Practice Final

CMSC 426

The final will be cumulative for the whole semester. You should consider the 1st and 2nd Practice midterms as providing notes and practice for the final. In addition, the final will cover the following material, which we have discussed since the second midterm.

- The Essential Matrix, and epipolar geometry.
  - Understand what the Essential matrix means. How can it be used to determine whether two image points come from the same scene point? What is its relationship to the epipolar constraint?
  - Understand how to build the Essential matrix when the camera translates but does not rotate.
  - Understand how to build an Essential matrix when there is also rotation.
  - Understand how the 8 point algorithm can be used to construct the Essential matrix.
- Image coordinates and world coordinates. You should understand what we mean when we represent an image point with world coordinates, and when we represent an image with image coordinates, relative to the camera.
  - Given a description of a camera's focal point and local coordinate system, you should be able to translate between camera coordinates and world coordinates.
- 3D Motion Matrices
  - Know that a 3D rotation matrix has rows (and columns) that are orthonormal.
  - Be able to construct a 3D rotation matrix for simple rotations.
  - Understand that the rows of a rotation matrix represent the x, y and z directions in a new coordinate system.
  - Understand how to write translation in matrix form, and combine rotation and translation in a single matrix.
- Deep Learning
  - Understand how gradient descent works. Given a function, you should be able to compute its gradient and find the result of a gradient descent step.
  - Understand what a perceptron is and how the perceptron algorithm works. Know that the perceptron is a linear separator.
  - Understand how a feedforward neural network works. Given weights bias and inputs, for example, you should be able to figure out what it computes.
  - Understand what is particular about Convolutional Neural Networks compared to a general feedforward neural network.
  - Understand the basic idea of backpropagation and how to use the chain rule to compute the gradients in a neural network.

This is also a good time to take stock of some themes that run throughout the course. There are a few main concepts, mostly mathematical, that underlie much of the material we have discussed:

- **Correlation and Convolution**
  First, we discussed how to use convolution to smooth an image, or to find image derivatives. Convolutions show up later in convolutional neural networks.
- **Gradients**
  Image gradients are a fundamental way in which we measure how an image is changing. Understanding how to compute an image gradient, and getting an intuition for its properties is very important. This includes, for example, understanding that the direction of the image gradient encodes the direction in which the image is changing most rapidly, while the magnitude of the gradient tells us how rapidly the image is changing. Image gradients are basic to edge detection, but they also show up in optical flow, and again in gradient descent.
- **Representing Motion with Matrices**
  We have talked about how to represent rotation and translation in a matrix. This includes understanding what a rotation matrix is and how to build one. We've also talked about affine transformations, similarity transformations, and scaled orthographic projection with matrices. 3D matrices are needed for 3D operations, including the Essential matrix.
- **3D Geometry: Perspective projection and epipolar geometry**
  The core of this is understanding the pinhole camera model, and how it can be used to determine the relationship between a camera, a 3D point, and its 2D image. This is fundamental in any vision task that attempts to recover the 3D structure of the world. Next, we have talked about the relationship between a 3D scene and two cameras. This gives rise to epipolar geometry, which we have used to constrain stereo matching, understand flow fields and the focus of expansion, and build the Essential matrix.

The following equations are important. You should understand these equations and be able to use them.

1D and 2D correlation

$$F \circ I(x) = \sum_{i=-N}^{N} F(i) I(x+i) \qquad F \circ I(x,y) = \sum_{j=-N}^{N} \sum_{i=-N}^{N} F(i,j) I(x+i, y+j)$$

1D and 2D Convolution

$$F * I(x) = \sum_{i=-N}^{N} F(i) I(x-i) \qquad F * I(x,y) = \sum_{j=-N}^{N} \sum_{i=-N}^{N} F(i,j) I(x-i, y-j)$$

Definition of a gradient: $\nabla I = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \right)$

Formula for using the gradient to determine how the image changes as you move in a particular direction

$$I(x + \Delta \cos\theta, y + \Delta \sin\theta) - I(x,y) \approx \langle v, \nabla I \rangle \text{ where } v \equiv (\Delta \cos\theta, \Delta \sin\theta)$$

Definition of the partial derivative.

$$\frac{\partial I(x,y)}{\partial x} = \lim_{\Delta x \to 0} \frac{I(x+\Delta x, y) - I(x,y)}{\Delta x}$$

Perspective projection with focal point at the origin and camera facing in the *z* direction. (x,y) = f(X/Z, Y/Z)

Equation relating disparity to depth in stereo $\boxed{Z = f\frac{T}{d}}$

Equations for the Essential matrix. E is the Essential matrix, T is a vector representing the translation between the two focal points, R is a matrix representing the rotation between the two cameras.

