6.801/6.866: Machine Vision, Lecture 6

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 6: Photometric Stereo, Noise Gain, Error Amplification, Eigenvalues and Eigenvectors review

This lecture covers an overview of some relevant concepts from linear algebra. If you would like a refresher on some of these concepts, please check out Professor Horn's posted linear algebra review **here**.

1.1 Applications of Linear Algebra to Motion Estimation/Noise Gain

Recall our problem of noise gain, in that if we observe a noisy estimate of y and we're asked to derive x, the noise from our observed value can greatly affect our estimate of x. We've analyzed this problem in 1D, and now we're ready to generalize this to higher dimensions using eigenvalues and eigenvectors.

For the sake of brevity, the notes from the linear algebra review will not be given here, but they are in the handwritten notes for this lecture, as well as in the linear algebra handout posted above. These notes cover:

- Definitions of eigenvalues/eigenvectors
- Characteristic Polynomials
- Rewriting vectors using an eigenbasis

1.1.1 Application Revisited: Photometric Stereo

Let us now return to an application where we determine shape from shading: photometric stereo. Let us first consider the simple case where the surface we seek to reconstruct is Lambertian:

$$E \propto \cos\theta_i = \rho \hat{\mathbf{n}} \cdot \hat{\mathbf{s}}_i$$

Recall the following, to help frame the setup of this problem:

- For shape from shading problems, our goal is to estimate and reconstruct a 3D scene from a 2D image given information about the shading and surface orientation of the scene.
- In 2D, this shape from shading problem has 2 Degrees of Freedom (abbreviated in these course notes as DOF).
- Recall that we can use albedo (ρ) to actually make this problem easier - can create a problem with 3 unknowns and 3 equations. Without the use of albedo, recall that we had a quadratic constraint we had to follow, which, by **Bezout's theorem**, suggests that there are at two solutions to such a system of equations. In this case, we have 3 linear equations and 3 unknowns, so by Bezout's theorem we have only one solution, as desired.

Recall that for such a system, suppose we have three brightness measurements of the form:

1. $E_1 = \rho \hat{\mathbf{n}} \cdot \hat{\mathbf{s}}_1$ - Intuitively, look at pixel with light source located at $\mathbf{s}_1$ and take measurement E_1.
2. $E_2 = \rho \hat{\mathbf{n}} \cdot \hat{\mathbf{s}}_2$ - Intuitively, look at pixel with light source located at $\mathbf{s}_2$ and take measurement E_2.

3. $E_3 = \rho\hat{\mathbf{n}} \cdot \hat{\mathbf{s}}_3$ - Intuitively, look at pixel with light source located at $\mathbf{s}_3$ and take measurement E_3.

Let's review the linear algebraic system of equations by combining the equations above into a matrix-vector product.

$$\begin{bmatrix} -s_1^T \\ -s_2^T \\ -s_3^T \end{bmatrix} \mathbf{n} = \begin{bmatrix} E_1 \\ E_2 \\ E_3 \end{bmatrix}$$

Note the following:

- Since the matrix on the left-hand side corresponds to the three-dimensional position of the light source in the given frame of reference, we have:

$$\begin{bmatrix} -s_1^T \\ -s_2^T \\ -s_3^T \end{bmatrix} \in \mathbb{R}^{3\times 3}, \quad \begin{bmatrix} E_1 \\ E_2 \\ E_3 \end{bmatrix} \in \mathbb{R}^3$$

- $\mathbf{n} = \rho\hat{\mathbf{n}}$, i.e. we do not need to deal with the second order constraint $\hat{\mathbf{n}}^T\hat{\mathbf{n}} = 1$. This eliminates the second-order constraint from our set of equations and ensures we are able to derive a unique solutions by solving a system of only linear equations.
- Define $\mathbf{S} \triangleq \begin{bmatrix} -s_1^T \\ -s_2^T \\ -s_3^T \end{bmatrix}$

Like many other linear system of equations we encounter of the form $\mathbf{Ax} = \mathbf{b}$, we typically want to solve for $\mathbf{x}$. In this case, we want to solve for $\mathbf{n}$, which provides information about the surface orientation of our object of interest:

$$\mathbf{Sn} = \mathbf{E} \longrightarrow \mathbf{n} = \mathbf{S}^{-1}\mathbf{E}$$

Like the other "inverse" problems we have solved so far in this class, we need to determine when this problem is ill-posed, i.e. when $\mathbf{S}$ is not invertible. Recall from linear algebra that this occurs when the columns of $\mathbf{S}$ are not linearly independent/the rank of $\mathbf{S}$ is not full.

An example of when this problem is ill-posed is when the light source vectors are coplanar with one another, i.e.

$\mathbf{s}_1 + \mathbf{s}_2 + \mathbf{s}_3 = 0$ (Note that this can be verified by simply computing the vector sum of any three coplanar vectors.)

Therefore, for optimal performance and to guarantee that $\mathbf{S}$ is invertible:

- We make the vectors from the light sources to the objects ($\mathbf{s}_i$) as non-coplanar as possible.
- The best configuration for this problem is to have the vectors from the light sources to the surface be orthogonal to one another. Intuitively, consider that this configuration captures the most variation in the angle between a given surface normal and three light sources.

