoothed and multiscale optical flow for solar astronomy, including an index showing their PDE <https://www.aanda.org/articles/aa/pdf/2007/12/aa5553-06.pdf> (probably most relevant astronomical implementation I’ve found)
  + Tracking transverse velocity on Sun with so called “affine velocity estimators” <https://articles.adsabs.harvard.edu/pdf/2009JKAS...42...61C> (and a pretty good way to present the plot in their figure 1!)
  + <https://iopscience.iop.org/article/10.1086/592761/pdf> (a small review of existing methods following that 2007 A&A work),
  + Relevant to the PUNCH powerpoint under the AGU section is <https://arxiv.org/pdf/2206.09640.pdf> by Huw Morgan in 2022
  + Also relevant to the PUNCH powerpoint but is a simpler view originally by Robin Colaninno <https://iopscience.iop.org/article/10.1086/507943/pdf>  (uniformly expanding solar wind), as well as her version with stereo viewingfoggy <https://iopscience.iop.org/article/10.1088/0004-637X/698/1/852/pdf>
  + New method for tracking the movement of ionospheric plasma by Benton and Mitchell 2012
  + 2023 method of tracking with solar images using a trained neural net <https://arxiv.org/pdf/2304.03909.pdf>
  + randomly about satellite tracking: <https://arxiv.org/pdf/2204.07025.pdf>
  + Solar differential rotation: <https://iopscience.iop.org/article/10.3847/1538-4365/ab63d7/pdf> and <https://iopscience.iop.org/article/10.3847/1538-4365/abc702/pdf>
  + <https://ui.adsabs.harvard.edu/abs/2022ApJ...927..100K/abstract> (Joel Kastner at RIT did a very brief thing with planetary nebulae, but it’s not very in depth)
  + Tracking solar granulation <https://www.aanda.org/articles/aa/pdf/2013/07/aa21628-13.pdf>
  + neural net tracking of granular structures, plasma flows, sunspots on the Sun <https://www.frontiersin.org/articles/10.3389/fspas.2020.00025/full>
* Cloud and meteorological tracking (very similar techniques but applied to movies)
  + An Adaptive Tracking Algorithm for Convection in Simulated and Remote Sensing Data April 2021 (<https://github.com/openradar/TINT> TINT inspired by TITAN) or see paper at <https://journals.ametsoc.org/view/journals/apme/60/4/JAMC-D-20-0119.1.xml>
  + Tobac <https://gmd.copernicus.org/preprints/gmd-2019-105/gmd-2019-105.pdf> or <https://fsenf-tobac-tutorials.readthedocs.io/en/latest/docs/RunningTutorials.html> and <https://github.com/tobac-project/tobac/blob/main/tobac/feature_detection.py>
  + An Automatic Tracking and Recognition Algorithm for Thunderstorm Cloud-Cluster (TRACER)
  + First Observations of Tracking Clouds Using Scanning ARM Cloud Radars
  + something randomly related about ocean currents: <https://pdfs.semanticscholar.org/0041/3ece76cd44ca0afcf12700cdf0edf0b15aa5.pdf> and <https://rslab.ut.ac.ir/documents/81960329/82017906/Spatiotemporal%20monitoring%20of%20upwelled%20water%20motions%20using%20OF%20method%20in%20the%20Eastern%20Coasts%20of%20Caspian%20Sea.pdf>
* Plasmas/fusion (mostly cross correlations):
  + <https://diginole.lib.fsu.edu/islandora/object/fsu:777110/datastream/PDF/view> (mostly finding boundaries)
  + <https://www.osti.gov/pages/servlets/purl/1818872> (novel 2d velocity estimation method for large transient events in plasmas by Lampert 2021)
  + <https://pure.mpg.de/rest/items/item_2140663_1/component/file_2140662/content> (application of optical flow method for imaging diagnostic in JET by Craciunescu 2008?)
  + <https://github.com/harryh5427/GPI-blob-tracking/tree/master/code> (tracking blobs with feature ID and contour drawing but only applied to movies…)
* Biological
  + Tried asking people at RIT. They mostly do medical imaging, but didn’t give much for replies… <http://www.mackenziemathislab.org/deeplabcut> (more of a biological example that uses machine learning)
  + For another biological example that’s qualitative see <http://www.wgilpin.com/flowtrace_docs/>
  + Using optical flow in “image segmentation” of cells (hard to tell if an appropriate use of the historical term “segmentation”, but it conveys the point) <https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0261763#sec011>
* Pure neural nets and machine Learning
  + Trained on fluid flows <https://github.com/Panpanx/flow>
  + Access to all the possible PyTorch models and training sets <https://ezflow.readthedocs.io/en/latest/index.html> (can also see <https://ptlflow.readthedocs.io/en/latest/index.html> for a more basic example)
  + A default pytorch example used for videos <https://pytorch.org/vision/stable/auto_examples/plot_optical_flow.html#sphx-glr-download-auto-examples-plot-optical-flow-py>
  + A python code for a more specific example with “dense foggy scenes” <https://github.com/wending94/Optical-Flow-Estimation-in-Dense-Foggy-Scenes-with-Domain-Adaptive-Networks> or else see random paper about imaging in the rain: <https://arxiv.org/pdf/1704.05239.pdf>
* Other
  + There is a case of trying to do multi-wavelength versions at <https://www.cs.princeton.edu/~ravian/publications/SPIE_6543.pdf> if of interest
  + I found out TrackPy, bubbles, and grains have overlaps but seems not set up at all for this.
  + Couldn't find much out from LLE people.
  + Next stop was going to be asking around the EES dept. Maybe someone who does cloud-related things, Lee Murray? Didn’t hear much