Finger Exercises 15

Groups of Objects and Antialiasing

≈ 60 minutes, plus a bonus

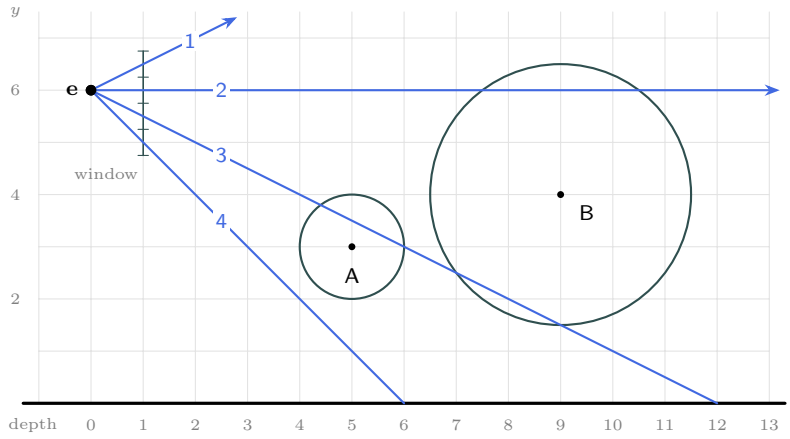
CS 453 — Computer Graphics

You need a pencil, a ruler and a few coloured pencils. Every number was chosen to come out by hand. The letters are the lecture's: the ray $\mathbf{p}(t) = \mathbf{e} + t\,\mathbf{d}$; the hit record with its surface s, its t and its normal $\mathbf{n}$; the outward normal $\mathbf{n}_{\text{out}} = (\mathbf{p} - \mathbf{c})/R$; the interval $[t_0, t_1]$; a random number ξ with $0 \le \xi < 1$; and N samples per pixel. Nothing here is an assignment task; the sheet is Lecture 15 only.

Short on time? Do Exercises 2, 4, 7 and 10 (about 30 minutes): the side a ray came from, one run of the group's loop, three other loops, and one pixel sampled by hand.

Part 1 · What one hit returns

Exercises 1–3



Scene S is used in Parts 1 and 2. The side view shows every point with $x = 0$ at its true size: $-z$ runs to the right and y up.

eye $\mathbf{e} = (0, 6, 0)$
ball A $\mathbf{c} = (0, 3, -5)$, $R = 1$
ball B $\mathbf{c} = (0, 4, -9)$, $R = 2.5$
floor $y = 0$, $\mathbf{n} = (0, 1, 0)$

Four rays leave the eye, one through each pixel of a column of four. Rays 1 to 4 have $\mathbf{d} = (0, v, -1)$ with $v = 0.5,\ 0,\ -0.5,\ -1$.

Every $\mathbf{d}$ has z part -1, so one step of t goes 1 unit into the scene: t **is the depth**, the number under the floor. Lecture 14's two recipes give every crossing of the four rays; they are worked out for you:

	$\mathbf{d}$	ball A	ball B	floor
ray 1	$(0,\ 0.5,\ -1)$	no crossing	no crossing	$t = -12$
ray 2	$(0,\ 0,\ -1)$	no crossing	$t = 7.5$ and 10.5	never: $\mathbf{d} \cdot \mathbf{n} = 0$
ray 3	$(0,\ -0.5,\ -1)$	$t = 4.4$ and 6	$t = 7$ and 9	$t = 12$
ray 4	$(0,\ -1,\ -1)$	no crossing	no crossing	$t = 6$

1 Fill the record

about 4 min

What a hit hands back: the surface, t and the normal. And what a miss hands back.

For each ray, write the record of its *first* hit, the nearest crossing with $t \ge 0$, and mark the hit point in the picture. Ray 3's card is done. ($\mathbf{p}$ is on the card because $\mathbf{n}$ needs it.)

hit record — ray 3
s: ball A
t: 4.4
$\mathbf{p}$: $(0,\ 3.8,\ -4.4)$
$\mathbf{n}$: $(0,\ 0.8,\ 0.6)$

hit record — ray 1
s: _____
t: _____
$\mathbf{p}$: _____
$\mathbf{n}$: _____

hit record — ray 2
s: _____
t: _____
$\mathbf{p}$: _____
$\mathbf{n}$: _____

hit record — ray 4
s: _____
t: _____
$\mathbf{p}$: _____
$\mathbf{n}$: _____

(a) On ray 1's card, which lines stay empty? How does a program know that it must not read them?

(b) Ray 3 also crosses ball B and the floor. Why are they not on its card?