Example: Determining Depth of Craters on the Moon:

As it turns out, we cannot simply image the moon directly, since the plane between the earth/moon and the sun makes it such that all our vectors $\mathbf{s}_i$ will lie in the same plane/are coplanar. Thus, we cannot observe the moon's tomography from the surface of the earth.

1.2 Lambertian Objects and Brightness

Fun Fact: "Lambert's Law" was derived from a monk observing light passing through an oil spot in a piece of paper.

"Lambert's Law":

$$E_i \propto \cos\theta_i = \hat{\mathbf{n}} \cdot \hat{\mathbf{s}}_i$$

To better understand this law, let us talk more about surface orientation.

1.2.1 Surface Orientation

- For an extended surface, we can take small patches of the surface that are "approximately planar", which results in us constraining the surface to be a **2-manifold** [1] (technically, the language here is "locally planar").
- Using these "approximately planar" patches of the surface results in many unit normal vectors $\hat{\mathbf{n}}_i$ pointing in a variety of different directions for different patches in the surface.
- We can also understand surface orientation from a calculus/Taylor Series point of view. Consider $z(x+\delta x, y+\delta y)$, i.e. an estimate for the surface height at (x, y) perturbed slightly by a small value δ. The Taylor Series expansion of this is given by:

$$z(x+\delta x, y+\delta y) = z(x,y) + \delta x \frac{\partial z}{\partial x} + \delta y \frac{\partial z}{\partial y} + \cdots = z(x,y) + p\delta x + q\delta y$$

$$\text{Where } p \triangleq \frac{\partial z}{\partial x}, q \triangleq \frac{\partial z}{\partial y}$$

- The surface orientation ($p \triangleq \frac{\partial z}{\partial x}, q \triangleq \frac{\partial z}{\partial y}$) captures the gradient of the height of the surface z.
- Note that the surface unit normal $\hat{\mathbf{n}}$ is perpendicular to any two lines in the surface, e.g. tangent lines.

We can use the cross product of the two following tangent lines of this surface to compute the the unit surface normal, which describes the orientation of the surface:

1. $(\delta x, 0, p\delta x)^T = \delta x(1, 0, p)^T$
2. $(0, \delta y, q\delta x)^T = \delta y(0, 1, q)^T$

Since we seek to find the *unit* surface normal, we can remove the scaling factors out front (δx and δy). Then the cross product of these two vectors is:

$$(1,0,p)^T \times (0,1,p)^T = \det \begin{bmatrix} \hat{\mathbf{x}} & \hat{\mathbf{y}} & \hat{\mathbf{z}} \\ 1 & 0 & p \\ 0 & 1 & q \end{bmatrix} = \mathbf{n} = \begin{bmatrix} -p \\ -q \\ 1 \end{bmatrix}$$

We can now compute the unit surface normal by normalizing the vector $\mathbf{n}$:

$$\hat{\mathbf{n}} = \frac{\mathbf{n}}{||\mathbf{n}||_2} = \frac{\begin{bmatrix} -p & -q & 1 \end{bmatrix}^T}{\sqrt{p^2+q^2+1}}$$

With this surface normal computed, we can also go the opposite way to extract the individual components of the surface orientation (p and q):

$$p = \frac{-\mathbf{n} \cdot \hat{\mathbf{x}}}{\mathbf{n} \cdot \hat{\mathbf{z}}}, \quad q = \frac{-\mathbf{n} \cdot \hat{\mathbf{y}}}{\mathbf{n} \cdot \hat{\mathbf{z}}}$$

Note: Do the above equations look similar to something we have encountered before? Do they look at all like the coordinate-wise perspective projection equations?

We will leverage these (p, q)-space plots a lot, using *reflectance maps* (which we will discuss later). Now, instead of plotting in velocity space, we are plotting in surface orientation/spatial derivative space. Individual points in this plane correspond to different surface orientations of the surface.

1.2.2 Surface Orientation Isophotes/Reflectance Maps for Lambertian Surfaces

Recall the equation to derive isophotes of a Lambertian surface (since the brightness depends on the incident angle between the surface and the light source):

$$\hat{\mathbf{n}} \cdot \hat{\mathbf{s}} = E_1 \text{ Where } E_1 \text{ is a constant, scalar level of brightness}$$

If we expand our light source vector $\hat{\mathbf{s}}$, we get the following for this dot product:

$$\hat{\mathbf{n}} \cdot \hat{\mathbf{s}} = \frac{\begin{bmatrix} -p & -q & 1 \end{bmatrix}^T}{\sqrt{p^2+q^2+1}} \cdot \frac{\begin{bmatrix} -p_s & -q_s & 1 \end{bmatrix}^T}{\sqrt{p_s^2+q_s^2+1}} = E_1$$

Carrying out this dot product and squaring each side, we can derive a parametric form of these isophotes in (p, q) space (note that the light source orientation (p_s, q_s) is fixed (at least for a single measurement), and thus we can treat it as constant:

$$(1 + p_s p + q_s q)^2 = c^2(1 + p^2 + q^2)(1 + p_s^2 + q_s^2)$$

If we plot these isophotes in (p, q) space, we will see they become **conic sections**. Different isophotes will generate different curves. When we have multiple light sources, we can have intersections between these isophotes, which indicate solutions.

1.3 References

[1] Manifolds, https://en.wikipedia.org/wiki/Manifold.

6.801 / 6.866 Machine Vision
Fall 2020