6.801/6.866: Machine Vision, Lecture 4

Professor Berthold Horn, Ryan Sander, Tadayuki Yoshitake
MIT Department of Electrical Engineering and Computer Science
Fall 2020

These lecture summaries are designed to be a review of the lecture. Though I do my best to include all main topics from the lecture, the lectures will have more elaborated explanations than these notes.

1 Lecture 4: Fixed Optical Flow, Optical Mouse, Constant Brightness Assumption, Closed Form Solution

1.1 Review

Let's frame the topics today by briefly reviewing some concepts we've covered in the previous lectures. Feel free to skip this section if you already feel comfortable with the material.

- Image formation
  - Where in the image? Recall **perspective projection**:

    $$\frac{x}{f} = \frac{X}{Z}, \frac{y}{f} = \frac{Y}{Z} \tag{1}$$

    Differentiating this expression gives:

    $$\frac{u}{f} = \frac{U}{Z} - \frac{X}{Z}\frac{W}{Z}, \frac{v}{f} = \frac{V}{Z} - \frac{Y}{Z}\frac{W}{Z} \tag{2}$$

    From these, we can find the **Focus of Expansion (FOE)**, or, more intuitively: "The point in the image toward which you are moving."

    How long until we reach this point? This is given by **Time to Contact (TTC)**:

    $$\text{Time to Contact} = \frac{Z}{W} = \frac{1}{C} \tag{3}$$

  - How bright in the image? For this, let us consider an image solid, where the brightness function is parameterized by x, y, and t: $E(x, y, t)$.

1.1.1 Constant Brightness Assumption Review with Generalized Isophotes

Recall the **constant brightness** assumption, which says that the **total derivative** of brightness with respect to time is zero: $\frac{dE(x,y,t)}{dt} = 0$. By chain rule we obtain the **BCCE**:

$$\frac{dx}{dt}\frac{\partial E}{\partial x} + \frac{dy}{dt}\frac{\partial E}{\partial y} + \frac{\partial E}{\partial t} = 0 \tag{4}$$

Recall our variables: $u \triangleq \frac{dx}{dt}, v \triangleq \frac{dy}{dt}$. Then BCCE rewritten in the standard notation we've been using:

$$uE_x + yE_y + E_t = 0 \tag{5}$$

Recall our method of using **least-squares regression** to solve for optimal values of u, v that minimize the total computed sum of the LHS of the BCCE over the entire image (note that integrals become discrete in the presence of discretized pixels, and derivatives become differences):

$$u^*, v^* = \arg\min_{u,v} \iint (uE_x + yE_y + E_t)dxdy \tag{6}$$

The first-order conditions (FOCs) of this optimization problem give:

$$u \iint E_x^2 + v \iint E_xE_y = -\iint E_xE_t \tag{7}$$
$$u \iint E_yE_x + v \iint E_y^2 = -\iint E_yE_t \tag{8}$$

Written in matrix-vector form, our equations become:

$$\begin{bmatrix} \iint E_x^2 & \iint E_xE_y \\ \iint E_yE_x & \iint E_y^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = -\begin{bmatrix} \iint E_xE_t \\ \iint E_yE_t \end{bmatrix}$$

(**New**) Now, to introduce a new variation on this problem, let us suppose we have the following spatial parameterization of brightness (you'll see that this brightness function creates **linear isophotes**) for **linear f**:

$$E(x, y) = f(ax + by) \tag{9}$$

If f is differentiable over the domain, then the spatial derivatives E_x and E_y can be computed as follows, using the chain rule:

- $E_x = f'(ax + by)a$
- $E_y = f'(ax + by)b$

Where f' is the derivative of this scalar-valued function (i.e, we can define the input to be $z = ax + by$, and the derivative f' is therefore equivalent to $\frac{df(z)}{dz}$).

Isophote Example: If $E(x, y) = ax + by + c$, for $a, b, c \in \mathbb{R}_+$, then the **isophotes** of this brightness function will be linear.

1.1.2 Time to Contact (TTC) Review

Recall the following symbols/quantities from TTC:

- $C = \frac{Z}{w}$
- TTC $= \frac{w}{Z} = \frac{1}{Z}\frac{dZ}{dt} = \frac{d}{dt}log_e(z)$, therefore we can simply take the slope of the line corresponding to the logarithm of Z to compute TTC.

