Project 3 – Photometric Stereo

All characters and legal bodies appearing in this scenario are fictitious. Any resemblance to real persons, living or dead, or to existing corporations, is purely coincidental. (Having said that, I encourage attempts to draw parallels to existing scientific results)

1 Preamble

Inspired by the work you did for OptoCorp in such a short time, you decide to take some freelance projects working from your study at home. However, experimenting with X-rays is too dangerous without OptoCorp's safety precautions. Instead, you decide to try your luck in developing systems with the ability to reconstruct surfaces in a much safer region of the electro-magnetic spectrum : the visible region.

2 More Details[1]

The image *irradiance* of a point in the image plane is defined to be the power per unit area of radiant energy falling on the image plane. *Irradiance* is the proper term for image *intensity*, which is also synonym with *gray-value* and *brightness*. The relation between the *irradiance* $I(x', y')$ (which is the incoming energy) at a point (x', y') in the image plane and the *radiance* $L(x, y, z)$ (outgoing energy) radiated by the corresponding point (x, y, z) in the scene is given by

$$I(x', y') = L(x, y, z) \tag{1}$$

Two factors determine the radiance reflected by a patch of scene surface:

- The illumination falling on the patch of scene surface. This is determined by the distribution of light sources in the scene, and their relative position to the surface patch.
- The fraction of the incident illumination that is reflected by the surface patch. This is determined by the optical properties of the surface.

In general, this means that relation (1) between *radiance* and *irradiance* is a function of the direction of *incident* light and the direction of the *emitted* light relative to the surface patch. We shall restrict our attention to a particular type of surfaces known as *Lambertian*. Such a surface appears equally bright from all viewing direction and does not absorb any

[1] This exposition is adapted from [2].

incident illumination. In that case, the amount of brightness perceived from a Lambertian surface illuminated by a point light source is a function only of the direction of *incident* light.

$$I(x, y) = R(p, q) = \lambda \cos\theta \tag{2}$$

where θ is the angle between the light direction (pointing from the surface towards the light source) and the normal to the surface, and λ is some constant. $R(p, q)$ is called the *reflectance map* and is a function of the gradient of the surface at point (x, y) [2]:

$$p = \frac{\partial z}{\partial x} \qquad q = \frac{\partial z}{\partial y}$$

Another radiometric effect that we have omitted is the fact that not all incident light is reflected by the surface. This can be taken into account by adding a position dependent factor $0 \le \rho(x, y) \le 1$ to the *image irradiance equation* :

$$I(x, y) = \rho(x, y) R(p, q) \tag{3}$$

ρ is called *albedo* (from latin : *whiteness*). For the rest of the project we shall assume that the albedo is constant for the entire surfaces in question.

3 Project outline

- Introduction of basic notions in differential geometry of curves and surfaces.
- Development of some numerical tools using the calculus of variations.
- Using these tools to reconstruct a surface from a single image ("Shape From Shading").
- Reconstruction of a surface from multiple images of the same surface in different lighting conditions ("Photometric Stereo").

4 Geometry, optics, and what comes between them

Before you get to test your ideas, let's develop some geometric tools.

[2] For those who know about the mechanism of image formation, this is only true under the assumption of parallel projection (i.e., the camera ("viewer") is exactly above the surface so the scene coordinates (x, y) equal the image coordinates (x', y')).

4.1 Basic notions in differential geometry of curves and surfaces

4.1.1 differential geometry of curves

Consider a curve parametrized by $\gamma(p) \equiv (x(p), y(p))$. The tangent to the curve is given by $\mathbf{T} \equiv \dot{\gamma}(p) = (\dot{x}(p), \dot{y}(p))$. Since the parametrization is arbitrary, we can choose to parametrize the curve by its arclength,

$$s(p) = \int_0^p \|\dot{\gamma}(\tilde{p})\| d\tilde{p} \tag{4}$$

Which has some nice properties:

