UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

CS440/ECE448 Artificial Intelligence

Exam 1

Spring 2022

Exam 1 will be February 21, 2022

Your Name: _____

Your NetID: _____

Your Section: _____

Instructions

- Please write your name and NetID on the top of every page.
- This will be a CLOSED BOOK exam. No additional written materials (books, cheat-sheets, etc.) or electronic devices (phones, tablets, calculators, computers etc.) are allowed.
- No calculators are permitted. You need not simplify explicit numerical expressions.

Question 1 ***(0 points)***

Let A and B be independent binary random variables with $P(A=1)=0.1$, $P(B=1)=0.4$. Let $C = A \vee B$, and let $D = A \oplus B$ (A XOR B).

(a) What is $P(C=1)$?

Solution:

$$\begin{aligned} P(C=1) &= P(A=1,B=1)+P(A=1,B=0)+P(A=0,B=1) \\ &= (0.1)(0.4)+(0.1)(0.6)+(0.9)(0.4) = 0.46 \end{aligned}$$

where the last line follows from the independence of A and B.

(b) What is $P(D=1)$?

Solution:

$$\begin{aligned} P(D=1) &= P(A=1,B=0)+P(A=0,B=1) \\ &= (0.1)(0.6)+(0.9)(0.4) = 0.42 \end{aligned}$$

(c) What is $P(D|A=1)$?

Solution:

$$\begin{aligned} P(D=1|A=1) &= P(D=1,A=1)/P(A=1) \\ &= P(A=1,B=0)/P(A=1) \\ &= \frac{0.06}{0.1} = 0.6 \end{aligned}$$

$$P(D=0|A=1) = 1 - P(D=1|A=1) = 0.4$$

(d) Are A and D independent? Why?

Solution: No. $P(D=1|A=1) \neq P(D=1)$.

Question 2 ***(0 points)***

Use the axioms of probability to prove that, for a binary random variable A, $P(A=0) = 1–P(A=1)$.

Solution:

- From the third axiom, $P(A \vee \neg A) = P(A) + P(\neg A) - P(A \wedge \neg A)$.
- The event $(A \vee \neg A)$ is always true, so from the second axiom, $P(A \vee \neg A) = 1$. The event $(A \wedge \neg A)$ is always false, so from the second axiom, $P(A \wedge \neg A) = 0$.
- Combining the two statements above, $1 = P(A) + P(\neg A)$. Q.E.D.

Question 3 ***(0 points)***

20% of students at U of I live north of University Ave. Amongst these students, 10% study engineering. Furthermore, 15% of the entire student body studies engineering. Given that we know that a student studies engineering, what is the probability that the student does not live north of University Ave?

Solution: Define $G = 1$ if a student lives north of University Ave, $E = 1$ if a student studies engineering. We are given that $P(G=1) = 0.2$ and $P(E=1|G=1) = 0.1$, from which we may infer that $P(E=1, G=1) = 0.02$. We are also given that $P(E=1) = 0.15$, from which we may infer that

$$P(\neg G, E) = P(E) - P(E, G) = 0.13$$
$$P(\neg G|E) = \frac{P(\neg G, E)}{P(E)} = \frac{0.13}{0.15} = \frac{13}{15}$$

Question 4 ***(0 points)***

Consider the following joint probability distribution, for binary random variables A and B:

$$P(A=1, B=1) = 0.12$$
$$P(A=1, B=0) = 0.18$$
$$P(A=0, B=1) = 0.28$$
$$P(A=0, B=0) = 0.42$$

What are the marginal distributions of A and B? Are A and B independent and why?

Solution: $P(A=1) = 0.3, P(A=0) = 0.7, P(B=1) = 0.4, P(B=0) = 0.6$. They are independent, because $P(A=1)P(B=1) = P(A=1, B=1) = 0.12, P(A=1)P(B=0) = P(A=1, B=0) = 0.18$, and so on.