Now, let's suppose that objects are moving both in the world and in the image. Let's denote s as our image coordinate and S as our world coordinate. Then:

$$\frac{s}{f} = \frac{S}{Z} \tag{10}$$

Then we can write:

$$sZ + sf = 0 \tag{11}$$

Differentiating:

$$Z\frac{ds}{dt} + s\frac{dZ}{dt} = 0 \implies \frac{\frac{ds}{dt}}{S} = \frac{\frac{dZ}{dt}}{Z} \tag{12}$$

The above relationship between derivative ratios can be interpreted as: "The change in the image's size is the same as the change in distance."

1.2 Increasing Generality of TTC Problems

Let us now consider adding some additional generality to the Time to Contact (TTC) problem. We've already visited some of these cases before:

- **Simple case**: Solving for C
- **More General case**: Solving for A, B, C
- **Even More General case**: (Motivation) What if the optical axis isn't perpendicular to the wall? What if the camera plane is tilted, e.g. $Z = aX + bY + C$ for some $a, b, C \in \mathbb{R}$? In this case, we can solve the problem **numerically** rather than through a **closed-form expression**.

Another motivating question for developing TTC methods: What if the surface is non-planar? This is a common scenario for real-world TTC systems. In this case, we have two options:

- Parameterize the geometric models of these equations with **polynomials**, rather than **planes**.
- Leave the planar solution, and look for other ways to account for errors between the modeled and true surfaces.

In practice, the second option here actually works better. The first option allows for higher modelling precision, but is less robust to local optima, and can increase the sensitivity of the parameters we find through least-squares optimization.

If you want to draw an analog to machine learning/statistics, we can think of modeling surfaces with more parameters (e.g. polynomials rather than planes) as creating a model that will **overfit** or not generalize well to the data it learns on, and create a problem with too many unknowns and not enough equations.

1.3 Multiscale and TTC

If you recall from last lecture, we saw that TTC and FOE estimation "fell apart" as we got really close. This is due to measurement sensitivity and the fact that the pixels occupy increasingly more space as we get closer and closer. This is where **multiscale** can help us - it enables us to use more coarse resolutions for these estimation problems. The implicit averaging done through downsampling allows us to "smoooth out" any measurement noise that may be present, and will consequently reduce the magnitide of pixel brightness gradients.

Additionally, multiscale is computationally-efficient: Using the infinite geometric series, we can see that downsampling/downscaling by a factor of 2 each time and storing all of these smaller image representations requires only 33% more stored data than the full size image itself:

$$\sum_{n=0}^{\infty}((\frac{1}{2})^2)^n = \frac{1}{1-\frac{1}{4}} = \frac{4}{3} = 1 + \frac{1}{3} \tag{13}$$

More generally, for any downsampling factor $r \in \mathbb{N}$, we only add $\frac{1}{r^2-1} \times 100\%$ amount of additional data:

$$\sum_{n=0}^{\infty}(\frac{1}{r^2})^n = \frac{1}{1-\frac{1}{r^2}} = \frac{r^2}{r^2-1} = \frac{(r^2-1)+1}{r^2-1} = 1 + \frac{1}{r^2-1} \tag{14}$$

(Note that we have r^2 rather than r in the denominator because we are downsampling across both the x and y dimensions.)

1.3.1 Aliasing and Nyquist's Sampling Theorem

Though multiscale is great, we also have to be mindful of **aliasing**. Recall from 6.003 (or another Signals and Systems course) that aliasing causes overlap and distortion between signals in the frequency domain, and it is required that we sample at a spatial frequency that is high enough to not produce aliasing artifacts.

Nyquist's Sampling Theorem states that we must sample at twice the frequency of the highest-varying component of our image to avoid aliasing and consequently reducing spatial artifacts.

1.3.2 Applications of TTC

A few more applications of TTC:

- Airplane Wing Safety - Using TTC to make sure wings don't crash into any objects at airports.
- NASA TTC Control - TTC systems were used to ensure that NASA's payload doesn't crash into the surface of earth/other planets/moons. In this application, feedback control was achieved by setting a nominal "desired TTC" and using an amplifier, imaging, and TTC estimation to maintain this desired TTC.
- Autonomous Vehicles - e.g. a vehicle is coming out of a parking lot and approaching a bus - how do we control when/if to brake?

Let's discuss the NASA TTC Control example a little further. Using our equation for TTC:

$$\frac{Z}{W} = T \tag{15}$$

We can rewrite this as a first-order Ordinary Differential Equation (ODE):

$$\frac{Z}{\frac{dZ}{dt}} = T \implies \frac{dZ}{dt} = \frac{1}{T}Z \tag{16}$$

Since the derivative of Z is proportional to Z, the solution to this ODE will be an exponential function in time:

$$Z(t) = Z_0 e^{\frac{-t}{T}} \tag{17}$$

Where Z_0 depends on the initial conditions of the system.

