CS 184/284A Spring 2024
Exam 2 (In Person)

April 22nd, 2024
Time Limit: 110 minutes

Name: _____

SID Number: _____

Room: _____

Instructions:

- Please wait to look inside the booklet until we officially start the exam.
- Please carefully read and sign the Honor Code described at the bottom of this page.
- Put your initials on the top of every page, in case the pages become separated during scanning.
- This exam is closed book, except for two 8.5×11 page of notes (double sided), printed or handwritten.
- This exam is 110 minutes long, and has a total of 100 points. There are 22 pages (including this cover page) and 7 parts.
- Question difficulty varies, so don't get stuck on a time-consuming problem.
- Answer each question in the space provided. Partial credit may be given on certain problems.
- You may choose to detach the scratch paper at the end of this booklet. We will not be grading the scratch paper.

Question	Points
1	20
2	20
3	15
4	15
5	15
6	15
EC	0
Total:	100

Honor Code and Academic Integrity Certification

The Honor Code is the commitment and work of students individually and as a community to uphold honesty and integrity. This commitment is comprehensive. Specifically on exams, it means not giving or receiving any help, or using any resources that are not permitted. The Honor Code requires that students take an active role in seeing to it that others as well as themselves uphold the letter and spirit of the Honor Code.

By signing I certify that:

- All the work submitted in my name for this exam is my work alone.
- I have not given or received any help during this exam.
- I have not used any resources during this exam aside from my own page of notes.
- I will inform course staff immediately about any Honor Code violations I become aware of.

I understand that a false statement on this declaration is a violation of the Honor Code, and would be subject to the highest level of disciplinary sanction under the University Code of Conduct.

Signature: _____

Date: _____

1. (Total : 20 points) True / False

   Mark each statement True or False. (1 point each)

   (1a) T ◯ F ◯ **[Soln: T]** Two spectral power distributions are metamers if they cause the same responses for the L, M and S cone cells.

   (1b) T ◯ F ◯ **[Soln: F]** The appearance of a color is independent of the colors surrounding it.

   (1c) T ◯ F ◯ **[Soln: F]** Most people find it more intuitive to generate a dark orange color with R, G, B sliders rather than H, S, V sliders.

   (1d) T ◯ F ◯ **[Soln: T]** Estimating the difference between two colors by Euclidean distance between their color coordinates will be more perceptually uniform in CIELAB rather than CIE XYZ.

   (1e) T ◯ F ◯ **[Soln: T]** These 2 photographic exposures are the same: 1/200th of a second at F4, compared to 1/100th of a second at F2.

   (1f) T ◯ F ◯ **[Soln: F]** Two lenses with the same F-number will have the same aperture diameter in millimeters.

   (1g) T ◯ F ◯ **[Soln: T]** The depth of field is larger when the camera is focused for a magnification of 0.05 compared to a magnification of 0.5.

   (1h) T ◯ F ◯ **[Soln: T]** The light field inside the camera is a 4D function of the radiance flowing along each ray from the lens aperture to the sensor.

   (1i) T ◯ F ◯ **[Soln: F]** In JPEG image encoding, one of the lossy compression steps is to subsample the luma channel by a factor of 2 or more.

   (1j) T ◯ F ◯ **[Soln: F]** Forward Euler integration is a commonly used algorithm for computing time steps in computer simulations, and can usually be made stable by using small time steps.

   (1k) T ◯ F ◯ **[Soln: T]** Inverse kinematics problems rarely have closed form solutions so they are often solved using numerical optimization methods such as gradient descent.

   (1l) T ◯ F ◯ **[Soln: T]** Some motion capture systems use multiple cameras and markers that reflect infrared radiation so that the 3D location of each marker can be computed using the video from the cameras.

   (1m) T ◯ F ◯ **[Soln: T]** Performing a uniform scaling and then a rotating transform is the same as performing the rotation first followed by the scaling.

   (1n) T ◯ F ◯ **[Soln: F]** When starting to rasterize a new frame, we must reset values in the Z-buffer to zero.

   (1o) T ◯ F ◯ **[Soln: T]** In the Painter's Algorithm for hidden-surface removal, objects are rasterized into the framebuffer from furthest to closest.

   (1p) T ◯ F ◯ **[Soln: T]** If a piece of mesh has 2 extraordinary vertices and 3 non-quad faces, after 5 levels of Catmull-Clark subdivision there will be 5 extraordinary vertices and zero non-quad faces.

   (1q) T ◯ F ◯ **[Soln: T]** Cubic Hermite interpolation can be used to interpolate a path from a starting point to an ending point in $\mathbb{R}^N$, given the derivative vectors at the start and end.

   (1r) T ◯ F ◯ **[Soln: F]** A microfacet BRDF with a concentrated normal distribution function will appear matte.

   (1s) T ◯ F ◯ **[Soln: T]** One nit is equal to one candela per square meter.

   (1t) T ◯ F ◯ **[Soln: T]** The path tracing algorithm estimates the solution to the Rendering Equation.