$$\boxed{p'^T E p = 0} \qquad s = \begin{pmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{pmatrix} \qquad E=SR$$

Gradient Descent. This update is the main point: $\mathbf{x_n} \leftarrow \mathbf{x_c} - \eta \nabla f_{\mathbf{x}}(\mathbf{x_c})$

The chain rule. This is used to compute the gradient: $\frac{\partial C}{\partial w_i} = \frac{\partial C}{\partial a}\frac{\partial a}{\partial z}\frac{\partial z}{\partial w_i}$

What a layer of a neural network computes:

$$a_j^l = \sigma\left(\sum_k w_{jk}^l a_k^{l-1} + b_k^l\right)$$

The quadratic loss function:

$$C_{quadratic} = \frac{1}{2n}\sum_x \|y(x) - a^L(x)\|^2$$

An example of a nonlinear function

$$RELU(z) = \max(0, z)$$

Practice Problems

1. 3D rotations
   a. Which of the following matrices represent 3D rotations? Explain your answer for each.

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

This is a rotation, just the identity transformation, meaning a rotation of 0 degrees.

$$\begin{pmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

Also a rotation. Note that the rows are orthonormal.

$$\begin{pmatrix} 1/\sqrt{3} & 1/\sqrt{3} & 1/\sqrt{3} \\ 0 & 1/\sqrt{2} & -1/\sqrt{2} \\ -1/\sqrt{2} & 1/\sqrt{2} & 0 \end{pmatrix}$$

No. The second and third rows are not orthogonal.

$$\begin{pmatrix} 1/\sqrt{3} & 1/\sqrt{3} & 1/\sqrt{3} \\ 0 & 1/\sqrt{2} & -1/\sqrt{2} \\ -\sqrt{2}/\sqrt{3} & 1/\sqrt{6} & 1/\sqrt{6} \end{pmatrix}$$

Yes, the rows are orthonormal. (By the way, the determinant of this matrix is 1. If the rows are orthonormal, the determinant could be -1, which would mean that there is a rotation and a reflection.)

2. Consider a neural network that has no nonlinearity. It has three input units. These are connected to an output unit, with weights 1, 2 and 3. There is no bias term. The loss function is the L2 norm between the output and label. We are given an input of (3,2,3) and a label of 20. With this input, what will the network output?

*The output is just the product of each input by the corresponding weight, added together. 3*1 + 2*2 + 3*3 = 16. There is no nonlinearity here.*

3. What is the gradient of the loss with respect to the weights? If a gradient descent step consists of changing the weights based on the gradient times .01 (the step size) what will the new weights be? What is the loss with the new weights (I suggest you use Matlab for this)?

We can write:
C = (y-a)^2
*a = w1*x1 + w2x2 + w3x3.*

dC/da = -2(y-a).
da/dwi = xi.

(I'm being a little lazy here and using d to indicate the partial derivative, instead of inserting the partial symbol. So dC/da is the partial of C with respect to a).

So dC/dwi is -2(y-a)xi.

So the gradient is:
-2(y-a)(x1,x2,x3).

With the given input we have a = 16 (see last problem). This gives us a gradient of: -2(20-16)(3,2,3) = -8(3,2,3) = (-24,-16,-24). Taking a step of .01 times the negative gradient we have the new weights as:
(1,2,3) + (.24, .16, .24) = (1.24, 2.16, 3.24).

4. Using Matlab, repeat this process until it converges. What weights do you get? What loss?

In Matlab:

```
>> x = [3;2;3];
>> w = [1;2;3];
>> y = 20;
>> for i = 1:100 a = x'*w; g = -2*(y-a)*x; w = w - .01*g; end

w =

   1.5455
   2.3636
   3.5455

>> a

a =

   20
```

So the loss is 0.

5. Suppose we add Relu as a nonlinearity. So the output of the network is the max of 0 and the output of the previous network. How would this change the backpropagation step and the result?

We could write this as:

*z = w1*x1 + w2x2 + w3x3.*

and

a = max(z,0)

So dC/dwi is now (dC/da)(da/dz)(dz/dwi). We have:

dC/da = -2(y-a).
dz/dwi = xi.

We just need to compute da/dz. That is, if we change z a little bit, how does this change a? The answer is that if z is positive, the derivative equals 1, and if z is negative the derivative is 0. If z is exactly 0, things seem a little tricky, since the derivative isn't *really defined at this point. But it turns out that since this is just a single value, we can get away with setting the derivative to 0 at this point. So if we define h to be a function so that h(z) = 1 when z > 0, and h(z) = 0 otherwise, we can write*

da/dz = h(z). And we get the gradient of C with respect to the weights as:

-2(y-a)h(z)(x1,x2,x3).