This method requires that deceleration is not uniform, which is not the most energy efficient approach for solving this problem. As you can imagine, energy conservation is very important in space missions, so let's next consider a constant deceleration approach. Note that under constant deceleration, we have $\frac{d^2z}{dt^2} \triangleq a = 0$. Then we can express the first derivative of Z w.r.t. t as:

$$\frac{dZ}{dt} = at + v_0 \tag{18}$$

Where v_0 is an initial velocity determined by the boundary/initial conditions. Here we have the following boundary condition:

$$\frac{dZ}{dt} = a(t - t_0) \tag{19}$$

This boundary condition gives rise to the following solution:

$$Z = \frac{1}{2}at^2 - at_0t + c = \frac{1}{2}a(t - t_0)^2 \tag{20}$$

Therefore, the **TTC** for this example becomes:

$$T = \frac{Z}{\frac{dZ}{dt}} = \frac{\frac{1}{2}a(t - t_0)^2}{a(t - t_0)} = \frac{1}{2}(t - t_0) \tag{21}$$

1.4 Optical Flow

Motivating question: What if the motion of an image is non-constant, or it doesn't move together? We have the **Brightness Change Constraint Equation (BCCE)**, but this only introduces one constraint to solve for two variables, and thus creates an under-constrained/ill-posed problem.

How can we impose additional constraints? To do this, let us first understand how motion relates across pixels, and information that they share. Pixels don't necessarily move exactly together, but they move together in similar patterns, particularly if pixels are close to one another. We'll revisit this point in later lectures.

What Else Can We Do? One solution is to divide the images into equal-sized patches and apply the **Fixed Flow Paradigm**, as we've done with entire images before. When selecting patch size, one trade-off to be mindful of is that the smaller the patch, the more uniform the brightness patterns will be across the patch, and patches may be too uniform to detect motion (note: this is equivalent to the matrix determinants we've been looking at evaluating to zero/near-zero).

1.5 Vanishing Points (Perspective Projection)

Before we dive into what vanishing points are, let's discuss why they're useful. Vanishing points can be useful for:

- Camera calibration - has applications to robotics, autonomous vehicles, photogrammetry, etc.
- Finding relative orientation between two coordinate frames/systems

Now, let's discuss what it is. Suppose we have several parallel lines in the world, and we image them by projecting them onto the 2D image plane. Then **vanishing points** are the points in the image (or, more commonly, outside of the image) where these lines converge to in the image. To discuss these mathematically, let's first discuss about defining lines in a 3D world:

- **Vector Form**: $\mathbf{R} = \mathbf{R_0} + s\hat{\mathbf{n}}$
  Here, we can express this using our standard vector form of perspective projection:

$$\frac{1}{f}\mathbf{r} = \frac{1}{\mathbf{R}\cdot\hat{\mathbf{z}}}\mathbf{R} \tag{22}$$
$$= \frac{1}{(\mathbf{R_0} + s\hat{\mathbf{n}})\cdot\hat{\mathbf{n}}}(\mathbf{R_0} + s\hat{\mathbf{n}}) \tag{23}$$

- **Parametrically**: $(x_0 + \alpha s, y_0 + \beta s, z_0 + \gamma s)$
  Here, we can expand this to our standard Cartesian form of perspective projection to apply our transformation:

$$\frac{x}{f} = \frac{1}{Z_0 + \gamma s}(x_0 + \alpha s) \tag{24}$$
$$\frac{y}{f} = \frac{1}{Z_0 + \gamma s}(y_0 + \beta s) \tag{25}$$

To build intuition, let's consider what happens when we travel far along the lines (i.e. as s gets very large) in our parametric definition of lines:

- $\lim_{x\to\infty} \frac{x}{f} = \lim_{x\to\infty} \frac{1}{Z_0+\gamma s}(x_0 + \alpha s) = \frac{\alpha s}{\gamma s} = \frac{\alpha}{\gamma}$ **(x-coordinate)**
- $\lim_{y\to\infty} \frac{y}{f} = \lim_{y\to\infty} \frac{1}{Z_0+\gamma s}(x_0 + \beta s) = \frac{\beta s}{\gamma s} = \frac{\beta}{\gamma}$ **(x-coordinate)**

The 2D point $(\frac{\alpha}{\gamma}, \frac{\beta}{\gamma})$ is the vanishing point in the **image plane**. As we move along the line in the world, we approach this point in the image, but we will never reach it. More generally, we claim that **parallel lines in the world have the same vanishing point in the image**.