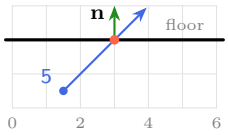
(c) Ray 3 leaves ball A at $t = 6$ and goes into ball B at $t = 7$; ray 2 leaves ball B at $t = 10.5$. Mark these three points too. Then draw the outward normal at every ball crossing you have marked, five in all: a short arrow, straight out from the ball's centre. Exercise 2 needs them.

2 Which side?

about 5 min

The sign of $\mathbf{d} \cdot \mathbf{n}_{out}$: the outward normal points against the ray, or along it.

A ray that goes into a ball travels straight on inside it and meets the ball again on its way out. So each ball crossing below is a hit of its own: the first from outside, the second from inside. Ray 5 is new: it comes from under the floor, with $\mathbf{d} = (0, 1, -1)$, and is drawn on the right.



	the hit, and its point $\mathbf{p}$	$\mathbf{n}_{\text{out}}$	$\mathbf{d} \cdot \mathbf{n}_{\text{out}}$	came from	`front_face`	stored $\mathbf{n}$
	ray 3 into A, at $(0, 3.8, -4.4)$	$(0, 0.8, 0.6)$	$-0.4 - 0.6 = -1$	outside	true	$(0, 0.8, 0.6)$
1	ray 3 out of A, at $(0, 3, -6)$					
2	ray 3 into B, at $(0, 2.5, -7)$					
3	ray 2 out of B, at $(0, 6, -10.5)$					
4	ray 4 on the floor, at $(0, 0, -6)$					
5	ray 5 on the floor, at $(0, 0, -3)$					

(a) Before any arithmetic, look at your arrows from Exercise 1(c). At each ball crossing of rows 1 to 3, is the angle between $\mathbf{d}$ and $\mathbf{n}_{\text{out}}$ more or less than a right angle? Predict the sign of $\mathbf{d} \cdot \mathbf{n}_{\text{out}}$.

(b) Fill the table. The floor has no inside, but it has two sides: its normal $\mathbf{n} = (0, 1, 0)$ plays the part of $\mathbf{n}_{\text{out}}$, and "outside" means "from above".

(c) The worked row and row 1 are one ray through one ball. Compare their two dot products. Why must they be opposite? (Hint: the triangle $\mathbf{c}$, way in, way out.)

(d) In the last column every stored $\mathbf{n}$ gives $\mathbf{d} \cdot \mathbf{n} < 0$. What would be lost if `front_face` were not stored?

(e) A ball could also tell the side from its two roots. How? Why can the floor not do that?

3 Use the record

about 5 min

Why the stored normal is flipped at a hit from inside: the code that uses the normal needs it on the ray's side.

Lecture 16's lighting rule: brightness $= \max(0,\ \mathbf{n} \cdot \mathbf{l})$, with $\mathbf{l}$ the unit arrow from the hit toward the lamp. **Program X** stores $\mathbf{n}_{\text{out}}$ in every record and never flips it. **Program Y** stores the normal of the side that was hit, as in the lecture.

	the hit	the lamp, and $\mathbf{l}$	$\mathbf{n}_{\text{out}} \cdot \mathbf{l}$	program X	stored $\mathbf{n} \cdot \mathbf{l}$	program Y
1	ray 3 into B	the sun, straight overhead: $(0, 1, 0)$				
2	ray 2 out of B	a lamp at B's centre: $(0, -0.8, 0.6)$				
3	ray 3 out of A	a lamp at A's centre: $(0, 0, 1)$				

(a) Fill the table: under each program, the brightness it gives the hit. Exercise 2 has every $\mathbf{n}_{\text{out}}$ and stored $\mathbf{n}$. Check row 2's $\mathbf{l}$: it is $(\mathbf{c} - \mathbf{p})/R$ for ball B.

(b) Rows 1 and 2 both give program X a zero. One zero is right and one is wrong. Which is which, and why?

(c) A lamp hangs at the centre of a hollow ball. Show that $\mathbf{l} = -\mathbf{n}_{\text{out}}$ at every point of the inner wall. What brightness does each program give the whole inner wall?

(d) Program X can be repaired without changing what it stores. What must its lighting code do before it uses a normal? Where does program Y do that work, and how many times?

Part 2 · A group of surfaces

Exercises 4–7

The group's loop, as in the lecture. Each surface's own `hit` answers with its first crossing inside $[t_0, t_1]$, or with a miss: a record whose t is ∞.