✍ **1.** Show that the magnitude of the tangent (i.e., the "speed" of a particle sliding along the curve) is 1 in arclength parametrization :

$$\sqrt{\langle \mathbf{T}, \mathbf{T} \rangle} = \|\dot{\gamma}(s)\| = 1 \tag{5}$$

✍ **2.** Show that the vector $\dot{\mathbf{T}}$ is parallel to the normal to the curve, i.e., $\dot{\mathbf{T}} = \kappa \mathbf{N}$, and that

$$\kappa(s) = \ddot{x}(s)\dot{y}(s) - \ddot{y}(s)\dot{x}(s)$$

(κ is called the *curvature* of the curve)

4.1.2 differential geometry of surfaces

✍ **3.** Given a surface that can be represented as the graph of a function

$$z \equiv z(x, y), \quad (x, y) \in \Omega$$

show that the (non-unit) normal to the surface at the point (x, y, z) is given by

$$\mathbf{N}(x, y) = [1, 0, z_x]^T \times [0, 1, z_y]^T = [-z_x, -z_y, 1]^T \tag{6}$$

and that an area element on the surface is given by

$$da = \|\mathbf{N}(x, y)\| dxdy = \sqrt{z_x^2 + z_y^2 + 1} dxdy \tag{7}$$

✍ **4.** Find the necessary condition for a surface with given boundary values

$$z(x,y) = z_0(x,y), \quad (x,y) \in \partial\Omega \tag{8}$$

to have minimum area by finding the Euler-Lagrange equation resulting from

$$\min_{\mathbf{z}} \int_\Omega da \tag{9}$$

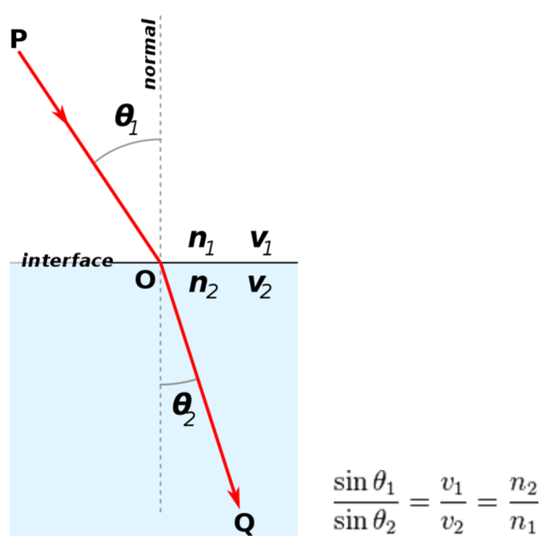
Express the condition in terms of the *mean curvature* of the surface

$$K \equiv (1 + z_x^2)z_{yy} - 2z_x z_y z_{xy} + (1 + z_y^2)z_{xx}$$

(such a surface is called a **minimal surface**).

4.2 Fermat's principle of least time

Figure 1: Snell's law : the ratio of the sines of the angles of incidence and refraction is equivalent to the ratio of phase velocities (i.e, the refractive index) in the two media. This is a particular example of *Fermat's principle of least time*.



A well known principle in geometrical optics is **Fermat's principle of least time**:

> *the path taken by a ray of light between two points is the path that can be traversed in the* ***least time***[3].

[3] As a matter of fact, light actually follows extremal paths. These paths can sometimes be the longest rather than the shortest

From this variational principle one can derive most of the results in classical geometrical optics.

We shall denote by $n(x,y)$ the refractive index of some inhomogeneous (but isotropic) medium. The velocity of light in a refractive medium is given by $v = \frac{c}{n}$, where c is the speed of light in the vacuum.

✍ **5.** From Fermat's principle

$$t^\star = \min \int_{t_0}^{t_1} dt = \min_{\gamma(p)} \int_{p_0}^{p_1} \frac{n(\gamma(p))}{c} \|\dot{\gamma}(p)\| dp \tag{10}$$

Derive the equations

$$\left(\langle \nabla n, \mathbf{N} \rangle - \kappa n \right) \mathbf{N} = 0 \tag{11}$$

Guidance: after deriving the Euler-Lagrange equations, switch to arclength parametrization. By virtue of (5), you will be able to simplify the expressions greatly.

