Chakrabarti et al., "Low-level Vision by Consensus in a Spatial Hierarchy of Regions," CVPR 2015

$$d[n], \{I_i\}, \{\theta_i\} = \arg\min_{\{I_i\}} \sum_{i} \tau_i + \sum_{i \in \{\theta_i\}} \sum_{n} \left( \theta_i[n,1] - d_i[n] \right)^2 + \lambda \left( \theta_i[n,1] - d[n] \right)^2$$

- Last time: imposing planarity by considering a dense multi-scale set of overlapping patches, and reasoning about which of them are planar.
- Turns out, these estimates do better than SLIC + MRF particle belief-propagation, starting from the same initial disparity map $d_0$, while also being faster (6s vs 5mins).
- But in addition to $d[n]$, we can also look at the values of $\{I_i\}$: which patches are planar.
- For each $n$, consider the set of planar patches that include it: $J_n = \{P_i\}$ such that $n \in P_i$ and $I_i = 1$.
- The size of $J_n$ can be used as a confidence metric: this is the area we smoothed over to get $d[n]$.
Standard Rectified Stereo

Arbitrary camera pair

Cameras differ only by translation in the x direction

What if the two cameras are translated in pretty much the viewing direction?

For example, you don't have a camera pair but two images from a camera moving forward?

You can use a homography but there's very little overlap between the new and old views.

Standard Rectified Stereo

What if you have more than 2 cameras?

Rectification is pair-wise: you can't rectify all three together

Standard Rectified Stereo

Cameras differ only by translation in the x direction
General case: compute optical flow between two image frames \( I_x, y \) and \( I_{x+1}, y \).

- Find flow field \( u(x, y), v(x, y) \) so that \( I_x(x, y) \) matches \( I_{x+1}(x + u(x, y), y + v(x, y)) \).
- Consider the case where the flow is due to movement of the camera, but the scene is static.
  - Camera moves with respect to the scene, but objects in scene don't move with respect to each other.

This is an epipolar system. You can add additional constraints the following way:

- First find a sparse set of matches (more on this later).
- Then, estimate fundamental matrix \( F \) such that \( p F p^{+1} = 0 \) is a good fit for the pairs homogeneous co-ordinates \((x_2, p_{+-1})\) of these matches.
- Now your 2D flow-search problem turns into a 1-D search (don't rectify, use plane-sweep, search along a different line for each \((x, y)\)).
  - You are adding the constraint that: \( [x + u(x, y), y + v(x, y), 1]^T F [x, y, 1]^T = 0 \)
- Once you get a flow-field (and \( F \)) this way, does this give you depth?

NO! Because your cameras aren't calibrated.
**MONOCULAR FLOW**

- Once you get a flow-field (and ) this way, does this give you depth?
- NO! Because your cameras aren’t calibrated.
- If camera translates by \([t_x, t_y, t_z]\) or by \([2t_x, 2t_y, 2t_z]\), you will get the same .
- Your flow corresponds to a disparity along each epipolar line. \(\vec{d} = t/z\). So there is a scale ambiguity because you don’t know \(t\).
- Your camera could be moving twice as fast but your scene could also be twice as far away. There is a global scaling in the world that you can’t resolve.
- Nevertheless, you can use this to get a more accurate flow (subject of homework paper review).

Basic idea: estimate \(F\), assume \([t_x, t_y, t_z]\) = 1 (you know direction of translation, but not magnitude).

- Build a cost-volume \(C[x, y, z]\)
- \(z\) here corresponds to true \(z’\) as \(z’ = z[t_x, t_y, t_z]\)
- But irrespective, the planarity model \(1/z = \alpha x + \beta y + \gamma\) holds. Use our stereo tricks!

**MONOCULAR FLOW**

But what if in the scene are moving?

- If there is only one object in the scene, and it is moving “rigidly”
  - We’re back to an epipolar system.
  - Doesn’t matter if camera moves relative to object or object moves relative to camera.
- What if we have multiple objects moving in different ways?
  - Option 1: Assume that movement is small: most of the movement is from camera.
    - Require that the true flow is close to the epipolar lines instead of exactly on them.
    - Penalize \(\|x + u(x, y), y + v(x, y), 1\|_T F \|_{[x, y, 1]}^2\) instead of saying it is exactly 0.