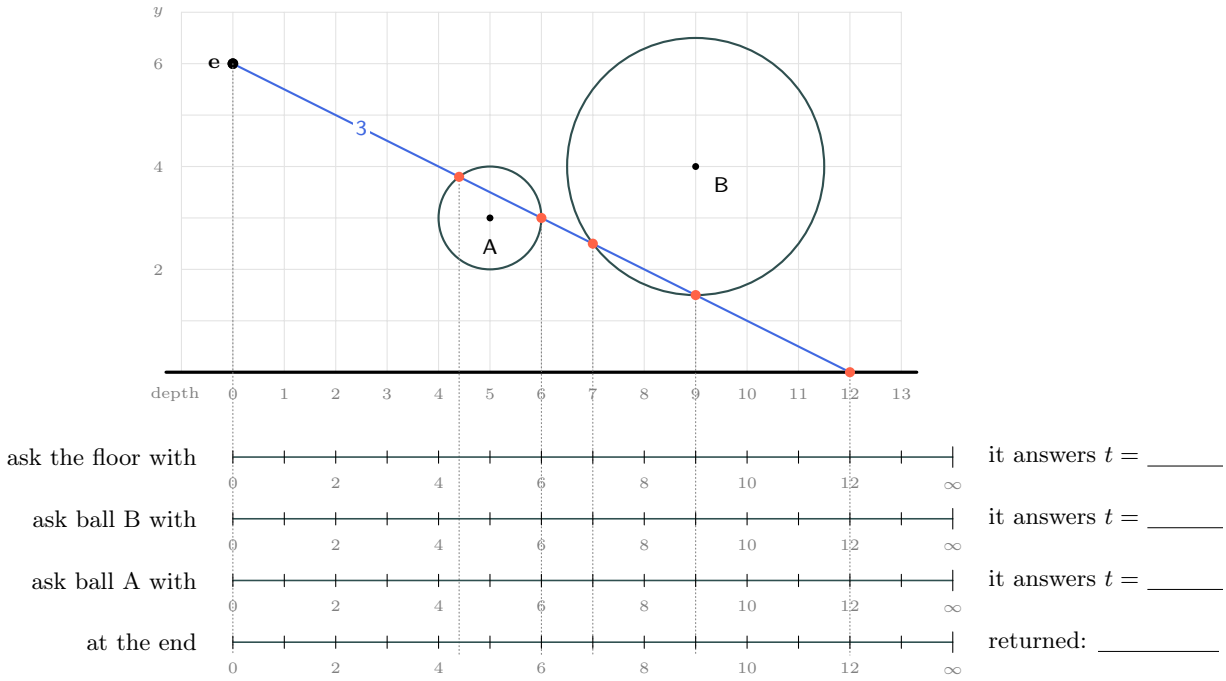
```
def hit(self, ray, t0 = 0.00001, t1 = math.inf):   # a group is asked like any surface
    closest_hit = HitRecord(None, math.inf, Vec3(0,0,0), True)   # a miss
    for surf in self.surface_list:           # ask each surface, in list order
        rec = surf.hit(ray, t0, t1)          # only hits in [t0, t1]
        if rec.t < math.inf:                 # something was hit
            closest_hit = rec                # keep this record
            t1 = rec.t                       # shrink the interval
    return closest_hit                       # the nearest hit, or a miss
```

4 Shade the band

about 7 min

One run of the loop, drawn: the interval each surface is asked with, and what it answers.

The group's list is: floor, ball B, ball A. It is asked about ray 3 with $t_0 = 0$ and $t_1 = \infty$. The red dots are ray 3's five crossings, and the dotted lines carry them down to four copies of the t-axis.



(a) On each of the first three axes, shade the interval $[t_0, t_1]$ that the surface is asked with, and write its answer. On the last axis, shade the interval at the end, and write whose record the group returns.

(b) How many times did the group keep a new record? Which records were thrown away again?

(c) The loop never compared 7 with 12. Who did?

(d) Ball B's hit at $t = 7$ was real, and still its record was thrown away. Which part of ball B's work was wasted?

(e) Run the loop for rays 1, 2 and 4 in your head, with the same list. (Why does the floor answer rays 1 and 2 with a miss?) Then colour the column of four pixels on the right: ball A is orange, (1, 0.4, 0.2); ball B blue, (0.2, 0.5, 0.7); the floor sand, (0.9, 0.8, 0.6); the sky light blue, (0.6, 0.8, 1).