✍ **6.** Derive Snell's law (see Fig 1) using a piecewise constant index of refraction

$$n(x,y) = \begin{cases} n_1, & y \le l \\ n_2, & y > l \end{cases}$$

Hint: can you derive a conservation law from (10) for this particular kind of $n(x,y)$?

4.3 The Brachistochrone Problem

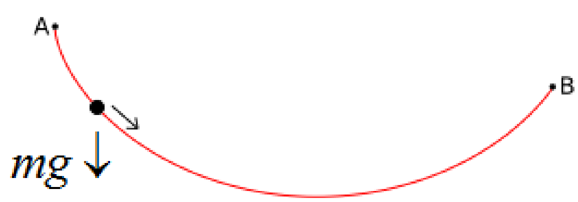


Figure 2: The brachistochrone problem : find the curve that will carry a point mass from A to B in the shortest time under the act of gravity.

The brachistochrone problem (Fig 2) is a related classical problem in mechanics: given two end points $A \equiv (x_0, y_0)$, and $B \equiv (x_1, y_1)$, find the curve that will carry a point mass from A to B in the shortest time under the act of gravity.

✍ **7.** Show that the functional for the brachistochrone problem is given by:

$$\int_{t_0}^{t_1} dt = \int_{x_0}^{x_1} \sqrt{\frac{1+y_x^2}{2gy}} dx \tag{12}$$

✍ **8.** Show that the brachistochrone curve is a *cycolid*

$$\begin{aligned} x &= \frac{1}{2}k^2(\theta - \sin\theta) \\ y &= \frac{1}{2}k^2(1-\cos\theta) \end{aligned} \tag{13}$$

Use the fact that x does not appear explicitly in (12) to derive these equations. What is k?

✍ **9.** Show that the brachistochrone problem is a particular example of *Fremat's principle*, and find the corresponding $n(x,y)$.

4.4 The eikonal equation

The E-L equations (11) are ODEs that allow us to find the shortest path between any two points in the medium, provided that these points are given in advance. From Fermat's principle, one can derive a PDE that will at once characterize the distance from a single source point (or a set of source points) to the rest of the points. First, let's denote by S the **optical path length** (OPL). By setting $c=1$ in (10), S is practically the time it takes the ray of light to travel from (x_0,y_0) to (x_1,y_1):

$$S = \min_{\gamma(p)} \int_{p_0}^{p_1} n(\gamma(p))\|\dot{\gamma}(p)\|dp \tag{14}$$

$S \equiv S(p_0,x_0,y_0,p_1,x_1,y_1)$ is a function of the end points. If the initial point (p_0,x_0,y_0) is held fixed, we can regard S as a function of the end point (p_1,x_1,y_1), and ask how does S change when we vary (x_1,y_1)[4]. Since (x_1,y_1) are our variables now, let's drop the subscript and denote them simply by (x,y). We can now compute $\Delta S = S(p,x+dx,y+dy)-S(p,x,y)$. By definition,

$$\Delta S = \delta J \equiv J(\gamma^{\star\star}) - J(\gamma^{\star})$$

where $J(\gamma^\star), J(\gamma^{\star\star})$ are the functional from (14) evaluated along the extremal curves $\gamma^\star$ (the extremal curve from (p_0,x_0,y_0) to (p_1,x_1,y_1)), and $\gamma^{\star\star}$ (the extremal curve from (p_0,x_0,y_0) to (p_1,x_1+dx_1,y_1+dy_1)). Repeating the derivation from class:

$$\delta J = \int_{\gamma^\star}\left(\frac{\partial\mathcal{L}}{\partial\gamma} - \frac{d}{dp}\frac{\partial L}{\partial\dot{\gamma}}\right)dp + \frac{\partial\mathcal{L}}{\partial\dot{\gamma}}\delta\gamma\Big|_{p_0}^{p_1}$$

[4]In general, one can also vary the independent variable p, and compute the **general variation** of a functional, from which the well-known Hamilton-Jacobi equation can be derived. See solutions.