Question 5 ***(0 points)***

Laplace invented "Laplace smoothing" in order to estimate the probability that the sun will rise tomorrow. Suppose he had historical records indicating that the sun had been observed to rise on 1,826,200 consecutive days (and the event "the sun did not rise today" has never been observed). What probability would Laplace smoothing estimate for the event "The sun wil rise tomorrow"?

Solution:

$$P(R=1) = \frac{1826200+1}{1826200+2}$$

Question 6 ***(0 points)***

Y is a random variable denoting the class of a newspaper title: $Y=0$ means the article is about sports, $Y=1$ means the article is about science. The title is only three words long; its three words are the random variables W_1, W_2, and W_3. Depending on whether the article is about sports or science, the title may contain any word from the following vocabulary: $\{\text{Illini}, \text{win}, \text{discover}, \text{everything}\}$. The prior probability of an article about science is $P(Y=1)=0.4$. Assume a naïve Bayes model, with word likelihoods of

$$\begin{aligned}
P(W_i = \text{Illini}|Y=0) &= 0.3 \\
P(W_i = \text{Illini}|Y=1) &= 0.3 \\
P(W_i = \text{win}|Y=0) &= 0.3 \\
P(W_i = \text{win}|Y=1) &= 0.1 \\
P(W_i = \text{discover}|Y=0) &= 0.1 \\
P(W_i = \text{discover}|Y=1) &= 0.4
\end{aligned}$$

Now you download the article, and discover that its title is $X = \text{Illini discover everything}$. What is $P(Y=1|W_1=\text{Illini}, W_2=\text{discover}, W_3=\text{everything})$? Leave your answer in the form of an expression composed of numbers; do not simplify.

Solution:

$$P(Y=1|X) = \frac{P(Y=1,X)}{P(Y=0,X)+P(Y=1,X)}$$

$$P(Y=1,X) = P(Y=1)P(W_1|Y=1)P(W_2|Y=1)P(W_3|Y=1) = (0.4)(0.3)(0.4)(0.2)$$

$$P(Y=0,X) = P(Y=0)P(W_1|Y=0)P(W_2|Y=0)P(W_3|Y=0) = (0.6)(0.3)(0.1)(0.3)$$

$$P(Y=1|X) = \frac{(0.4)(0.3)(0.4)(0.2)}{(0.6)(0.3)(0.1)(0.3)+(0.4)(0.3)(0.4)(0.2)}$$

Question 7 ***(0 points)***

You're on a phone call with your friend, trying to help figure out why their computer won't start. There are only two possibilities, $Y = \text{CPU}$, or $Y = \text{PowerSupply}$, with prior probability $P(Y = \text{CPU}) = 0.3$.

You ask your friend whether the computer makes noise when they try to turn it on. There are two possibilities, $X = \text{quiet}$, and $X = \text{loud}$. You know that a power supply problem often leaves a quiet computer, but that the relationship is stochastic, as shown:

$$P(X = \text{noise}|Y = \text{CPU}) = 0.8, \quad P(X = \text{noise}|Y = \text{PowerSupply}) = 0.4$$

(a) What is the MAP classifier function $f(X)$, as a function of X?

Solution: The joint probabilities of evidence and label are:

$$P(\text{noise}, \text{CPU}) = 0.24, \quad P(\text{noise}, \text{PowerSupply}) = 0.28$$
$$P(\text{quiet}, \text{CPU}) = 0.06, \quad P(\text{quiet}, \text{PowerSupply}) = 0.42$$

Choosing the maximum *a posteriori* label given each observation gives

$$f(\text{noise}) = \text{PowerSupply}, \quad f(\text{quiet}) = \text{PowerSupply}$$

In other words, regardless of whether the computer is noisy or quiet, the power supply is always the most probable source of the problem.

(b) What is the Bayes error rate?

Solution: The Bayes error rate is the probability of error of the optimal classifier, which is $P(\text{noise}, \text{CPU}) + P(\text{quiet}, \text{CPU}) = 0.3$.

(c) CPU damage is more expensive than power supply damage, so let's define a false alarm to be the case where your classifier says $f(X) = \text{CPU}$, but the actual problem is $Y = \text{PowerSupply}$. Under this definition, what are the false-alarm rate and missed-detection rate of the MAP classifier?