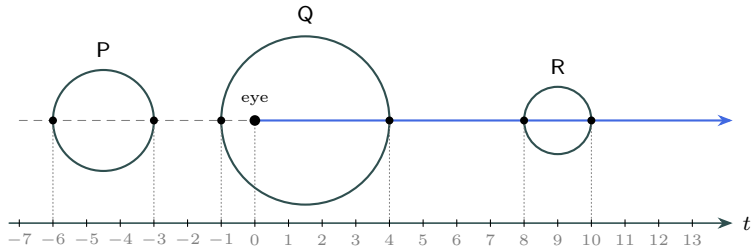
ray 1
ray 2
ray 3
ray 4

5 Two strips

about 6 min

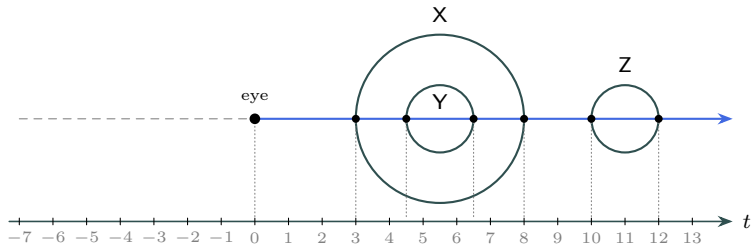
The same loop with a sphere behind the eye, around the eye or inside another one, and with a short interval.

Each strip shows one ray from the eye (the black dot) and three spheres centred on its line. A small dot is a crossing; read its t on the axis under it. The dashed part of the line is behind the eye.



Trace 1. Order P, Q, R; `hit(ray, 0, ∞)`
t_1 after each: ____, ____, ____
returned: _____

Trace 2. Order R, Q, P; `hit(ray, 0, ∞)`
t_1 after each: ____, ____, ____
returned: _____



Trace 3. Order Y, Z, X; `hit(ray, 0, ∞)`
t_1 after each: ____, ____, ____
returned: _____

Trace 4. Order Y, Z, X; `hit(ray, 0, 2)`
t_1 after each: ____, ____, ____
returned: _____

(a) Fill the four traces.

(b) Strip 1: where is the eye? Which of its two roots does Q answer with, and what is `front_face` in the record that comes back?

(c) Strip 2: Y lies inside X. Can any ray that starts outside X ever get Y's record back? Why?

(d) Trace 4 returns a miss, although the ray's line crosses all three spheres. Lecture 11 used such a short interval. For which rays, and why must they stop early?

6 Orders to order

about 4 min

The order of the list changes the work, never the answer.

Ray 3 again. Its first crossings are: ball A at 4.4, ball B at 7, the floor at 12. A list of three surfaces can stand in six orders.

order of the list	t_1 after each surface	changes of t_1	returned
floor, B, A	12, 7, 4.4	3	A
A, B, floor	____, ____, ____	_____	_____
A, floor, B	____, ____, ____	_____	_____
B, A, floor	____, ____, ____	_____	_____
B, floor, A	____, ____, ____	_____	_____
floor, A, B	____, ____, ____	_____	_____

(a) Fill the table. Which orders change t_1 only once? Say the rule in words: when does a surface change t_1?

(b) A ray crosses all n surfaces of a list. What are the fewest and the most changes of t_1, and which orders give them? (Averaged over all orders it is $1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}$: check with your six rows.)

(c) "Nearest first" is the cheapest order for ray 3. A second eye stands far behind ball B and looks back toward ball A. Which order is cheapest for its rays? Why does nobody sort the list?

(d) Two surfaces answer with exactly the same t. Whose record does the loop keep: the earlier one's in the list, or the later one's? (Is $t = t_1$ inside $[t_0, t_1]$?)

7 Three other loops

about 6 min

What shrinking t_1 is for: run three other loops and see what each one hands back.

Three students wrote the body of the group's loop in other ways. Each starts with `closest_hit = HitRecord(None, math.inf, Vec3(0,0,0), True)`, and each is asked about ray 3 (A at 4.4 and 6, B at 7 and 9, floor at 12) with $t_0 = 0$ and $t_1 = \infty$. For both lists below, the lecture's loop returns ball A's record, $t = 4.4$.

Loop 1

```
for surf in self.surface_list:
    rec = surf.hit(ray, t0, t1)
    if rec.t < math.inf:
        closest_hit = rec
return closest_hit
```

list B, A, floor returns: _____

list floor, B, A returns: _____

right for every order? _____

Loop 2

```
for surf in self.surface_list:
    rec = surf.hit(ray, t0, t1)
    if rec.t < math.inf:
        return rec
return closest_hit
```

list B, A, floor returns: _____

list floor, B, A returns: _____

right for every order? _____

Loop 3

```
for surf in self.surface_list:
    rec = surf.hit(ray, t0, t1)
    if rec.t < closest_hit.t:
        closest_hit = rec
return closest_hit
```

list B, A, floor returns: _____

list floor, B, A returns: _____

right for every order? _____

(a) Run each loop on the two lists, and fill its three blanks.