2. (Total : 20 points) Color Reproduction for Aliens

You magically transport to an alien planet. They are almost exactly like us humans in every way, except that their color perception is different. The species as a whole has three types of cone cells, A, B and C, but each individual alien person is dichromatic, having only two types of cone cells in the retina. One third of the aliens have alphachroma vision (A and B cones only), one third have betachroma vision (B and C cones only), and one third have gammachroma vision (A and C cones only).

The alien engineers have invented displays for their smartphones that contain two types of color pixels. All aliens have exactly the same physical phone, but each alien must select whether they are alphachroma, betachroma or gammachroma in their phone software settings, in order for displayed colors to look correct for them.

You are a color display engineer, and you take it upon yourself to reverse-engineer how the alien smartphones implement their color reproduction. Without loss of generality, you'll derive this for just the alphachroma aliens. Given a spectral power distribution (SPD) vector s of a color in the real world, you will calculate how to reproduce this color on the smartphone for an alphachroma alien, by deriving intensities α_1 and α_2 for the two types of color pixels in the smartphones.

For reference, here is a list of all the variables described above:

- r_A, r_B, r_C: (linearly independent) vectors representing the spectral response of the A, B and C cone cells (similar to what was described in lecture for the spectral response functions of human cone cells).
- s_1, s_2: vectors representing the spectral emission functions for the two types of color pixels in the alien smartphone displays
- s: vector representing the SPD of a real color in the world that is to be reproduced on the display
- α_1, α_2: the scalar intensities for colors s_1 and s_2 that you will derive, to implement color reproduction of s for the alphachroma aliens.

(2a) [2 points] Spectral Power Distribution Emitted by Alphachroma Smartphones.

First, write down an expression for the SPD emitted by an alphachroma alien's smartphones, as a function of α_1 and α_2. You may use any of the variables defined above.

Solution: $\alpha_1 s_1 + \alpha_2 s_2$

(2b) [4 points] Alphachroma Alien Color Response.

Next, write down expressions for the vector color response of the alphachroma alien visual system to the real SPD s and the displayed SPD s_{display}. In these vector color responses, A, B (and C in parts below) are scalars representing the response of corresponding cone cell types in the retina. You may use any of the variables defined above.

Solution:

$$\begin{bmatrix} A \\ B \end{bmatrix}_{\text{real}} = \begin{bmatrix} - & - & r_A & - & - \\ - & - & r_B & - & - \end{bmatrix} \begin{bmatrix} | \\ s \\ | \end{bmatrix}$$

$$\begin{bmatrix} A \\ B \end{bmatrix}_{\text{display}} = \begin{bmatrix} - & - & r_A & - & - \\ - & - & r_B & - & - \end{bmatrix} \begin{bmatrix} - & - & s_1 & - & - \\ - & - & s_2 & - & - \end{bmatrix}^T \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix}$$

Also acceptable:

$$\begin{bmatrix} A \\ B \end{bmatrix}_{\text{real}} = \begin{bmatrix} r_A \cdot s \\ r_B \cdot s \end{bmatrix}$$

$$\begin{bmatrix} A \\ B \end{bmatrix}_{\text{display}} = \begin{bmatrix} r_A \cdot s_1 & r_A \cdot s_2 \\ r_B \cdot s_1 & r_B \cdot s_2 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix}$$

(2c) [6 points] Alphachroma Alien Color Reproduction Formula.

Finally, for the display to reproduce the color of s for the alphachroma aliens, the vector color responses in the previous part must be equal. Set them equal and solve for the intensities α_1 and α_2 as a function of the input s. Try to simplify your answer as much as possible, and multiple final forms will be accepted.

Solution:

$$\begin{bmatrix} - & - & r_A & - & - \\ - & - & r_B & - & - \end{bmatrix} \begin{bmatrix} | \\ s \\ | \end{bmatrix} = \begin{bmatrix} - & - & r_A & - & - \\ - & - & r_B & - & - \end{bmatrix} \begin{bmatrix} - & - & s_1 & - & - \\ - & - & s_2 & - & - \end{bmatrix}^T \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix}$$

$$\begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = \left(\begin{bmatrix} - & - & r_A & - & - \\ - & - & r_B & - & - \end{bmatrix} \begin{bmatrix} - & - & s_1 & - & - \\ - & - & s_2 & - & - \end{bmatrix}^T \right)^{-1} \begin{bmatrix} - & - & r_A & - & - \\ - & - & r_B & - & - \end{bmatrix} \begin{bmatrix} | \\ s \\ | \end{bmatrix}$$

Also acceptable:

$$\begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = \begin{bmatrix} r_A \cdot s_1 & r_A \cdot s_2 \\ r_B \cdot s_1 & r_B \cdot s_2 \end{bmatrix}^{-1} \begin{bmatrix} r_A \cdot s \\ r_B \cdot s \end{bmatrix}$$

(2d) [4 points] There are analogous but different equations for setting the s_1 and s_2 pixel intensities to perform correct color reproductions on the smartphone for betachroma or gammachroma viewers. Give a brief explanation for why an alphachroma smartphone does not produce colors that look correct to a betahchroma or gammachroma viewer.