Since $\gamma^\star$ is an extremal curve, the first term is zero (E-L equations are satisfied). The point (p_0, x_0, y_0) is held fixed, so $\left.\frac{\partial \mathcal{L}}{\partial \dot{\gamma}} \delta\gamma \right|_{p_0} = 0$. We are left with

$$\frac{\delta J}{\delta \gamma} = \nabla S = \frac{\partial \mathcal{L}}{\partial \dot{\gamma}} \tag{15}$$

✍ **10.** Show that (15) applied to (14) gives the *eikonal* equation:

$$\nabla S = n \cdot \mathbf{T} \tag{16}$$

The *eikonal* equation is a fundamental PDE in geometrical optics relating the optical path length to the properties of the medium through which the light traverses (namely, the refractive index). It can also be derived from Maxwell's equations by applying a high frequency approximation (where the rules of geometrical optics apply). This is an example of a special non-linear PDE, a static *Hamilton-Jacobi* equation. At some points the derivative of S may not exist, which requires a special notion of "solution" called a *viscosity solution*. When discretizing such equations, one has to take special care to make sure that the numerical solution converges to the correct viscosity solution as the grid is refined. In the next section, you will use one such numerical scheme.

4.4.1 Numerical solver for the eikonal equation

We will now explore a numerical solver for the eikonal equation, and test it on the brachistochrone problem.

In order to find the shortest path between two points, a two step procedure is applied:

- Solving $|\nabla S| = n$ on a 2D grid, with boundary conditions $S(x_0, y_0) = 0$.
- According to (16), the tangent to the shortest path connecting (x_0, y_0) and (x_1, y_1) is parallel to ∇S. Thus, we can trace this path by performing gradient descent on $S(x, y)$ starting from (x_1, y_1) until we reach $(x_0, y_0) = \operatorname*{argmin}_{x,y} S(x, y)$

✍ **11.** Show that if C is a point on the shortest path connecting A and B, then the path segment $[A, C]$ must be the shortest path connecting A and C (and likewise for the path segment $[C, B]$)

Download the following files from the website, and use them according to the provided instructions:

- `FSM.m` – A numerical solver for the eikonal equation.
- `updateFSM.m` – This performs the discretization of the PDE in a correct way.
- `runFSM.m` – This is the file you need to run, which provides instructions.

12. Use the file `runFSM.m` to compute $S[x,y]$, which is an approximate solution on a 2D regular grid of:

$$\begin{aligned} |\nabla S(x,y)| &= n(x,y) \\ S(x_0,y_0) &= 0 \end{aligned} \tag{17}$$

using the corresponding $n(x,y)$ for the Brachistochrone problem, and the initial point

$$(x_0,y_0) = (1,1)$$

((17) is usually referred to in the literature as the eikonal equation).

13. Find the shortest path connecting the point (x_0,y_0) and a point of your choice (x_1,y_1). To do that, perform gradient descent iterations on $S[x,y]$, starting from (x_1,y_1), until you reach (x_0,y_0), and trace the curve generated by this minimization procedure. To get a finer approximation of the gradient (rather than the coarse approximation by finite differences such as $\dfrac{\partial S}{\partial x} \approx S[x+1,y] - S[x,y]$ etc.) use some of Matlab's built in interpolation functions (see the **Tips** section).

14. Compare this curve to the analytic solution of this problem given in (13) by drawing them in different colors on top of $S[x,y]$. Use red for the numerically computed curve and green for the analytically computed curve. You can find k in (13) from the boundary conditions using Matlab's `solve` command. Use a grid of size 300×300 at least.