(b) For each wrong loop: which member of the list wins? Two balls overlap in a picture, the nearer one in front. What does each wrong loop draw where they overlap?

(c) Loop 3 is right. What does the lecture's loop save by shrinking t_1 instead? (What must ball B do when it is asked with $[0,\ 4.4]$, and what when it is asked with $[0,\ \infty)$?)

(d) A fourth student shrank the wrong end: `t0 = rec.t` in place of `t1 = rec.t`. What does that loop return for ray 3, in any order? What would its picture of Scene S show?

Part 3 · Antialiasing

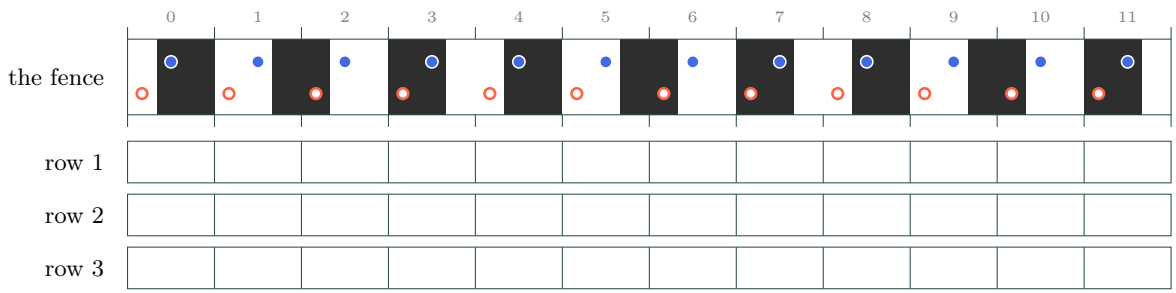
Exercises 8–11

8 A far fence

about 6 min

One ray per pixel against the pixel's whole area: what point sampling does to fine detail.

A fence of black and white bars stands far away: in the picture each bar is only $\frac{2}{3}$ of a pixel wide. One row of twelve pixels looks at it. A blue dot is a pixel's centre, where its one ray goes. A red ring shows where that centre will lie on the fence once the fence has moved $\frac{1}{3}$ of a pixel to the right.



(a) Row 1: one ray through each centre, the blue dots. Write B or W in each pixel. How wide are the stripes in your row? How wide are the bars?

(b) Row 2: the same for the red rings, after the move. How far did your stripes move, and which way?

(c) Row 3: which share of each pixel is white, before the move? (Count thirds of a pixel.) Many samples, averaged, would give this number.

(d) From far away our eyes see the fence as an even gray, $\frac{1}{2}$. Which row is closer to that, row 1 or row 3? Is row 3 perfect? What does that say about the lecture's square, the "simplest choice"?

(e) What would rows 1 and 3 show if every bar were exactly $\frac{1}{2}$ pixel wide?

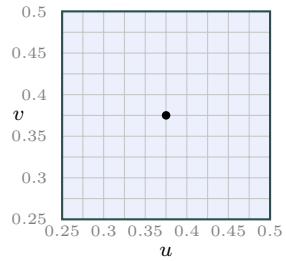
9 From ξ to a ray

about 4 min

Two random numbers place one sample: centre plus offset is $i + \xi$.

The camera is the lecture's: the eye at the origin, looking along $-z$, the window 1 unit ahead. The window's left edge is $l = -1$, its bottom edge $b = -0.75$, and its pixels have side 0.25. We sample pixel $(i, j) = (5, 4)$, drawn large beside the table. A sample goes through $u = l + (i + \xi_1) \times 0.25$ and $v = b + (j + \xi_2) \times 0.25$, and its ray has the direction $(u,\ v,\ -1)$.

$(\xi_1,\ \xi_2)$	offset $(\xi_1 - 0.5,\ \xi_2 - 0.5)$	$(i + \xi_1,\ j + \xi_2)$	$(u,\ v)$
(0.5, 0.5)	(0, 0)	(5.5, 4.5)	(0.375, 0.375)
(0.2, 0.6)			
(0.6, 0.2)			
(0, 0)			



(a) Fill the table, mark the three sample points in the large pixel, and write the direction of each sample's ray.

