Optical and Scene Flow

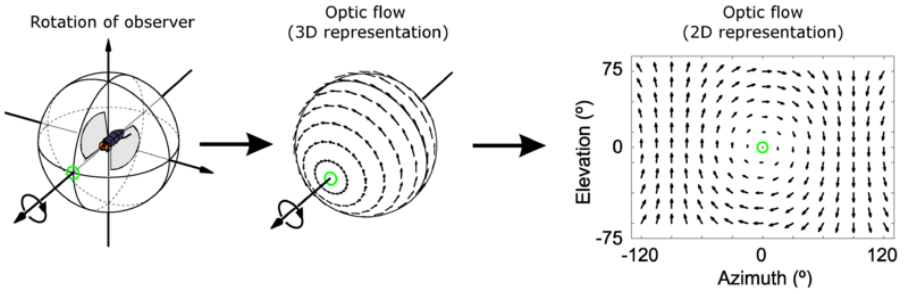
Bryan Chiang and Jeannette Bohg

February 14, 2022

1 Overview

Given a video, **optical flow** is defined as a 2D vector field describing the *apparent* movement of each pixel due to relative motion between the camera (observer) and the scene (objects, surfaces, edges). The camera or the scene or both may be moving. Figure 1 shows a fly rotating in a counter-clockwise direction (from the fly's point-of-view). Although the scene is static, the 2D optical flow of apparent motion indicates a rotation in the opposite (clockwise) direction around the origin.

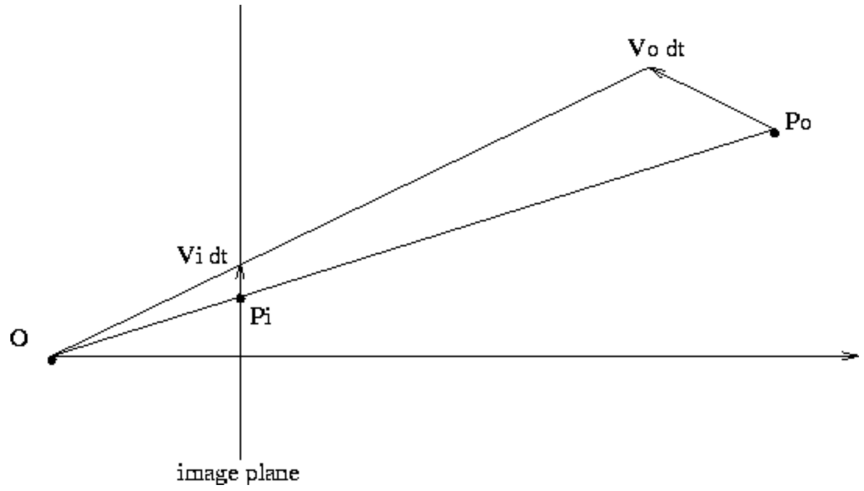
Figure 1: Optical flow example [3].



1.1 Motion field

Optical flow is not to be confused with the **motion field**, a 2D vector field describing the projection of 3D motion vectors for points in the scene onto the image plane of the observer. Figure 2 illustrates the motion field in a simple 2D case (imagine viewing a 3D scene from a top-down perspective). The 2D object point $\mathbf{P}_o$ is projected to a a1D point $\mathbf{P}_i$ in the image plane as viewed by observer O. If the object point $\mathbf{P}_o$ is displaced by $\mathbf{V}_o \cdot dt$ (called the motion vector), the corresponding projected 1D point moves by $\mathbf{V}_i \cdot dt$. The 1D motion field here consists of all velocity values $\mathbf{V}_i$ for all i located in the image plane.

Figure 2: Example of a motion field for a 2D scene [5].



Generalizing to a 3D scene, the motion field for pixel (x, y) is given by

$$\begin{pmatrix} u \\ \\ v \end{pmatrix} = \begin{pmatrix} \frac{dx}{dt} \\ \\ \frac{dy}{dt} \end{pmatrix} = \mathbf{M}\mathbf{x}' \tag{1}$$

where $\mathbf{x}' = [\frac{dx}{dt}, \frac{dy}{dt}, \frac{dz}{dt}]^T$ represents the motion of a 3D point and $\mathbf{M} \in \mathbb{R}^{2\times 3}$ contains the partial derivatives of pixel displacement with respect to the 3D point locations.