Solution: The MAP classifier always guesses "Power Supply," so the false alarm and missed detection rates are

$$P(f(X) = \text{CPU}|Y = \text{PowerSupply}) = 0.0$$
$$P(f(X) = \text{PowerSupply}|Y = \text{CPU}) = 1.0$$

Question 8 ***(0 points)***

Consider the following binary logic function:

$$y = \neg\left((x_1 \wedge \neg x_2 \wedge \neg x_3) \vee (x_1 \wedge x_2 \wedge \neg x_3)\right)$$

Convert truth values to numbers in the obvious way: let $x_i = 1$ be a synonoym for $x_i =$ **True**, and let $x_i = 0$ by a synonym for $x_i =$ **False**. Let $\vec{x} = [x_1, x_2, x_3]^T$ and $\vec{w} = [w_1, w_2, w_3]^T$, let $\vec{x}^T\vec{w}$ denote the dot product of vectors $\vec{x}$ and $\vec{w}$, and let $u(\cdot)$ denote the unit step function. Find a set of parameters w_1, w_2, w_3 and b such that the logic function shown above can be computed as $y = u(w^T x + b)$.

Solution: Drawing up a truth table, we get

x_1	x_2	x_3	y
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

If we plot these eight points in 3D space, we find that the dots for which $Y = 0$ and the dots for which $Y = 1$ are separated, for example, by the plane $-x_1 + x_3 + 0.5 = 0$, with the $Y = 1$ dots on the side of this plane where $-x_1 + x_3 + 0.5 > 0$. The linear classifier can therefore use $\vec{w} = [-1, 0, 1]^T$, $b = 0.5$.

Question 9 ***(0 points)***

We want to implement a classifier that takes two input values, where each value is either 0, 1 or 2, and outputs a 1 if at least one of the two inputs has value 2; otherwise it outputs a 0. Can this function be implemented by a linear classifier? If so, construct a linear classifier that does it; if not, say why not.

Solution: In this case the input space of all possible examples with their target outputs is:

	0	1	2
2	1	1	1
1	0	0	1
0	0	0	1

Since there is clearly no line that can separate the two classes, this function is not linearly separable and so it cannot be learned by a Perceptron.

Question 10 ***(0 points)***

Consider a problem with a binary label variable, Y, whose prior is $P(Y=1)=0.4$. Suppose that there are 100 binary evidence variables, $X=[X_1,\ldots,X_{100}]$, each with likelihoods given by $P(X_i=1|Y=0)=0.3$ and $P(X_i=1|Y=1)=0.8$ for $1\leq i\leq 100$.

(a) Specify the classifier function, $f(\vec{x})$, for a naive Bayes classifier, where $\vec{x}=[x_1,\ldots,x_{100}]^T$ is the set of observed values of the evidence variables. You might find it useful to define $N(\vec{x})=$ the number of nonzero elements of the binary vector $\vec{x}$; note that $0\leq N(\vec{x})\leq 100$.

Solution:

$$f(\vec{x})=\begin{cases} 1 & P(Y=1)\prod_{i:x_i=1}P(X_i=1|Y=1)\prod_{i:x_i=0}P(X_i=0|Y=1)> \\ & (1-P(Y=1))\prod_{i:x_i=1}P(X_i=1|Y=0)\prod_{i:x_i=0}P(X_i=0|Y=0) \\ 0 & \text{otherwise}\end{cases}$$

Plugging in the parameter values, we get:

$$f(\vec{x})=\begin{cases} 1 & 0.4(0.8)^{N(\vec{x})}(0.2)^{100-N(\vec{x})}>0.6(0.3)^{N(\vec{x})}(0.7)^{100-N(\vec{x})} \\ 0 & \text{otherwise}\end{cases}$$

Another way to write this would be:

$$f(\vec{x})=\begin{cases} 1 & 0.4\prod_{i=1}^{100}(0.8)^{x_i}(0.2)^{1-x_i}>0.6\prod_{i=1}^{100}(0.3)^{x_i}(0.7)^{1-x_i} \\ 0 & \text{otherwise}\end{cases}$$

(b) The naive Bayes classifier can be written as

$$f(\vec{x})=\begin{cases} 1 & \vec{w}^T\vec{x}+b>0 \\ 0 & \text{otherwise}\end{cases},$$

where $\vec{w}^T\vec{x}$ is the dot product between the vectors $\vec{w}$ and $\vec{x}$. Find $\vec{w}$ and b (write them as expressions in terms of constants; don't simplify).