Solution: The analogous color reproduction equations for betachroma, for example, have r_A and r_B replaced with, respectively, r_B and r_C. These formulas cannot produce the same results for all spectral power distributions s if r_A, r_B and r_C are linearly independent. Therefore, the s_1 and s_2 intensities set by the alphachroma setting on the phone will not match the values required for betachroma color reproduction. A

similar argument applies to gammachroma viewers.

(2e) [4 points] You realize you could upend the alien color display industry by inventing a display containing 3 color pixels that could enable all aliens to see colors correctly on any phone, without different software settings. Explain how to make this technical approach work. Assume the new, third pixel has an emission spectrum s_3.

Solution:

The idea would be to use the three degrees of freedom for the intensity of s_1, s_2 and s_3 to create an output spectral power distribution from the display that is metameric to A, B and C cones simultaneously. (This is analogous to trichromatic human color reproduction in class.) Alphachroma, betachroma and gammachroma aliens would see the display differently, but correctly for their individual color vision.

Not necessarily required for full credit, here are the color reproduction formulas, assuming that the intensities for the three pixels are given by C_1, C_2 and C_3.

$$\begin{bmatrix} C_1 \\ C_2 \\ C_3 \end{bmatrix} = \begin{bmatrix} r_A \cdot s_1 & r_A \cdot s_2 & r_A \cdot s_3 \\ r_B \cdot s_1 & r_B \cdot s_2 & r_B \cdot s_3 \\ r_C \cdot s_1 & r_C \cdot s_2 & r_C \cdot s_3 \end{bmatrix}^{-1} \begin{bmatrix} r_A \cdot s \\ r_B \cdot s \\ r_C \cdot s \end{bmatrix}$$

3. (Total : 15 points) Cameras and Sensors

(3a) [5 points] Antialiasing in Photographic Imaging

A new camera-manufacturer has prototyped a camera, but unfortunately it is producing spatial aliasing patterns, such as moirè when taking pictures with fine details like patterned clothing. They call you for help. Mark each of the following true if it might reduce spatial aliasing, and false if it will not.

(i) T ◯ F ◯ **[Soln:]** Use a lens that is optically sharper.

(ii) T ◯ F ◯ **[Soln:]** Use a sensor with a larger number of pixels.

(iii) T ◯ F ◯ **[Soln:]** Apply a 2×2 box blur convolution to process images read off the sensor.

(iv) T ◯ F ◯ **[Soln:]** Add an optical low-pass filter of birefringent crystal in front of the sensor.

(v) T ◯ F ◯ **[Soln:]** Remove the color filter array on the sensor, which means the camera will only be able to take grayscale photographs.

Solution:

1. (i) is false, because the sharp lens will produce higher spatial frequencies before sampling.
2. (ii) is true, because the sampling rate is increasing.
3. (iii) is false, because you cannot avoid aliasing by filtering out high frequencies *after* sampling
4. (iv) is true – such filters were discussed in class.
5. (v) is true, because it eliminates the lower sampling frequency in the separate R, G, B channels.

(3b) Imaging SNR

Compared to a given phone camera, another standalone, compact camera has a focal length, sensor width and sensor height that are each 2 times larger. The f-number is the same between the cameras. The compact camera has twice as many sensor pixels.

3b.i. [3 points] In this situation, each pixel in the compact camera will receive N times more light than a pixel at the corresponding location in the phone camera. What is N? Write a 1-2 sentence explanation for partial credit.

Solution: $N = 2$. The irradiance on the sensor surface is the same, because the f-number of the lenses are the same. The compact camera has 4 times the area, however, so will collect 4 times as many photons. Since the compact camera has twice as many pixels, the photon count per pixel is 2 times higher in the compact camera.

3b.ii. [1 point] Assume that the phone camera and compact camera in the previous question are photon-shot limited (that is, all sources of device-related read noise are small compared to shot noise). What is the per-pixel SNR improvement of the compact

camera over the phone camera? Express your answer in terms of N if you wish, to avoid cascading errors.

Solution: $\sqrt{2}$. In shot limited situation, the SNR is the square root of the signal level. 2 times higher signal in the compact camera is a $\sqrt{2}$ times improvement in SNR.

3b.iii. [2 points] Continuing the previous problem, if a photo of the same scene is taken with both cameras, and the photos are averaged down (that is, downsampled) to the same display pixel resolution, what is the per-pixel improvement in SNR for the compact camera photo over the phone camera photo? Show your work.

Show your work:

Solution: Factor of 2 improvement in the compact camera photo. As described in solution above, the compact camera sensor captures a total of 4 times as many photons. If these photons are averaged into the same output pixel resolution as the compact camera, the signal level will be 4 times higher in the compact camera, hence a factor of 2 increase in SNR.

(3c) Focus and Field of View.

In this problem, assume a thin lens model.