The motion field is an ideal 2D representation of 3D motion as projected onto the image plane. It is the "ground truth" that we cannot observe directly; we can only estimate the optical flow (apparent motion) from our noisy observations (video). It is important to note that the optical flow is not always the same as the motion field. For instance, a uniform rotating sphere with a fixed light source has no optical flow but a non-zero motion field. In contrast, a fixed uniform sphere with a light source moving around it has a non-zero optical flow but a zero motion field. These two cases are illustrated in Figure 3.

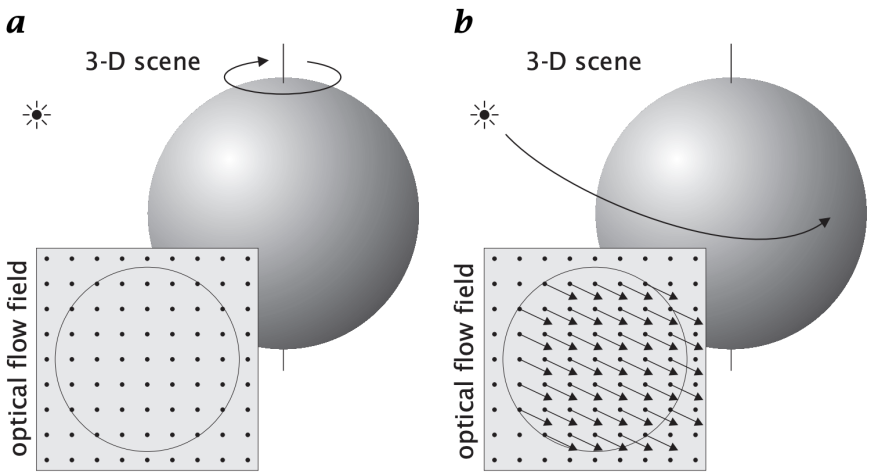


Figure 3: Physical vs. optical correspondence [4].

2 Computing the optical flow

We define a video as an ordered sequence of frames captured over time. $I(x, y, t)$, a function of both space and time, represents the intensity of pixel (x, y) in the frame at time t. In dense optical flow, at every time t and for every pixel (x, y), we want to compute the apparent velocity of the pixel in both the x-axis and y-axis, given by $u(x, y, t) = \frac{\Delta x}{\Delta t}$ and $v(x, y, t) = \frac{\Delta y}{\Delta t}$, respectively. The optical flow vector for each pixel is then given as $\mathbf{u} = [u, v]^T$. In the following sections, we describe the Lucas-Kanade method, which uses a semi-local approach to independently solve for $\mathbf{u}$ different pixel patches using least-squares.

From the **brightness constancy assumption**, we can assume that the apparent intensity in the image plane for the same object does not change across different frames. This is represented by Equation (2) for a pixel that moved Δx and Δy in the x and y directions between times t to $t + \Delta t$.

$$I(x, y, t) = I(x + \Delta x, y + \Delta y, t + \Delta t) \tag{2}$$

One common simplification is to use $\Delta t = 1$ (consecutive frames), such that the velocities are equivalent to the displacements $u = \Delta x$ and $v = \Delta y$. We can then obtain $I(x, y, t) = I(x + u, y + u, t + 1)$ as in the lecture slides.

Next, by the **small motion assumption**, we assume the motion $(\Delta t, \Delta y)$ is small from frame to frame. This allows us to linearize I with a first-order Taylor series expansion, illustrated by Equation (3).

$$
\begin{aligned}
I(x+\Delta x, y+\Delta y, t+\Delta t) &= I(x,y,t) + \frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t + \dots \\
&\approx I(x,y,t) + \frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t
\end{aligned} \tag{3}
$$

The ... represents the higher-order terms in the Taylor series expansion which we subsequently truncate out in the next line. Substituting the result from Equation (3) into Equation (2), we arrive at the optical flow constraint equation:

$$
\begin{aligned}
0 &= \frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t \\
&= \frac{\partial I}{\partial x}\frac{\Delta x}{\Delta t} + \frac{\partial I}{\partial y}\frac{\Delta y}{\Delta t} + \frac{\partial I}{\partial t} \\
&= I_x u + I_y v + I_t
\end{aligned} \tag{4}
$$

I_x, I_y, I_t are short-hand for the two spatial derivatives and time derivative, respectively.

$$
\begin{aligned}
-I_t &= I_x u + I_y v \\
&= \nabla I^T \mathbf{u} \\
&= \nabla I \cdot \vec{\mathbf{u}}
\end{aligned} \tag{5}
$$

We recognize this as linear system in the form of $\mathbf{A}\mathbf{x} = \mathbf{b}$. $\nabla I = [I_x, I_y]^T \in \mathbb{R}^{2\times 1}$ represents the spatial gradient of intensity, and $\vec{\mathbf{u}} \in \mathbb{R}^{2\times 1}$ is the flow vector we want to solve for.