Solution:

$$f(\vec{x})=\begin{cases} 1 & \ln(0.4)+\sum_{i=1}^{100}x_i\ln(0.8)+(1-x_i)\ln(0.2)>\ln(0.6)+\sum_{i=1}^{100}x_i\ln(0.3)+(1-x_i)\ln(0.7) \\ 0 & \text{otherwise}\end{cases}$$

so the parameters are

$$b=\ln(0.4)-\ln(0.6)+100\ln(0.2)-100\ln(0.7)$$
$$w_i=\ln(0.8)-\ln(0.2)-\ln(0.3)+\ln(0.7),\quad 1\leq i\leq 100$$

Question 11 ***(10 points)***

You are a Hollywood producer. You have a script in your hand and you want to make a movie. Before starting, however, you want to predict if the film you want to make will rake in huge profits, or utterly fail at the box office. You hire two critics A and B to read the script and rate it on a scale of 1 to 5 (assume only integer scores). Each critic reads it independently and announces their verdict. Of course, the critics might be biased and/or not perfect, therefore you may not be able to simply average their scores. Instead, you decide to use a perceptron to classify your data. There are two features: $x_1 =$ score given by reviewer A, and $x_2 =$ score given by reviewer B.

Movie Name	A	B	Profit
Pellet Power	1	1	No
Ghosts!	3	2	Yes
Pac is bac	4	5	No
Not a Pizza	3	4	Yes
Endless Maze	2	3	Yes

(a) (5 points) Train the perceptron to generate $f(\vec{x}) = 1$ if the movie returns a profit, $f(\vec{x}) = -1$ otherwise. The initial weights are $b = -1$,$w_1 = 0$,$w_2 = 0$. Present each row of the table as a training token, and update the perceptron weights before moving on to the next row. Use a learning rate of $\alpha = 1$. After each of the training examples has been presented once (one epoch), what are the weights?

Solution: The first row of the table is correctly classified, therefore the weights are not changed. The second row is incorrectly classified, therefore the weights are updated as $\vec{w} = \vec{w} + y\vec{x} = [0,3,2]$. Using these weights results in misclassification of the third row, therefore the weights are updated again to $[-1,-1,-3]$. Using these weights results in misclassification of the fourth row, therefore the weights are updated again to $[0,2,1]$. These weights correctly classify the fifth row.

(b) (3 points) Suppose that, instead of learning whether or not the movie is profitable, you want to learn a perceptron that will always output $f(\vec{x}) = +1$ when the total of the two reviewer scores is more than 8, and $f(\vec{x}) = -1$ otherwise. Is this possible? If so, what are the weights b, w_1, and w_2 that will make this possible?

Solution: Yes, a perceptron can learn this function. Any weights such that $w_1 = w_2$ and $b < -8w_1$ are correct; for example, the weights $[-8.1,1,1]$.

(c) (2 points) Instead of either part (a) or part (b), suppose you want to learn a perceptron that will always output $f(\vec{x}) = +1$ when the two reviewers agree (when their scores are exactly the same), and will output $f(\vec{x}) = -1$ otherwise. Is this possible? If so, what are the weights b, w_1 and w_2 that will make this possible?

Solution: This problem is the logical complement of the XOR problem, therefore it is not linearly separable, and cannot be learned by a perceptron.

Question 12 ***(0 points)***

An image classification algorithm is being trained using the multiclass perceptron learning rule. There are 10 classes, each parameterized by a weight vector $\vec{w}_k$, for $0 \leq k \leq 9$. During the last round of training, all of the training tokens were correctly classified. Which of the weight vectors were updated, and why?