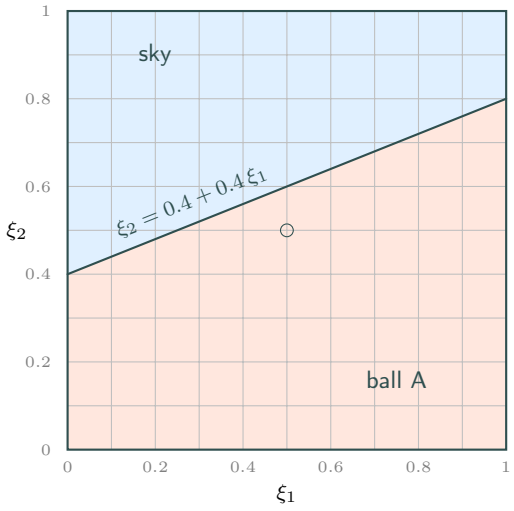
(b) Which ξ gives lecture 13's ray through the centre? Can a sample of pixel (5, 4) land on its right edge, $u = 0.5$? Which pixel owns that edge?

(c) Backwards: a sample ray goes through $(u, v) = (-0.45,\ 0.1)$. Which pixel sent it, with which ξ?

10 Be the sampler

about 9 min

Random samples in one pixel: count them, average their colours, and compare with the true share.



One pixel on the edge of ball A, drawn large. The ball's edge crosses it as a straight line from (0, 0.4) to (1, 0.8). Below the line the pixel shows ball A, (1, 0.4, 0.2); above it the sky, (0.6, 0.8, 1). A sample with the random numbers $(\xi_1,\ \xi_2)$ lands at the point $(\xi_1,\ \xi_2)$ of this square. Each row of the table holds four samples.

row	ξ_1	ξ_2	ξ_1	ξ_2	ξ_1	ξ_2	ξ_1	ξ_2
0	0.13	0.78	0.89	0.96	0.83	0.67	0.31	0.34
1	0.94	0.32	0.37	0.93	0.09	0.84	0.57	0.38
2	0.59	0.87	0.50	0.50	0.99	0.15	0.33	0.28
3	0.40	0.45	0.33	0.46	0.80	0.80	0.66	0.19
4	0.20	0.70	0.85	0.84	0.35	0.21	0.01	0.83
5	0.08	0.15	0.76	0.43	0.03	0.10	0.35	0.26
6	0.48	0.51	0.74	0.56	0.77	0.12	0.82	0.87
7	0.14	0.74	0.75	0.81	0.81	0.46	0.23	0.12
8	0.95	0.62	0.65	0.86	0.24	0.34	0.57	0.79
9	0.27	0.62	0.36	0.64	0.32	0.13	0.15	0.11

(a) The true share: which part of the pixel's area lies below the line? Which colour should a perfect pixel have?

(b) One ray through the centre, the small circle: which colour does the pixel get?

(c) Your row is the last digit of your roll number. Plot its four samples. How many land on ball A? Add the four colours, then divide by 4.

(d) Now 16 samples: your row and the three rows after it (after row 9 comes row 0). Plot them, count, and find the pixel's colour.

(e) Which of (b), (c), (d) came closest to (a)? Compare counts with a classmate who has another row. Which shares can four samples report? Is 60% one of them?

(f) Lecture 12 put its samples at the middles of equal pieces. Cut the pixel into 2×2 equal cells and put one sample at the middle of each. Write the four $(\xi_1,\ \xi_2)$. How many land on ball A? Where do the samples of an $n \times n$ grid go?

11 To bytes, and the bill

about 4 min

Each colour number x is clamped into $[0,\ 0.999]$; its byte is the whole part of $256\,x$. And what N samples cost.

x	0.5	0.85	0.55	1	1.25	-0.1	0.004
after the clamp	0.5						
$256\,x$	128						
byte	128						

(a) Fill the table. Sixteen samples with ten on ball A gave the pixel $(0.85,\ 0.55,\ 0.5)$ in Exercise 10: which three bytes are stored?

(b) Leave the clamp out. Which byte does $x = 1$ ask for, and why is that a problem? Which of this sheet's four colours would run into it? The whole part of $256\,\xi$ never has this problem for a random number ξ: why not?

(c) Which numbers x become the byte 128? How wide is that range, and is it the same for the bytes 0 and 255?

(d) The bill. A picture has 400×300 pixels, 16 samples per pixel, and a group of 25 surfaces. How many rays? How many calls of a surface's `hit`? At 2 million calls a second, how long does the picture take? And with 1 sample per pixel, and with 100?