3c.i. [2 points] You want to take a photo of your friend. You intend to compose the photo as a portrait that shows exactly 1 meter of your friend's height (from waist to top of the head). You take the picture with a lens that has a focal length of 100mm on a camera with a 25x25mm image sensor. Rounded to the nearest meter, how far away from your friend should you stand to take the photo? Show your work. Hint: for this part of the question you can safely assume that the camera is focused at infinity. (meters)

Show your work:

Solution: 4 meters. This is by similar triangles: need $\frac{25}{100} = \frac{0.5 \times 2\text{m}}{z}$, where z is the distance from your friend to the lens in meters.

3c.ii. [2 points] Unfortunately your friend is hit by a shrinking ray gun, and ends up exactly

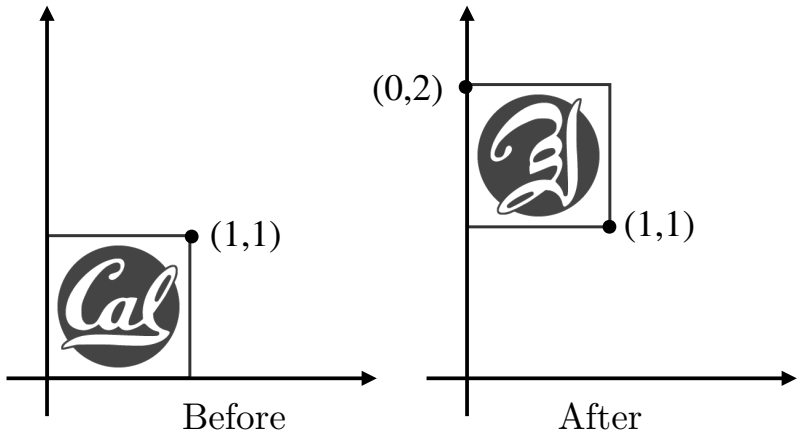
25 mm tall! You resolve to take a photo of your friend to document this tragedy, but now want your friend's full height to exactly fill the photo. How far away from the lens should you place your friend to take the photo? Show your work. Hint: for this part of the question you cannot assume the camera will be focused at infinity. (millimeters)

Show your work:

Solution: 200mm. Magnification desired is unity, since we want the friend to appear 25mm tall on the 25mm sensor. To achieve unit magnification, one can solve the thin lens equation, or recall that this occurs with both the subject and sensor at 2x the focal length.

4. (Total : 15 points) Graphics Pipeline & Geometry

(4a) [6 points] Transforms



Following the conventions in lecture, which of the following 2D transforms will produce the result depicted in the image above? Mark each option as True or False (2 points each).
Assume that ReflectHorizontal is a 2D transform that reflects about the Y axis. Graph paper is included at the end of the exam (optional, and you may detach the graph paper page), though these questions can be answered without.

4a.i. T ◯ F ◯ **[Soln: T]** Translate(1,2) · Rotate(90°) · ReflectHorizontal

4a.ii. T ◯ F ◯ **[Soln: F]** $\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos 90^\circ & -\sin 90^\circ & 0 \\ \sin 90^\circ & \cos 90^\circ & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix}$

4a.iii. T ◯ F ◯ **[Soln: T]** $\begin{bmatrix} 0 & -1 & 1 \\ -1 & 0 & 2 \\ 0 & 0 & 1 \end{bmatrix}$

Solution:

- 4a.i is True.
- 4a.ii is False. Reverse order of 4a.i, not the same.
- 4a.iii is True. Frame-to-World transform.

(4b) Texture Mapping

You are writing an app for browsing panorama photos on your computer. You store the input photos (12,800 x 1280 pixels) as mipmapped texture maps, and display them by drawing texture-mapped rectangles to the framebuffer. During this texture mapping, you apply bilinear filtering using the nearest mipmap level. As a reminder, in mipmapping the full-resolution image is at level 0, and higher levels of the mipmap store lower-resolution versions of the image, e.g. level 1 is a 2 x 2 downsampled version of the full-resolution image.

4b.i. [2 points] If the texture-mapped rectangle you draw covers 3200 x 320 screen pixels and panorama photos fill the rectangle exactly, which level of the mipmap will be used while texture-mapping?

Solution: We will use level 2, because $\log_2 \frac{12800}{3200} = 2$. Note that the image is uniformly scaled down by a factor of 4 x 4.

4b.ii. [3 points] While writing code for a gallery view that displays multiple panorama photos on the screen at once, you accidentally introduce a coding error that is common in graphics programming. Instead of drawing the photos into 1000 x 100 pixel rectangles, your code incorrectly draws the photos into 100 x 1000 pixel rectangles (that is, tall and skinny rectangles, rather than wide and short). To be clear, the photo images are not rotated, but rather the full horizontal field of view of the photo (i.e. originally 12,800 pixels) is squeezed into the 100 pixel rectangle width, and the full vertical field of the photo fills the 1000 vertical pixel rows. What level of the mipmap is being used in texture-mapping these rectangles? (Note: you should assume that the mipmap level is computed as described in lecture.) Explain your answer completely.