Solution: None. The perceptron learning rule updates the weight vectors only if the classifier makes a mistake.

Question 13 ***(0 points)***

Logistic regression is trained using gradient descent, with the goal of achieving the Bayes error rate (the lowest possible error rate) on testing data. There are many reasons why gradient descent might not successfully minimize the number of test-corpus errors. List at least three.

Solution: Here are a few:

1. **Wrong criterion:** The number of errors is not a differentiable criterion, so gradient descent has to minimize a differentiable approximation. Minimizing the differentiable approximation might not actually minimize the number of errors.
2. **Generalization error:** Minimizing error on the training corpus might not minimize error on the test corpus.
3. **Approximation error:** The Bayes error rate might not be achievable by a linear classifier. Since logistic regression learns a linear classifier, it might not be possible to achieve the Bayes error rate.
4. **Local optimum:** Gradient descent converges to a local minimum of the training criterion, not a global minimum.
5. **Computational limitations:** The amount of computation available for training might not be enough for gradient descent to fully converge.

Question 14 ***(0 points)***

The softmax function is defined as

$$f_k = \frac{\exp(\xi_k)}{\sum_j \exp(\xi_j)}$$

Find $df_5/d\xi_3$ in terms of f_3, f_5, ξ_3 and/or ξ_5.

Solution:

$$\frac{df_5}{d\xi_3} = -\frac{\exp(\xi_5)}{(\sum_j \exp(\xi_j))^2}\frac{d\sum_j \exp(\xi_j)}{d\xi_3} = -\frac{\exp(\xi_5)\exp(\xi_3)}{(\sum_j \exp(\xi_j))^2} = -f_5 f_3$$

Question 15 ***(0 points)***

A particular two-layer neural net has input vector $\vec{x} = [x_1, x_2]^T$, hidden layer activations $\vec{h}^{(1)} = [h_1^{(1)}, h_2^{(1)}]^T$, and a scalar output f. Its weights and biases are stored in the first-layer weight matrix $W^{(1)}$ and bias vector $\vec{b}^{(1)}$, and the second-layer bias vector $\vec{w}^{(2)}$ and bias $b^{(2)}$, respectively. The weights and biases are given to you; their values are also provided in Table 1. The hidden layer nonlineary is ReLU; the output nonlinearity is a logistic sigmoid.

Table 1: Variables used in Problem 15.

$$W^{(1)} = \begin{bmatrix} 3 & 4 \\ 0 & 9 \end{bmatrix}$$
$$\vec{b}^{(1)} = [-3, 3]^T$$
$$\vec{w}^{(2)} = [5, 4]^T$$
$$b^{(2)} = -7$$

(a) Suppose the input is $\vec{x} = [9, -6]^T$. What is $\vec{h}^{(1)}$? Write your answer as a vector of ReLUs of sums of products; do not simplify.

Solution:

$$\vec{h}^{(1)} = [\text{ReLU}((3)(9) + (4)(-6) + -3), \text{ReLU}((0)(9) + (9)(-6) + 3)]^T$$

(b) Suppose the hidden layer is $\vec{h}^{(1)} = [4, 5]^T$. What is f? Write your answer as a ratio of terms involving the exponential of a sum of products; do not simplify.

Solution:

$$f = 1/(1 + \exp(-(5)(4) - (4)(5) + 7))$$

Question 16 ***(0 points)***

You have a two-layer neural network trained as an animal classifier. The input feature vector is $\vec{x} = [x_1, x_2, x_3]^T$, where x_1, x_2, and x_3 are some features. There are two hidden nodes $\vec{h}^{(1)} = [h_1^{(1)}, h_2^{(1)}]^T$, and three output nodes, $\vec{f} = [f_1, f_2, f_3]^T$, corresponding to the three output classes f_1 =Pr(Y=dog|X=x), f_2 =Pr(Y=cat|X=x), and f_3 =Pr(Y=skunk|X=x). The hidden layer uses a sigmoid nonlinearity, the output layer uses a softmax. The weight matrices have elements $w_{j,k}^{(l)}$, and the biases are $b_j^{(l)}$.