5 Shape From Shading

The problem of reconstructing a surface from a single image is known as **shape from shading**. The term *shading* refers to changes in image intensity. This is one of the most fundamental tasks in computer vision. For simplicity, consider a smooth three-dimensional object defined by $(x,y,z(x,y))$, and a single parallel light source at infinity defined by its direction $\mathbf{l} = [l_1,\ l_2,\ l_3]^T$ (since the light is far away, it can be considered to be in the same direction relative to all points on the surface). The goal is to extract the surface normals $\mathbf{N}(x,y)$ (i.e., equation (6) normalized) from the shading image, and then integrate them to get the surface. The gray-level shading image of a *Lambertian* object with unit albedo for an observer direction $\mathbf{v} = [0,\ 0,\ 1]^T$ ("parallel projection") is given by

$$I(x,y) = R(p,q) = \lambda \langle \mathbf{l}, \mathbf{N} \rangle \tag{18}$$

where λ is a proportionality factor that can be set to 1 by rescaling the image. This is a single equation (per pixel) in two variables (p,q), and therefore ill-posed. We have already encountered the notion of regularization, that helps us in obtaining solutions to ill-posed problems, in the first project. We will briefly revisit it here. For example, (9) can be used as a regularizer to produce a minimal surface out of the shading image. However, since (9) is highly non-linear, a simpler version of it is often used:

✍ **15.** Show that the **Dirichlet energy** functional

$$\int_\Omega \left(z_x^2 + z_y^2\right) dxdy \tag{19}$$

Can be obtained from (9) by linearization, and find the E-L equations.

However, we will explore here another framework based on the ideas developed in the previous section that does not require adding an *explicit* regularization term for the solution of the shape from shading problem. Instead, the regularization is *implicit* in the scheme.

✍ **16.** For a light source direction which coincides with the direction of the observer $\mathbf{l} = \mathbf{v}$, show that from (18) you get an **eikonal equation**:

$$|\nabla z| = F(x, y) \tag{20}$$

What is $F(x,y)$ (in terms of the given image $I(x,y)$)? Note that $F(x,y)$ is not allowed to be 0 (why?). This should be taken care of in the code (later on) by replacing 0 with $\epsilon > 0$.

To test these ideas, you take out of the boydem a template of Mozart's face that you have been keeping exactly for this moment.

Download the following files from the website:

- `mozart.mat` – A template of Mozart's face represented as a surface $z(x,y)$.
- `I.mat` – A shading image of Mozart's face.

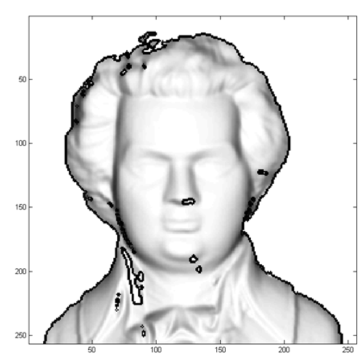


Figure 3: A shading (gray-level) image of Mozart's face using a light source direction of $[0,\ 0,\ 1]^T$ and a *Lambertian* reflectance model.

17. Can you reconstruct Mozart's face? Use `runFSM.m` to solve (20). As boundary conditions, set $z(x, y) = 0$ at the tip of Mozart's nose (adjust later to the correct value by inverting and translating the reconstructed surface). Compare with the ground truth surface, and use the following options for display:

```
colormap gray;
shading interp;
axis('tight');
view(110,45);
axis('off');
camlight;
```

6 Photometric Stereo

Reconstruction from a single image is a severely ill-posed task. A better reconstruction can be achieved if we have multiple shading images at our disposal acquired in different lighting conditions. This also leads to a simpler linear PDE (Poisson's) rather than the nonlinear eikonal equation. Each image introduces a linear equation relating the surface normals and the light source direction according to (18). These relations can be stacked into a system of linear equations **per pixel**:

$$\begin{bmatrix} I_1 \\ I_2 \\ I_3 \\ \dots \\ I_n \end{bmatrix} = \begin{bmatrix} l_{1,1} & l_{1,2} & l_{1,3} \\ l_{2,1} & l_{2,2} & l_{2,3} \\ l_{3,1} & l_{2,3} & l_{3,3} \\ & \dots & \\ l_{n,1} & l_{n,2} & l_{n,3} \end{bmatrix} \cdot \mathbf{n} \equiv \mathbf{L} \cdot \mathbf{n} \tag{21}$$