Solution: The answer is level 7. Note that the image is scaled down highly non-uniformly, because it is squashed horizontally by a factor of $\frac{12800}{100} = 128$, but scaled down vertically by only a factor of $\frac{1280}{1000} = 1.28$. The mipmapping formula in class avoid antialiasing as far as possible, by using the highest level implied by either horizontal or vertical texture scaling factors. Therefore, we will use mipmap level 7, because $\log_2 \frac{12800}{100} = 7$.

4b.iii. [4 points] Continuing the previous question, the texture-mapped 100 x 1000 rectangles look extremely blurry. Give a full technical explanation about why this is occurring. As part of your answer, compare this situation with what would happen if the programming error was not present and the image was texture-mapped correctly into a 1000 x 100 pixel rectangle (what mipmap level would be used if the error was not present?). Your answer should be 2-3 sentences.

Solution: At mipmap level 7, the stored image texture has a resolution of just 100 x 10 pixels. This is texture mapped into the buggy rectangle of far greater resolution, 100 x 1000 pixels, applying bilinear interpolation, creating a very low-resolution, distorted and blurry result. If it were not for the bug described, the

mipmapped texturing would use level round($\log_2 \frac{12800}{1000}$) = 4, which has a resolution of 800 x 80 pixels, which would match the resolution of the 1000 x 100 rectangle relatively closely and avoid the blurry result.

5. (Total : 15 points) Radiometry, Monte Carlo and Rendering

(5a) Solid Angle

You go to an art exhibition where you stand at a point such that your head is exactly at the center of a glass dome (i.e. hemisphere) with a radius of 10 meters (big!).

5a.i. [2 points] On the surface of the dome, the artist has painted ten, non-overlapping colored disks of one meter radius each. What is the total solid angle, in steradians, of these disk from your perspective?

Solution: The area of the disks is $10 \cdot \pi (1\text{m})^2$, so the solid angle is $10\pi\text{m}^2/(10\text{m})^2 = 0.1\pi$ sr.

5a.ii. [2 points] Artistically, the coverage of the disks seems a bit too sparse to you. How many non-overlapping disks in total would the artist need to have painted to cover half the solid angle of the dome?

Solution: 100 disks in total. The dome covers 2π steradians, as discused in class. Each disk covers 0.01π sr. $\left(\frac{1}{2} \cdot \frac{2\pi}{0.01\pi/\text{disk}}\right) = 100$ disks.

(5b) Russian Roulette

5b.i. [1 point] Why do we use the Russian Roulette method in path tracing? (1 sentence)

Solution: We use Russian Roulette in path tracing to give a finite-time algorithm that gives an unbiased Mont-Carlo integral estimate to the path-tracing integral, which would otherwise require infinite recursion to estimate.

5b.ii. [2 points] How does Russian Roulette work? (1-2 sentences)

Solution: In recursively tracing ray paths to estimate radiance, we randomly terminate the path with probability p. If we do not terminate, we weight the radiance returned by the recursive call by $1/(1-p)$ to keep the integral estimate unbiased.

(5c) [8 points] Monte Carlo Estimators

For each row, you are given a definite integral to evaluate with Monte Carlo integration, the number of random samples (1 or N) to draw, and the probability density function (PDF) to draw the random samples from. Your job is to write down an equation for the corresponding unbiased Monte Carlo estimator in the right-most column. The first two rows are completed as examples for you. In this question, H^2 denotes the hemisphere and S^2 denotes the sphere.

	Definite Integral	**N**	**PDF**	**MC Estimator**
	$\int_a^b f(x)dx$	1	X drawn uniformly at random from $[a, b]$	$(b-a)f(X)$
	$\int_a^b f(x)dx$	N	X_i drawn uniformly at random from $[a, b]$	$\frac{b-a}{N}\sum_{i=1}^{N} f(X_i)$
(a)	$\int_{H^2} f(\omega)d\omega$	N	ω_i drawn uniformly over H^2	
(b)	$\int_0^\pi f(x)dx$	1	X drawn from a pdf proportional to $\sin x$ from $[0, \pi]$	
(c)	$\int_0^\pi \int_0^\pi f(x, y)dxdy$	N	X_i,Y_i each drawn uniformly at random from $[0, \pi]$	
(d)	$\int_{S^2} f(\omega)d\omega$	N	ω_i drawn from a pdf proportional to non-negative function $g(\omega_i)$, $\omega_i \in S^2$.	

Solution:

(a) $\frac{2\pi}{N}\sum_{i=1}^{N} f(\omega_i)$

(b) $\frac{2f(X_i)}{\sin X_i}$. After normalization the PDF is $p(x) = \frac{1}{2}\sin(x)$, because we need $\int_0^\pi p(x)dx = \int_0^\pi \frac{1}{2}\sin x dx = 1$

(c) $\frac{\pi^2}{N}\sum_{i=1}^{N} f(X_i, Y_i)$

(d) $\left(\int_{S^2} g(\omega)d\omega\right)\frac{1}{N}\sum_{i=1}^{N}\frac{f(\omega_i)}{g(\omega_i)}$