(a) A Maltese puppy has the feature vector $\vec{x} = [2, 20, -1]^T$. Suppose all weights and biases are initialized to zero. What is $\vec{f}$?

Solution: If all weights and biases are zero, then the excitation of each hidden node is $0 \times 2 + 0 \times 20 + 0 \times (-1) + 0 \times 1 = 0$. The sigmoid of zero is $1/(1+\exp(0)) = 0.5$, but weights in the last layer are also all zero, so the excitations at the last layer are all zero. With a softmax nonlinearity, every output node is computing $\exp(0)/\sum_{i=1}^{3}\exp(0) = 1/3$. So

$$\vec{f} = \left[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right]^T$$

(b) Let $w_{i,j}^{(2)}$ be the weight connecting the i^{th} output node to the j^{th} hidden node. What is $df_2/dw_{2,1}^{(2)}$? Write your answer in terms of $h_i^{(2)}$, $w_{i,j}^{(2)}$, and/or the hidden node activations $h_j^{(1)}$, for any appropriate values of i and/or j.

Solution: Let's use the notation $\xi_i^{(2)}$ as the excitation of the i^{th} output node. That allows us to write the softmax as:

$$f_2 = \frac{\exp(\xi_2^{(2)})}{\sum_{j=1}^{3}\exp(\xi_j^{(2)})}, \quad \xi_j^{(2)} = b_j^{(2)} + \sum_i w_{ji} h_i^{(1)}$$

Then:

$$\begin{aligned}
\frac{df_2}{dw_{21}^{(2)}} &= \frac{1}{\sum_{i=1}^{3}\exp(\xi_i^{(2)})}\frac{d\exp(\xi_2^{(2)})}{dw_{21}^{(2)}} + \exp(\xi_2^{(2)})\frac{d(1/\sum_i \exp(\xi_i^{(2)})))}{dw_{21}^{(2)}} \\
&= \frac{1}{\sum_{i=1}^{3}\exp(\xi_i^{(2)})}\exp(\xi_2^{(2)})\frac{d\xi_2^{(2)}}{dw_{21}^{(2)}} + \exp(\xi_2^{(2)})\left(-\frac{1}{(\sum_{i=1}^{3}\exp(\xi_i^{(2)}))^2}\right)\frac{d(\sum\exp(\xi_i^{(2)})))}{dw_{21}^{(2)}} \\
&= \frac{\exp(\xi_2^{(2)})}{\sum_{i=1}^{3}\exp(\xi_i^{(2)})}h_1^{(1)} - \frac{\exp(\xi_2^{(2)})}{(\sum_{i=1}^{3}\exp(\xi_i^{(2)}))^2}\frac{d\exp(\xi_2^{(2)})}{dw_{21}^{(2)}} \\
&= \frac{\exp(\xi_2^{(2)})}{\sum_{i=1}^{3}\exp(\xi_i^{(2)})}h_1^{(1)} - \frac{\exp(\xi_2^{(2)})}{(\sum_{i=1}^{3}\exp(\xi_i^{(2)}))^2}\exp(\xi_2^{(2)})\frac{d\xi_2^{(2)}}{dw_{21}^{(2)}} \\
&= \frac{\exp(\xi_2^{(2)})}{\sum_{i=1}^{3}\exp(\xi_i^{(2)})}h_1^{(1)} - \frac{\exp(\xi_2^{(2)})^2}{(\sum_{i=1}^{3}\exp(\xi_i^{(2)}))^2}h_1^{(1)} \\
&= f_2(1-f_2)h_1^{(1)}
\end{aligned}$$

(c) Suppose that you are presented with an all-zero feature vector $\vec{x} = [0,0,0]^T$. Suppose that the first-layer weight matrix is also all zero, $w_{j,k}^{(1)} = 0$, but the bias is nonzero, specifically, it has the value $\vec{b}^{(1)} = [12,13]^T$. Suppose that, for this particular training token, $df_2/dh_1^{(1)} = 15$. What is $df_2/db_1^{(1)}$? Write your answer as a product of fractions involving exponentials of integers; there should be only constants in your answer, no variables, but you need not simplify.