1.5.1 Applications of Vanishing Points

Let's now discuss some of these applications in greater detail.

- **Protecting Pedestrians on a Road Side**: To protect pedestrians, the camera must transform its coordinate system. This transformation can be found using **vanishing points**.
- **Camera Calibration**: One way to calibrate a camera is to solve for the **Center of Projection (COP)** in the image space, using perspective projection. Calibration is typically achieved through **calibration objects**.

1.6 Calibration Objects

Let's discuss two calibration objects: **spheres** and **cubes**:

1.6.1 Spheres

:

- If image projection is directly overhead/straight-on, the projection from the world sphere to the image plane is a circle. If it is not overhead/straight on, it is elliptic.
- Relatively easy to manufacture

1.6.2 Cube

:

- Harder to manufacture, but generally a better calibration object than a sphere.
- Cubes can be used for detecting edges, which in turn can be used to find vanishing points (since edges are lines in the world).
- Cubes have three sets of four parallel lines/edges each, and each of these sets of lines are orthogonal to the others. This implies that we will have three **vanishing points -** one for each set of parallel lines.
- For each of these sets of lines, we can pick a line that goes through the Center of Projection (COP), denoted $\mathbf{p} \in \mathbb{R}^3$ (in the world plane). We can then project the COP onto the image plane (and therefore now $\mathbf{p} \in \mathbb{R}^{\nvDash}$).
- Let us denote the **vanishing points** of the cube in the image plane as $\mathbf{a}, \mathbf{b}, \mathbf{c} \in \mathbb{R}^2$. Then, because of orthogonality between the different sets of lines, we have the following relations between our three vanishing points and $\mathbf{p}$:
  - $(\mathbf{p} - \mathbf{a}) \cdot (\mathbf{p} - \mathbf{b}) = 0$
  - $(\mathbf{p} - \mathbf{b}) \cdot (\mathbf{p} - \mathbf{c}) = 0$
  - $(\mathbf{p} - \mathbf{c}) \cdot (\mathbf{p} - \mathbf{a}) = 0$

  In other words, the difference vectors between $\mathbf{p}$ and the vanishing points are all at right angles to each other.

  To find $\mathbf{p}$, we have three equations and three unknowns. We have terms that are quadratic in $\mathbf{p}$. Using **Bézout's Theorem** (*The maximum number of solutions is the product of the polynomial order of each equation in the system of equations*), we have $(2)^3 = 8$ possible solutions for our system of equations. More generally:

  $$\text{number of solutions} = \prod_{e=1}^{E} o_e \tag{26}$$

  Whhere E is the number of equations and o_e is the polynomial order of the e^{th} equation in the system.

  This is too many equations to work with, but we can subtract these equations from one another and create a system of 3 linearly dependent equations. Or, even better, we can leave one equation in its quadratic form, and 2 in their linear form, and this maintains linear independence of this system of equations:

  - $(\mathbf{a} - \mathbf{p}) \cdot (\mathbf{c} - \mathbf{b}) = 0$
  - $(\mathbf{b} - \mathbf{p}) \cdot (\mathbf{a} - \mathbf{c}) = 0$
  - $(\mathbf{p} - \mathbf{c}) \cdot (\mathbf{p} - \mathbf{a}) = 0$

1.7 Additional Topics

We'll also briefly recap some of the topics discussed in our synchronous section today. These topics are not meant to introduce new concepts, but are designed to generalize the concepts we've discussed such that they can be adapted for a broader range of applications.

1.7.1 Generalization: Fixed Flow

The motivating example for this generalization is a **rotating optimal mouse**. We'll see that instead of just solving for our two velocity parameters u and v, we'll also need to solve for our rotational velocity, ω.

Suppose we are given the following parameterizations of velocity:

- $u = u_0 - wy$
- $v = v_0 + wx$

Note that we can also write the radial vector of x and y, as well as the angle in this 2D plane to show how this connects to rotation:

$$\mathbf{r} = (x, y) = \sqrt{x^2 + y^2} \tag{27}$$
$$\theta = \arctan 2(y, x) \tag{28}$$
$$\omega = \frac{d\theta}{dt} \tag{29}$$

With this rotation variable, we leverage the same least-squares approach as before over the entirety of the image, but now we also optimize over the variable for ω:

$$u_0^*, v_0^*, \omega^* = \arg\min_{u_0, v_0, \omega} \{J(u_0, v_0, \omega) \triangleq \iint (u_0 E_x + v_0 E_y + wH + E_t)^2 dxdy\} \tag{30}$$

Like the other least-squares optimization problems we've encountered before, this problem can be solved by solving a system of first-order conditions (FOCs):

- $\frac{dJ(u_0,v_0,\omega)}{du_0} = 0$
- $\frac{dJ(u_0,v_0,\omega)}{dv_0} = 0$
- $\frac{dJ(u_0,v_0,\omega)}{d\omega} = 0$

1.7.2 Generalization: Time to Contact (for $U = 0, V = 0, \omega \neq 0$)

Let's now revisit TTC, but with the following parameterization: $U \neq 0, V \neq 0$, image is of a tilted plane.