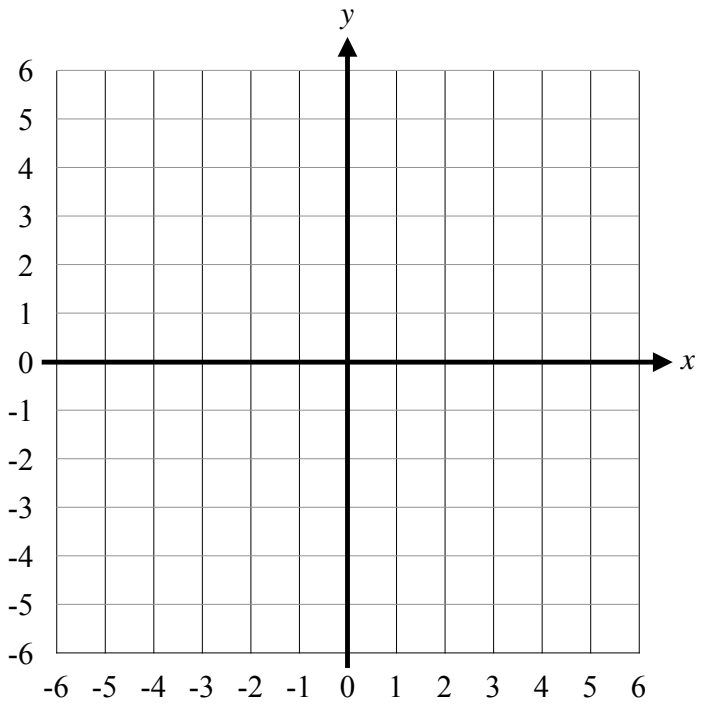
6. (Total : 15 points) Simulation and Virtual Reality

(6a) Physical Simulation

6a.i. [4 points] Euler Steps

There is a particle starting at location $p^{t=0} = [p_x, p_y] = [1, 0]$. The particle is supposed to follow a circular path where its velocity is given by $[v_x, v_y] = [p_y, -p_x]$. (In other words, the velocity is a function of the position and the velocity will be tangent to a circle centered at the origin and having radius $\sqrt{p_x^2 + p_y^2}$.) Forward, explicit Euler method will be used to compute the path of the particle with step size $dt = 1.0$.

On the graph below, plot the initial position plus the results for the first 5 integration steps, and write the coordinates in the answer box below.



Solution: $[1, 0], [1, -1], [0, -2], [-2, -2], [-4, 0], [-4, 4]$

6a.ii. [1 point] Describe what the result would look like if you were to start over with $dt = 0.1$. Be sure to note the difference between this result and the previous result that was $dt = 1.0$.

Solution: Another growing/diverging spiral, but tighter. (Diverging more slowly.)

(6b) Light Field Rendering for Virtual Reality

In this question, we will organize and process the light field data from a camera array, in order to render images corresponding to virtual pinhole cameras at any position in space. This function can be used to render left-eye and right-eye images for stereo viewing in virtual reality.

- As shown in Fig. A below, we will denote each ray in this problem by four coordinates (x, y, s, t), according to the ray's intersections points with two planes.

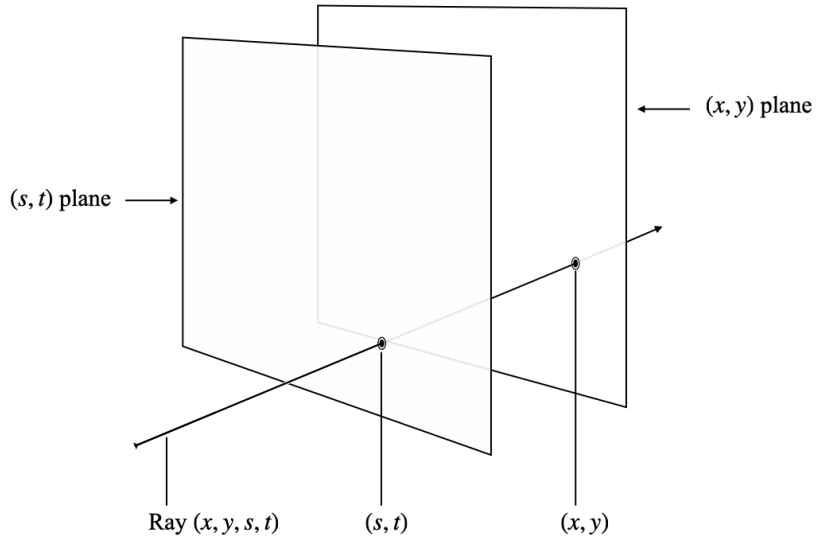


Fig. A

- As shown in Fig. B, our 4D light field data comes from a 2D grid of pinhole cameras on the (s, t) plane. We define (s, t) coordinates so that the grid of cameras exactly fits the unit square $[0, 1] \times [0, 1]$.