However, since ∇I is a fat matrix, this is an under-constrained system since we only have a single constraint equation for our two unknowns u, v. This constraint only takes us from two to one degree of freedom; the set of (u, v) solutions must lie along a line given by Equation (5) and illustrated in Figure 4. Specifically, the optical flow constraint can only give us the **normal flow**: the component of $\mathbf{u}$ along the direction of the spatial gradient ∇I. Figure 5 visualizes the spatial image gradient in the x and y direction of an example image. You can see that the high gradient magnitudes correspond to images edges.

Recall that the dot product between two vectors $\mathbf{a}$ and $\mathbf{b}$ is $\mathbf{a}\cdot\mathbf{b} = \|\mathbf{a}\|\|\mathbf{b}\|\cos(\theta)$, where θ is the angle between the two vectors. The projection of $\mathbf{a}$ on $\mathbf{b}$ can then be obtained by dividing both sides by $\|\mathbf{b}\|$, illustrated in Figure 6.

Thus, Equation (6) shows how the optical flow constraint gives us the projection of $\mathbf{u}$ on ∇I, which is the normal flow.

$$
\frac{\nabla I}{\|\nabla I\|} \cdot \mathbf{u} = \frac{-I_t}{\|\nabla I\|} \tag{6}
$$

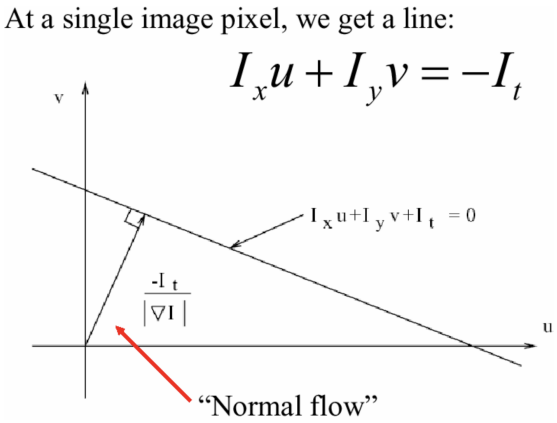


Figure 4: Solution set for (u, v) as a line in the form of $y = mx + b$.

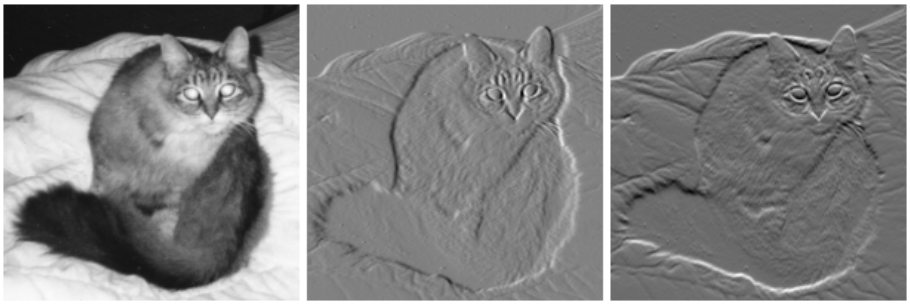


Figure 5: On the left, an intensity image of a cat. In the center, a gradient image in the x direction measuring horizontal change in intensity. On the right, a gradient image in the y direction measuring vertical change in intensity. Gray pixels have a small gradient; black or white pixels have a large gradient [1].

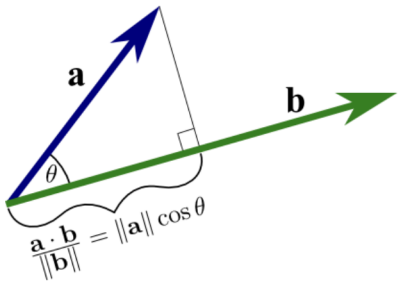


Figure 6: Dot product projection [2].