Try it tonight

optional, about 10 minutes, no marks

Lecture 14's ball in text characters, now with smooth edges. The eye is at the origin and the ball has radius 3 and centre $(0,\ 0,\ -5)$: the ray $(x,\ y,\ -1)$ hits it when $x^2 + y^2 \le 0.75^2$. Every character is one pixel, and it shows the share of its N samples that hit the ball.

```
import random
W, H, N = 40, 20, 16                  # characters across and down; samples each
ramp = ' .:-=+*#%@'                   # ten levels, from empty to full
for j in range(H - 1, -1, -1):        # the top row first
    row = ''
    for i in range(W):
        hits = 0
        for k in range(N):
            x = -1 + 2 * (i + random.random()) / W    # i + xi, not i + 0.5
            y = -1 + 2 * (j + random.random()) / H
            if x * x + y * y <= 0.5625:               # this sample's ray hits the ball
                hits += 1
        share = hits / N
        row += ramp[int(10 * min(share, 0.999))]
    print(row)
```

1. Run it with $N = 16$, then with $N = 1$. What changes along the edge of the ball? Run $N = 1$ twice: why do the two pictures differ?
2. Write `int(10 * share)` in place of `int(10 * min(share, 0.999))`. Which characters stop the program, and what has that to do with the 0.999 before the bytes?
3. Set $N = 4$ and swap the random samples for a 2×2 grid of cell middles: `(i + (k % 2 + 0.5) / 2)` across and `(j + (k // 2 + 0.5) / 2)` up. Which edge looks cleaner, random or grid? What is still wrong with the grid's edge?

12 Bonus: how many samples are enough? ★

about 10 min

For the brave. Why four samples gave such different answers in Exercise 10, and how slowly random samples settle.

In Exercise 10 the ball covers $p = 0.6$ of the pixel. So each random sample lands on ball A with probability 0.6, whatever the other samples did.

(a) Take $N = 4$. What is the probability that all four samples land on the ball? That none does? So how often does such an edge pixel still come out all ball or all sky?

(b) Fill the table (the count k on the ball has probability $\binom{4}{k} 0.6^k \, 0.4^{4-k}$), and check that the five numbers add up to 1. Which shares are the most likely? Compare with the ten rows of Exercise 10: one of them counts 1, five count 2, three count 3 and one counts 4.

samples on the ball, k	0	1	2	3	4
the share the pixel reports	0%	25%	50%	75%	100%
probability	_____	_____	_____	_____	_____

(c) Statistics gives the typical error of the reported share: $\sqrt{p\,(1-p)/N}$. Work it out for $N = 4$, 16, 64 and 256. What does multiplying the number of rays by 4 buy?

(d) A byte has 256 levels. How many random samples bring the typical error below one level, $1/256$? Lecture 12's evenly spaced samples were off by at most $1/(2N)$: how many of those would do?

(e) The 2×2 grid of Exercise 10(f) reports 50% for this pixel every time. Along an edge, which looks worse: every pixel wrong in the same way, or every pixel wrong in its own way? How could a sampler keep the grid's even spread and the random sampler's variety?

Answer Key

Check your work *after* attempting everything. If an answer surprises you, that item is worth re-reading in the slides.

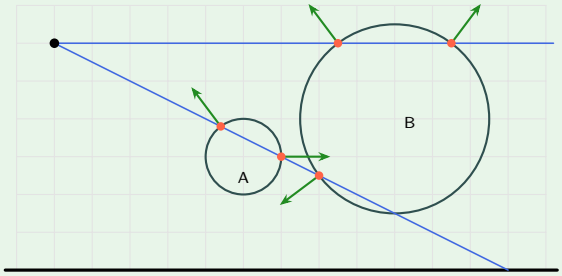
1.

card	s	t	$\mathbf{p}$	$\mathbf{n}$
ray 1	nothing	∞	—	—
ray 2	ball B	7.5	(0, 6, −7.5)	(0, 2, 1.5)/2.5 = (0, 0.8, 0.6)
ray 4	the floor	6	(0, 0, −6)	(0, 1, 0), the floor's own normal

1a. s, $\mathbf{p}$ and $\mathbf{n}$: a miss fills in only $t = \infty$. The program tests "is t less than ∞?" first, and reads the other lines only when the answer is yes.

1b. The card holds the first hit, the smallest t that is not negative: 4.4. Ball B (7) and the floor (12) lie behind ball A on this ray, so this pixel cannot see them.

1c.



The five crossings are (0, 3.8, −4.4) and (0, 3, −6) on ball A, and (0, 2.5, −7), (0, 6, −7.5) and (0, 6, −10.5) on ball B. Each green arrow continues the line from the ball's centre through the crossing. Where a ray goes in, the arrow leans back toward the eye; where a ray comes out, it leans forward, the way the ray is going.