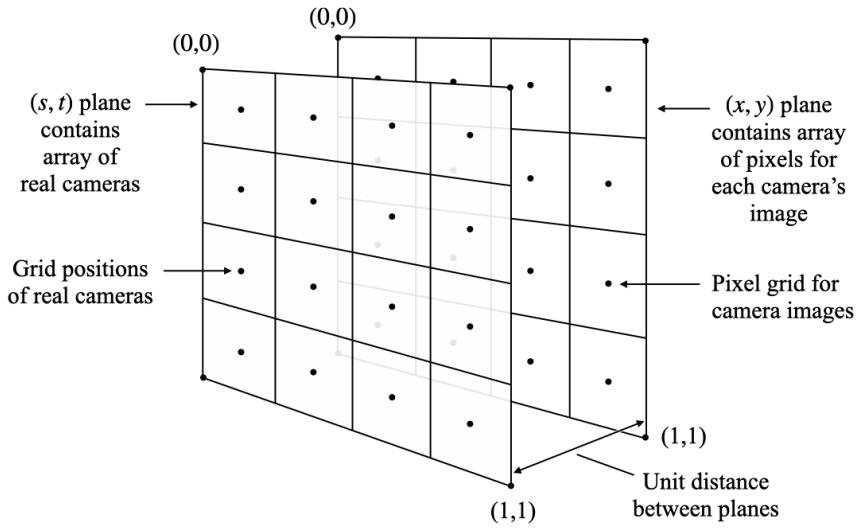


Fig. B

- As shown in Fig. C, every real pinhole camera's image spans the same unit square $[0,1] \times [0,1]$ on the (x, y) plane. In particular, this means that the same pixel in every image is on the same position on the (x, y) plane.

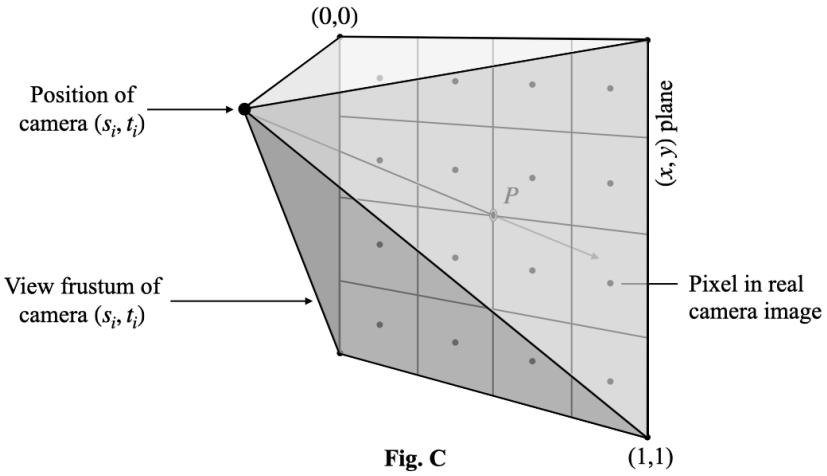


Fig. C

The light field data is stored in a C++ class, with the relevant portion of the class declaration shown here. For simplicity, we assume that the image pixels are scalar grayscale colors, represented as `float` values.

```
class LightField {
private:
    // Number of cameras horizontally & vertically in the grid of cameras
    unsigned int numCamerasS, numCamerasT;

    // The width and height, in pixels, of images taken by each camera
    unsigned int numPixelsX, numPixelsY;

    // A 2D array of camera images
    std::vector<std::vector<float> > cameraImages;

    // A helper function to get the image pixels for camera (si,ti)
    std::vector<float> & GetCameraImage(int si, int ti) {
        return cameraImages[ti * numCamerasS + si];
    }

    // ... Other parts of light field class omitted ....

public:
    // A function that you will implement below
    float NearestNeighborRaySampling(float x, float y,
                                     float s, float t);

    // A function that you will implement below
     void RenderVirtualImage(float viewX, float viewY, float viewZ,
                             unsigned int outputPixelsX,
                             unsigned int outputPixelsY,
                             std:vector<float> &outputPixels);

    // ... Other parts of Light Field class omitted ....
}
```

6b.i. [4 points] Write code for the `NearestNeighborRaySampling` function to sample ray values from the nearest pixel in the nearest camera.

Solution:

```
// Return the color of the nearest pixel in the nearest camera,
// to provide the color of the ray that passes through positions
// (x, y) on the world focal plane, and (s,t) on the camera plane.
// Return black (0.0f) if the ray falls outside the boundary of the
// camera array or outside the boundary of the images.
//
float LightField::NearestNeighborRaySampling(float x, float y,
                                             float s, float t)
{
    int si = Round(s * numCamerasS);
    int ti = Round(t * numCamerasT);
    int xi = Round(x * numPixelsX);
    int yi = Round(y * numPixelsY);

    // Range checking
    if (0 <= si && si < numCamerasS &&
        0 <= ti && ti < numCamerasT &&
        0 <= xi && xi < numPixelsX &&
        0 <= yi && yi < numPixelsY) {

        return GetCameraImage(si, ti)[yi * numPixelsX + xi];
    }

    return 0.0f;
}
```

6b.ii. [6 points] Write code for the `RenderVirtualImage` function, to compute an image for a virtual pinhole camera as shown in Fig.D below. The virtual pinhole is at position (V_x, V_y, V_z), where V_z is defined as the perpendicular distance from the (x, y) plane towards the (s, t) plane (e.g. the (s, t) plane is at $V_z = 1$). As shown, the virtual camera's image should have a field of view that exactly fits the unit square on the (x, y) plane as shown. Use nearest-neighbor sampling. Hints:

- First, geometrically solve for the intersection points $(x, y)_{i,j}$ and then $(s, t)_{i,j}$ for the ray associated with pixel (i, j), as shown in the figure.
- Use the `NearestNeighborRaySampling` function.