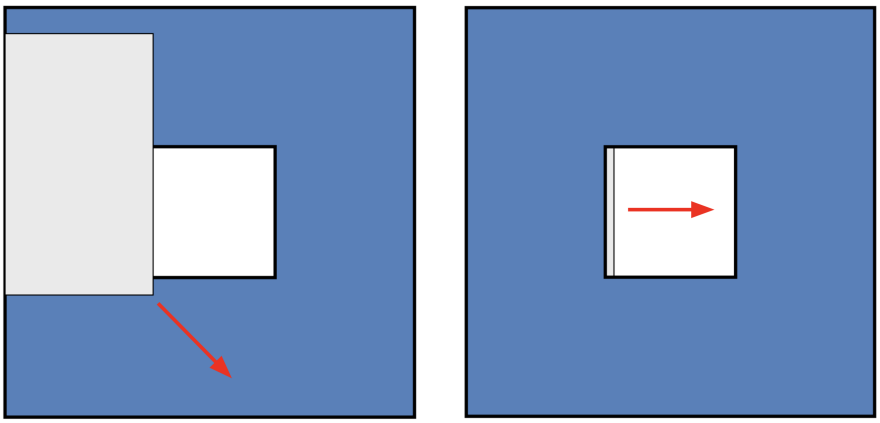


Figure 7: Visual example of the aperture problem.

Figure 4 plots the solution space for $\mathbf{u}$. However, we can also visualize the image space on top of this. The normal flow (solution space) is in the same direction of the spatial gradient of the image intensity at a given pixel. The solution set (the line) has the same direction as the edge in the image space since it is orthogonal to the spatial gradient.

Thus, while we know the component of $\mathbf{u}$ in direction of ∇I (normal flow), we do not know the component of $\mathbf{u}$ along the edge. This is represented by the solution set, which shows all the possible values this component could have. To find a possible value for $\mathbf{u}$, we first travel in the direction of the spatial gradient by a distance equivalent to the magnitude of the normal flow, and then we can travel any (unknown) distance in the direction of the edge, which represents our one degree of freedom.

This issue where we do not know the magnitude of the pixel movement in the edge direction is also known as the **aperture problem**, illustrated in Figure 7. The blue square is opaque and has a square cutout at the center that forms a small aperture. The light gray rectangle is our moving object. On the left, the gray rectangle is in front of the blue square. On the right, the gray rectangle is behind the blue square. In both cases, the rectangle moves down and to the right (at the same time). The red arrows indicate the perceived direction of movement in both cases. In the first case on the left, since the rectangle is not blocked by the blue square, we can clearly perceive its true direction of movement. However, for the case on the right, the rectangle is behind the blue square and is thus occluded: the movement of the rectangle is in the same direction as on the left, but we only perceive a movement directly to the right. This aligns with what we derived earlier. We only know how much the rectangle moved in the direction of the spatial gradient (here the x-axis), but not how much the it traveled in the direction of the image edge (here the y-axis).

To mitigate this problem, we can use the **spatial smoothness assumption**:

neighboring points belong to the same surface in the scene, and thus share the same optical flow $\mathbf{u}$. If we define an $N \times N$ neighborhood around the current pixel, we end up with a constraint $\nabla I(\mathbf{p}_i)^T \mathbf{u} = -I_t(\mathbf{p}_i)$ for each of the N^2 pixels, where $p_i = [x_i, y_i]^T$ is the location of the $i-$th pixel. Assuming $N^2 > 2$, we now have the following system of equations:

$$\mathbf{A}\mathbf{u} = \mathbf{b}$$
$$\begin{pmatrix} I_x(\mathbf{p}_1) & I_y(\mathbf{p}_1) \\ \vdots & \vdots \\ I_x(\mathbf{p}_{N^2}) & I_y(\mathbf{p}_{N^2}) \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = - \begin{pmatrix} I_t(\mathbf{p}_1) \\ \vdots \\ I_t(\mathbf{p}_{N^2}) \end{pmatrix} \tag{7}$$

We then minimize the error $\|\mathbf{A}\mathbf{u} - \mathbf{b}\|$ to find the least-squares solution of this overdetermined system, $\mathbf{u}_{\text{ls}} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}$ by multiplying both sides of Equation (7) with $\mathbf{A}^T$. We consider the solvability of this system by examining the following normal equations:

$$\mathbf{A}^T\mathbf{A}\mathbf{u} = \mathbf{A}^T\mathbf{b}$$
$$\begin{pmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{pmatrix} \mathbf{u} = - \begin{pmatrix} \sum I_x I_t \\ \sum I_y I_t \end{pmatrix} \tag{8}$$