Download the following files from the website:

- `Images.mat` – A $[256 \times 256 \times 15]$ array of 15 images of Mozart taken with different light source directions assuming the Lambertian model.
- `LightSources.mat` – A $[15 \times 3]$ matrix of 15 light source directions (the matrix $\mathbf{L}$ in (21)).

18. Solve the system of equations (21) to get the surface normals. The system is overdetermined, so solve it in the least squares sense. Notice that the matrix $\mathbf{L}$ does not depend on the position (x, y) so it can be inverted once for the entire image. This formulation also allows us to recover the *albedo* of the surface, which is the magnitude of the recovered normal. However, we shall assume the albedo is constant for the entire surface, so the magnitude should be ignored.

19. From (6) you have the connection between the surface normal and the surface gradient. Apply the method developed in class to reconstruct a surface from gradients with the **natural boundary conditions** (either pad the surface such that $[p,q]^T = [0,0]^T$ along the padded boundary or use the value of $[p,q]^T$ at the original boundary to satisfy the natural boundary conditions). Use the Jacobi method (see below) to solve the sparse system of linear equations arising from Poisson's equation. Show that the Laplacian can be discretized by

$$\nabla^2 z(x,y) \approx -4z[x,y] + z[x+1,y] + z[x-1,y] + z[x,y+1] + z[x,y-1] \tag{22}$$

In order to apply Jacobi method you need to construct a matrix A such that when multiplied by the column stack representation of z, it will produce the column stack representation of (the discrete version of) $\nabla^2 z$. This is done in the same fashion you constructed derivative matrices in the first project. Along the boundary of the domain, the matrix multiplication should produce the boundary conditions you used. Notice that the resulting matrix A is extremely sparse, with only up to 5 elements per row. For the derivatives p_x, q_y use the code for central differences you developed in the first project.

Remark : For Poisson's equation, Jacobi method reduces to iteratively applying

$$z_{k+1}[x,y] = \frac{1}{4}\left(z_k[x-1,y] + z_k[x+1,y] + z_k[x,y-1] + z_k[x,y+1] - b[x,y]\right) \tag{23}$$

for each pixel (with the appropriate boundary conditions). However, in Matlab it is much more efficient to work with matrix operations than with loops.

input : A, $\mathbf{x}_0$,$\mathbf{b}$, k_{max},ϵ
output: (approximate) solution of $A\mathbf{x} = \mathbf{b}$
Decompose A into $A = D + R$ where D is a diagonal matrix and R is a matrix containing only off-diagonal elements.
while ($\|Ax - b\| > \epsilon$ & $k < k_{max}$) **do**
  $\mathbf{x}_{k+1} = D^{-1}(\mathbf{b} - R\mathbf{x}_k)$
  $k = k + 1$
end
Return $\mathbf{x}^*$

Algorithm 1: Jacobi Method

20. Compare qualitatively the reconstruction results from a single image and from multiple images.

7 Tips

- You may find the following Matlab commands useful :
`interp1, interp2, griddedinterpolant, surf, surface, sparse, spdiags, del2, gradient, solve, syms`

8 Acknowledgements

- The Mozart image is taken from [1].
- Matlab's photmetric stereo demo by Artur Bernat.

References

[1] Ruo Zhang, Ping-Sing Tsai, James Edwin Cryer, and Mubarak Shah, *Shape from Shading: A Survey* IEEE Transactions On Pattern Analysis And Machine Intelligence, Vol. 21, No. 8, August 1999

[2] Jain, Ramesh; Kasturi, Rangachar; Schunck, Brian G. *Machine Vision* (1995)