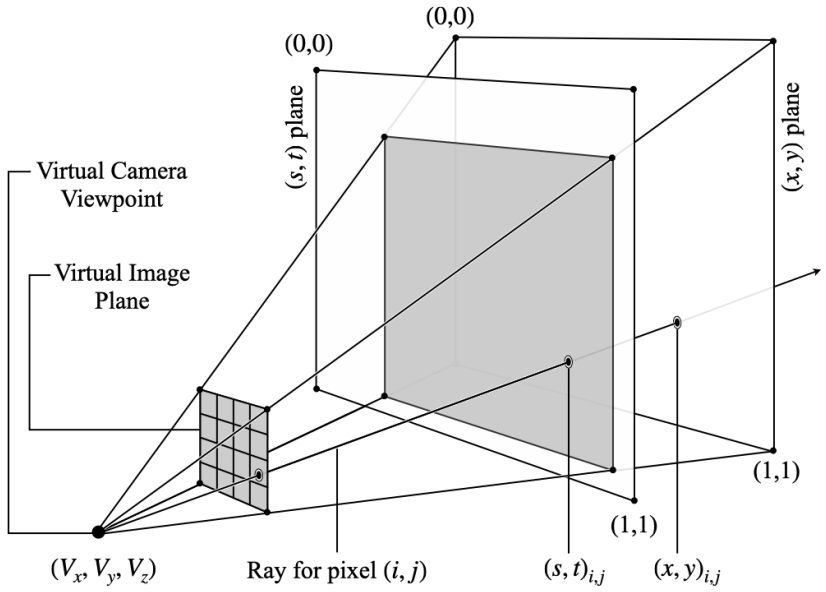


Fig. D

Solution:

```
void LightField::RenderImage(float viewX, float viewY, float viewZ,
                             unsigned int numOutputPixelsX,
                             unsigned int numOutputPixelsY,
                             std:vector<float> &outputPixels)
{
    for (int j = 0; j < numOutputPixelsY; ++j) {
        for (int i = 0; i < numOutputPixelsX; ++i) {
            float x = ((i + 0.5f) / numOutputPixelsX);
            float y = ((j + 0.5f) / numOutputPixelsY);

            float s = x - (x - viewX) / (viewZ - 1);
            float t = y - (y - viewY) / (viewZ - 1);

            float color = NearestNeighborRaySampling(x,y,s,t);

            int index = j * numOutputPixelsY + i;
            outputPixels[index] = color;
        }
    }
```

```
}
```

7. **Optional Extra Credit Challenge Questions (Maximum of 2 Bonus Points Per Part)**

Important: These problems are entirely optional, and are intended as challenges if you finish the exam early. They are worth a paltry amount of bonus points compared to the effort to answer them. **Do not attempt these problems** until you are completely satisfied with the rest of your exam. Any responses to these questions will be graded by the instructor.

(7a) (2 points (bonus)) Vertical Elongation of Reflected Lights in Water

In the photo above, notice a common phenomenon: reflection of lights in the water appear elongated vertically. Give a technical explanation of this optical phenomenon. Hint: consider the distribution of surface normals on the water surface.

(7b) (2 points (bonus)) Multiple Importance Sampling, Balance Heuristic

Monte Carlo (MC) integration is widely used in engineering, science, finance, etc. A powerful and common MC technique called Multiple Importance Sampling (MIS) was originally invented by graphics researchers. MIS allows you to take multiple guesses at the integrand function, importance sample proportional to each of these guesses in turn, and combine all the guesses in a single, unbiased MC estimator that achieves variance close to the best guess.

As an example, consider Monte Carlo integration of $\int_a^b f(x)dx$. In MIS, we draw multiple random samples X_i from probability density functions $p_i(x)$ (note that in this case the pdf can be different for each sample i). We evaluate the function f at each of these samples to obtain values $f(X_i)$, and form a weighted average of these values to estimate the desired integral. This gives the following Monte Carlo integral estimator:

$$F = \sum_{i=1}^{N} w_i(X_i) \frac{f(X_i)}{p_i(X_i)}$$

Many choices of weighting factor $w_i(x)$ are possible. One good choice, called the "balance heuristic", defines $w_i(x) = \frac{p_i(x)}{\sum_{k=1}^{N} p_k(x)}$.

1. Show that the balance heuristic MC estimator is unbiased.
2. The variance of the balance heuristic estimator will be close to the variance of the best $p_i(x)$. This is an excellent property, and has had a huge impact in practical Monte Carlo integration. Explain why the balance heuristic has this property.