For this, we can write Z as a function of the world coordinates X and Y:

$$Z = Z_0 + \frac{\partial Z}{\partial X}X + \frac{\partial Z}{\partial Y}Y \tag{31}$$
$$Z = Z_0 + pX + qY \tag{32}$$

Recall the following derivations for the image coordinate velocities u and v, which help us relate image motion in 2D to world motion in 3D:

- $\frac{u}{f} = \frac{U}{Z} - \frac{X}{Z}\frac{W}{Z}$
- $\frac{v}{f} = \frac{V}{Z} - \frac{Y}{Z}\frac{W}{Z}$

Some additional terms that are helpful when discussing these topics:

- **Motion Field**: Projection of 3D motion onto the 2D image plane.
- **Optical Flow**:
  - What we can sense
  - Describes motion in the image

We can transform this into image coordinates:

$$u = \frac{1}{2}(fU - xw), \ \ v = \frac{1}{2}(fV - yw) \tag{33}$$

Let's take $U = V = 0, u = -X\frac{w}{Z}, v = -Y\frac{W}{Z}$. Z (world coordinates) is not constant, so we can rewrite this quantity by substituting the image coordinates in for our expression for Z:

$$Z = Z_0 + px + qy \tag{34}$$
$$= Z_0 + p\frac{X}{f}Z + q\frac{Y}{f}Z \tag{35}$$

Now, we can isolate Z and solve for its closed form:

$$Z(1 - p\frac{X}{f} - q\frac{Y}{f}) = Z_0 \tag{36}$$
$$Z = \frac{Z_0}{1 - p\frac{X}{f} - q\frac{Y}{f}} \tag{37}$$

From this we can conclude that $\frac{1}{Z}$ is linear in x and y (the image coordinates, not the world coordinates). This is helpful for methods that operate on finding solutions to linear systems. If we now apply this to the **BCCE** given by $uE_x + vE_y + E_t = 0$, we can first express each of the velocities in terms of this derived expression for Z:

- $u = \frac{1}{Z_0}(1 - p\frac{X}{f} - q\frac{Y}{f})(-x\omega)$
- $v = \frac{1}{Z_0}(1 - p\frac{X}{f} - q\frac{Y}{f})(-y\omega)$

Applying these definitions to the BCCE:

$$\begin{aligned} 0 &= uE_x + vE_y + E_t && (38) \\ &= \frac{1}{Z_0}(1 - p\frac{X}{f} - q\frac{Y}{f})(-x\omega)E_x + \frac{1}{Z_0}(1 - p\frac{X}{f} - q\frac{Y}{f})(-y\omega)E_y + E_t && (39) \end{aligned}$$

Combining like terms, we can rewrite this constraint as:

$$0 = -\frac{\omega}{Z_0}(1 - p\frac{X}{f} - q\frac{Y}{f})(xE_x + yE_y) + E_t \tag{40}$$

Equivalently, dividing everything by -1:

$$0 = \frac{\omega}{Z_0}(1 - p\frac{X}{f} - q\frac{Y}{f})(xE_x + yE_y) - E_t \tag{41}$$

We can also express this with the "radial gradient" given by: $G \triangleq (xE_x + yE_y)$:

$$0 = \frac{\omega}{Z_0}(1 - p\frac{X}{f} - q\frac{Y}{f})G - E_t \tag{42}$$

For symbolic simplicity, take the following definitions:

- $R \triangleq \frac{w}{Z_0}$
- $P \triangleq -p\frac{w}{Z_0}$
- $Q \triangleq -q\frac{w}{Z_0}$

Using these definitions, the BCCE with this paramterization becomes:

$$0 = (R + Px + Qy)G - E_t \tag{43}$$

Now, we can again take our standard approach of solving these kinds of problems by applying least-squares to estimate the free variables P, Q, R over the entire continuous or discrete image space. Like other cases, this fails when the determinant of the system involving these equations is zero.