In the example we had above, z was always greater than 0, so nothing would have been any different.

6. Suppose we have z(x,y) = (x+y)sin(x+y). Using the chain rule, find the gradient of z with respect to (x,y) at the point x = 0, y = 0.

*We can set w = x+y. Then we have z = w*sin(w). dz/dw = sin(w) + wcos(w). We have dw/dx = 1 and dw/dy = 1. So dz/dx = dz/dy = sin(x+y) + (x+y)cos(x+y). So the gradient at x = 0, y=0 is zero.*

7. Essential Matrix
    a. Suppose we have a camera that is facing in the direction zvec = (1/sqrt(2), 0 1/sqrt(2)), that is, that is the z direction relative to the camera. The y direction for the camera is yvec = (0,1,0). What is the x direction for the camera?

We know that the x, y and z directions have to be orthonormal. So the inner product of xvec and yvec has to be 0. Since this inner product is xvec(2), this means that the y component of xvec has to be 0. Since xvec.zvec = 0, we must have (xvec(1) + xvec(3))/sqrt(2) = 0, so xvec(1) = -xvec(3). Since xvec must be a unit vector, we have xvec(1)^2 + xvec(3)^2 = 2xvec(1)^2 = 1. So xvec(1) = +- 1/sqrt(2). This means either xvec = (1/sqrt(2), 0, -1/(sqrt(2)) or (-1/sqrt(2), 0, 1/sqrt(2)). One of these choices gives us a left-handed coordinate system. One way to tell if the coordinate system is correct is to check the determinant of [xvec; yvec; zvec], which should be 1. Doing this we find that the x direction should be xvec = (1/sqrt(2), 0, -1/(sqrt(2)).

    b. Suppose the camera in (a) has a focal point at fp = (1,1,1). If there is a point in the world with coordinates p = (6,6,6), what will its coordinates be in the coordinate system defined by this camera?

To compute the x coordinate, we take (p-fp).xvec. That is, ((6,6,6) – (1,1,1)).(1/sqrt(2), 0, -1/(sqrt(2)). This works out to be 0. We compute the y and z coordinates in the same way, and get the point (0, 5, 10/sqrt(2)).

    c. What will be the coordinates of the point p in the image taken by this camera (using the point and camera from (a) and (b)? Assume the camera has a focal length of 1.

We can just use the formula (x,y) = f(X/Z, Y/Z). Since we've written the point's coordinates relative to the camera, it is as if we have a camera with a focal point at the origin and x, y and z directions in the normal directions. So we get the image point (0, 1/2sqrt(2)).

d. Suppose the Essential matrix looks like this:

$$\begin{pmatrix} 1 & 0 & 2 \\ 2 & 1 & 0 \\ 3 & 1 & a \end{pmatrix}$$

Suppose further that we know that a point with coordinates (1,1,1) in the first camera's coordinates matches a point with coordinates (0, 2, 1) in the second camera. What would you expect the value of a to be?

Multiplying $(1,1,1)E(0,2,1)^T$ *we get* $(6,2,a)(0,2,1)^T = 4 + a = 0$. *So* $a = -4$.

e. Actually, the resulting Essential matrix you got in problem (d) is not really a valid Essential matrix. Explain why this is the case.

*The Essential matrix must have rank 2. This can be seen by the fact that the E=S*R and*
$S = \begin{pmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{pmatrix}$
has rank 2. The solution given in (d) does not have rank 2, since its determinant is not 0.

f. Suppose we have two cameras. The first has a focal point at (0,0,0) and an image plane of z = 1. The second has a focal point at (10,0,0) and an image plane of z = 1. However, the second camera has been rotated so that its x direction is (0,1,0) and its y direction is (-1,0,0). What is the essential matrix that relates these cameras?

*E=S*R and* $S = \begin{pmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -10 \\ 0 & 10 & 0 \end{pmatrix}$. *In this case, we also have*
$R = \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ *so we get E=* $\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -10 \\ -10 & 0 & 0 \end{pmatrix}$. *We can verify this by noting that if a point in the first camera has coordinates (a,b,1) its x coordinate in the second image should be -b. Multiplying (a,b,1) by E we get (-10,0,-10b). Multiplying this by (c,d,1) and setting the result to 0 we get -10c-10b=0 so c = -b, as it should be.*

g. Suppose we have a camera with a focal point at (0,0,0), pointing in the z direction, with a focal length of 1. We have another camera with a focal point at (10,8,2). Where is the epipole in the first image?

The epipole is just the location in the image where the second focal point would appear, if it could be seen. This is just (5,4,1).