Solution:

$$\begin{aligned}
\frac{df_2}{db_1^{(1)}} &= \frac{df_2}{dh_1^{(1)}} \frac{dh_1^{(1)}}{d\xi_1^{(1)}} \frac{d\xi_1^{(1)}}{db_1^{(1)}} \\
&= \frac{df_2}{dh_1^{(1)}} \sigma'\left(\xi_1^{(1)}\right) \\
&= \frac{df_2}{dh_1^{(1)}} \left(\frac{\exp(-\xi_1^{(1)})}{(1+\exp(-\xi_1^{(1)}))^2} \right) \\
&= 15 \left(\frac{\exp(-12)}{(1+\exp(-12))^2} \right)
\end{aligned}$$

Question 17 ***(0 points)***

In a pinhole camera, a light source at (x,y,z) is projected onto a pixel at $(x',y',-f)$ through a pinhole at $(0,0,0)$. Write $\sqrt{(x')^2+(y')^2}$ in terms of x, y, z, and f.

Solution: From the idea of similar triangles, we have

$$\frac{x'}{f} = -\frac{x}{z}, \quad \frac{y'}{f} = -\frac{y}{z}$$

from which we derive

$$\sqrt{(x')^2+(y')^2} = \frac{f}{z}\sqrt{x^2+y^2}$$

Question 18 ***(0 points)***

The real world contains two parallel infinite-length lines, whose equations, in terms of the coordinates (x,y,z), are parameterized as $ax+by+cz=d$ and $ax+by+cz=e$; in addition, both of these lines are on the ground plane, $y=g$, for some constants (a,b,c,d,e,g). Show that the images of these two lines, as imaged by a pinhole camera, converge to a vanishing point, and give the coordinates (x',y') of the vanishing point.

Solution: From the idea of similar triangles, we have

$$\frac{x'}{f} = -\frac{x}{z}, \quad \frac{y'}{f} = -\frac{y}{z}$$

From which we derive

$$x = \frac{-zx'}{f}, \quad y = \frac{-zy'}{f}$$

So the equations of the two lines are

$$-\frac{ax'}{f} - \frac{by'}{f} + c = \frac{d}{z}$$
$$-\frac{ax'}{f} - \frac{by'}{f} + c = \frac{e}{z}$$

As $z \to \infty$, the right-hand-sides of these two equations both go to zero, and the equations of both lines converge to

$$ax' + by' = cf$$

In addition, we have $y=g$, so $y' = -fg/z \to 0$, and therefore $x' = cf/a$. The coordinates are $(x',y') = (cf/a, 0)$.

Question 19 ***(0 points)***

Consider the convolution equation

$$Z(x',y') = \sum_m \sum_n h(m,n) Y(x'-m, y'-n)$$

Where $Y(x',y')$ is the original image, $Z(x',y')$ is the filtered image, and the filter $h(m,n)$ is given by

$$h(m,n) = \begin{cases} \frac{1}{21} & 1 \le m \le 3, \quad -3 \le n \le 3 \\ -\frac{1}{21} & -3 \le m \le -1, \quad -3 \le n \le 3 \end{cases}$$

Would this filter be more useful for smoothing, or for edge detection? Why?

Solution: The sum of $h(m,n)$, over all m and n, is 0. So if it is filtering a constant-color region, the output would always be zero, regardless of the input color. So it's not very useful for smoothing.

Any given pixel of $Z(x',y')$ is the difference between the pixels $Y(x',y')$ to its left, minus those to its right. Since it's computing a difference, it would be useful for edge detection.

Question 20 ***(0 points)***

Under what circumstances is a difference-of-Gaussians filter more useful for edge detection than a simple pixel difference?

Solution: A difference-of-Gaussians filter first smooths the input image (using a Gaussian smoother), then computes a pixel difference. The smoothing step can reduce random noise. Therefore, this procedure is more useful if the input image has some random noise in it.