The summations are over each pixel in the neighbourhood. For the system to be solvable, $\mathbf{A}^T\mathbf{A}$ should not be small due to the presence of noise. For example, low-texture regions have small gradient magnitudes, leading to small eigenvalues and easily corrupted $\mathbf{u}$ values. $\mathbf{A}^T\mathbf{A}$ should also be well-conditioned such that the system is robust to measurement errors since $\mathbf{A}^T\mathbf{b}$ depends on all the gradients calculated from the image intensity data. For example, at an edge all the large gradients will be pointing in the same direction, resulting in one eigenvalue that is significantly larger than the other (large condition number). A good region where $\mathbf{u}$ can be solved unambiguously is a highly textured region with directionally varying spatial structures; we will have large gradients in different directions, leading to a well-conditioned $\mathbf{A}^T\mathbf{A}$.

To more intuitively see why directionally varying structures helps us solve for $\mathbf{u}$, consider Figure 8. We have two cases here. The gray circle represents our pixel neighborhood that we are optimizing over. The dashed line represents the object at a previous time step, and and the solid line represents the object at the current time step. This depicts the true movement of the object. In (c), we see the aperture problem at play again. We have a single normal flow vector (in green), so there are numerous possible solutions for $\mathbf{u}$ (in blue) along the constraint line (also in green). However, in (b), we have varying spatial structure in the form of a corner, which we can view as the two edges with different directions. We now have two normal flow vectors and two constraint lines (one in red, the other in green). There is only one possible solution for the flow vector $\mathbf{u}$ that satisfies both constraints lines, shown in the blue. Thus, we are incentivized to increase the neighborhood size N to increase the chances of including varying spatial structure. However, there is a tradeoff as we might

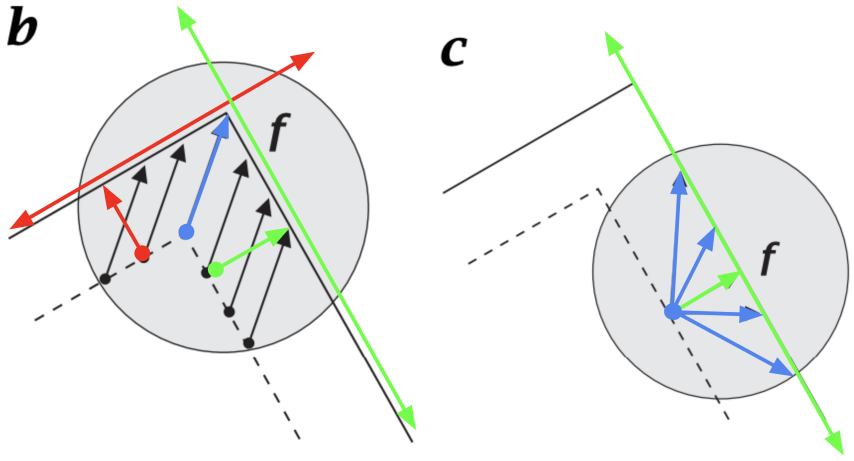


Figure 8: (b) shows the importance of having varying spatial structure in light of the aperture problem depicted in (c).

also cross motion boundaries to include pixels that belong to other surfaces and therefore it becomes more and more likely that $\mathbf{u}$ is not constant across the entire neighborhood.

In practice, Lucas-Kanade in this traditional formulation is not robust due to large camera motions, occlusions, and violations of the aforementioned set of assumption.

References

[1] https://commons.wikimedia.org/w/index.php?curid=10588443 By Njw000 Own work, Public Domain. Image gadient.

[2] Nykamp DQ. The dot product. `https://mathinsight.org/dot_product`, 2021. Accessed: 2022-01-31.

[3] Stephen J Huston and Holger G Krapp. Visuomotor transformation in the fly gaze stabilization system. *PLoS biology*, 6(7):e173, 2008.

[4] Bernd Jähne, Horst Haussecker, and Peter Geissler. *Handbook of computer vision and applications*, volume 2. Citeseer, 1999.

[5] Robyn Owens. Computer vision it412. `https://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/OWENS/LECT12/node3.html`, 1997. Accessed: 2022-01-31.