**SCENE FLOW**

- What if you had a stereo pair of cameras that was moving?
  - So you have \(t_x^{left}, t_x^{right}\) and \(t_y^{left}, t_y^{right}\).
  - Since you have left-right calibration, you can get absolute depth.
  - You can then also turn the flow-field on the image plane, to actual 3D motion flow.
    - \((x, y) \rightarrow (x’, y’)\) corresponds to \((X, Y, Z)\) moved to \((X’, Y’, Z’)\).
  - Aggregate information from matching between both left-right images and consecutive frames \(t\) and \(t+1\).

SCENE FLOW

Vogel et al., "3D scene flow estimation with a piecewise rigid scene model," IJCV 2015.

OPTICAL FLOW

- But all of this processing still requires computing correspondences.
  - Even for the simplest case of monocular video with no moving objects, you need some matches to estimate the common \( F \).
- Optical flow methods we talked about in 559A assumed "infinitesimal" motion
  - Lucas Kanade, Horn Schunck
  - Based on Taylor series expansion around \((x, y)\)
- In these cases, the actual flow values \( u, v \) can have large magnitudes.
- So at the initial stage (prior to epipolar reasoning), each \((x, y)\) in frame \( t \) can match to a large number of possible locations in frame \( t + 1 \).
- Pretty expensive to build a cost volume of \( C[x, y, x', y'] \)

MATCH SEARCH FOR OPTICAL FLOW


- But before that, let’s talk about the PatchMatch algorithm.
- The goal is to find a "flow" field \( u(x, y), v(x, y) \) between images \( I \) and \( J \) which minimizes

\[
\sum_{x,y} D(u(x, y), v(x, y), x, y) = \sum_{x'=x-\frac{P}{2}}^{x'=x+\frac{P}{2}} \sum_{y'=y-\frac{P}{2}}^{y'=y+\frac{P}{2}} ||I[x', y'] - I[x + u, y + v]||^2
\]

- Basically, find nearest neighbors for all overlapping \( P \times P \) matches, in terms of euclidean distance in intensity space.
  - Can also use other distance metrics.
- Note that no need for one to one correspondence. No expectation that \( u, v \) is small. Doesn’t even need to correspond to actual motion.
- In fact, the original PatchMatch paper was a graphics paper—used for image retargetting, inpainting, texture synthesis, ...
**MATCH SEARCH FOR OPTICAL FLOW**


- **PatchMatch Algorithm**
  
  Works well in practice (at minimizing euclidean distance)!

  - Note that looking at neighbors is not a smoothness 'constraint' like in MRF models. We are still making decisions based on the euclidean distance metric.
  
  - Instead, it is way of getting to better results without doing an exhaustive search.
  
    - Example of a randomized algorithm... 

  - If you apply PatchMatch to a pair of images, this gives you a flow-field where matching pixels will have similar appearance. 

  - This is enough if you want to closeness in appearance is all you care about. But these aren't accurate as motion estimates. 

    - Because, this doesn't enforce coherence in \( u(x, y), v(x, y) \)—only minimizes intensity difference. 

      - And we know that in many smooth regions, multiple matches will have small intensity difference.

**MATCH SEARCH FOR OPTICAL FLOW**


Key Insight: Apply patch-match in a coarse to fine way.

- Build image pyramid. 

- Start at coarsest level apply patch-match. 

- Initialize next level with those flow values. 

- Restrict search to a fixed radius around matches from the coarse level. 

  Restriction = we are minimizing a different cost

**MATCH SEARCH FOR OPTICAL FLOW**


- These constrained nearest neighbors will give you pretty good motion estimates. 

- But you’ll still get a fraction of bad matches: where estimated flow is very different from true flow. 

**Detect and Remove Outliers**

- Compute flow this way between \( I_t \rightarrow I_{t+1} \) and separately \( I_{t+1} \rightarrow I_t \). 

- Throw away flow values that don’t complete the cycle: 
  \( f_{t+1}(I_t(x, y)) + f_{t+1}(x, y) \neq -f_t(x, y) \) 

- Now you are left with fewer matches, but you are sure most of them are accurate.

Coarse to fine: Colors visualize \( u(x, y), v(x, y) \) values.
Dense Interpolation
- Solve energy minimization problem which says:
  - Output flow should be close to input sparse matches at locations they are available.
  - Neighboring pixels should have similar flow (smooth or explained by same ‘affine’ model)
  - Discontinuities should align with reference frame intensity edges (by lowering the smoothness weights at those locations).

Energy Minimization
- Further refine interpolated flow-field by going back and looking at original image pair.
- Corresponds to minimizing Lucas Kandae / Horn Schunck type energy
  but now with at Taylor expansion around the input flow field, instead of \((u = 0, v = 0)\).