2.

	$\mathbf{p} - \mathbf{c}$	$\mathbf{n}_{\text{out}}$	$\mathbf{d} \cdot \mathbf{n}_{\text{out}}$	came from	`front_face`	stored $\mathbf{n}$
1	(0, 0, −1)	(0, 0, −1)	$0 + 1 = 1$	inside	false	(0, 0, 1)
2	(0, −1.5, 2)	(0, −0.6, 0.8)	$0.3 - 0.8 = -0.5$	outside	true	(0, −0.6, 0.8)
3	(0, 2, −1.5)	(0, 0.8, −0.6)	$0 + 0.6 = 0.6$	inside	false	(0, −0.8, 0.6)
4	the floor	(0, 1, 0)	$-1 + 0 = -1$	above	true	(0, 1, 0)
5	the floor	(0, 1, 0)	$1 + 0 = 1$	below	false	(0, −1, 0)

2a. On the way in (row 2), $\mathbf{n}_{\text{out}}$ points back toward where the ray came from: more than a right angle with $\mathbf{d}$, so the dot product is negative. On the way out (rows 1 and 3), $\mathbf{n}_{\text{out}}$ points ahead, the way the ray is going: less than a right angle, positive.

2c. −1 and +1. The line from $\mathbf{c}$ perpendicular to the ray cuts the ray's path through the ball in half. So the way in lies as far before that middle point as the way out lies after it: $\mathbf{d} \cdot (\mathbf{p} - \mathbf{c})$ has the same size and the opposite sign. The same holds on ball B: ray 3 goes in with −0.5 (row 2) and comes out with +0.5; ray 2 goes in with −0.6 and comes out with +0.6 (row 3).

2d. The side. After the flip, $\mathbf{d} \cdot \mathbf{n}$ is negative at every hit, so the stored normal can no longer say whether the ray was outside or inside. That is why the record keeps `front_face`.

2e. If the first hit is the smaller root, both roots lie ahead and the ray started outside. If it is the larger root, the smaller one is negative: the ray started inside. The floor has only one root, so there is nothing to compare, and yet it has two sides. The dot product needs only $\mathbf{d}$ and the normal, so it works for every shape.

3.

	$\mathbf{n}_{\text{out}} \cdot \mathbf{l}$	program X	stored $\mathbf{n} \cdot \mathbf{l}$	program Y
1	$(0, -0.6, 0.8) \cdot (0, 1, 0) = -0.6$	0	-0.6	0
2	$(0, 0.8, -0.6) \cdot (0, -0.8, 0.6) = -1$	0	$(0, -0.8, 0.6) \cdot (0, -0.8, 0.6) = 1$	1
3	$(0, 0, -1) \cdot (0, 0, 1) = -1$	0	$(0, 0, 1) \cdot (0, 0, 1) = 1$	1

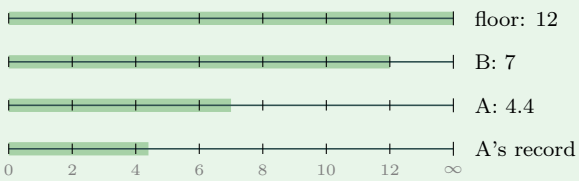
Row 2's $\mathbf{l}$: $\mathbf{c} - \mathbf{p} = (0, 4, -9) - (0, 6, -10.5) = (0, -2, 1.5)$, divided by 2.5.

3b. Row 1 is right: that patch is on the underside of ball B and faces down, away from the sun, so the sun really is behind it. Row 2 is wrong: the lamp is inside the ball and shines straight onto the inner wall, but $\mathbf{n}_{\text{out}}$ points away from it, so the rule takes the lamp to be behind the wall.

3c. $\mathbf{l}$ points from $\mathbf{p}$ to the centre: $(\mathbf{c}-\mathbf{p})/R$, and $\mathbf{n}_{\text{out}} = (\mathbf{p}-\mathbf{c})/R$ is exactly its opposite. So $\mathbf{n}_{\text{out}} \cdot \mathbf{l} = -1$ everywhere: program X draws the inside of a lit lantern black. Program Y's normal is $-\mathbf{n}_{\text{out}} = \mathbf{l}$, brightness 1 everywhere.

3d. Test $\mathbf{d} \cdot \mathbf{n}_{\text{out}}$ and flip the normal when it is positive; and every other piece of code that uses a normal must do the same. Program Y does it once, where the hit is made, and stores the side as `front_face`.

4.



The floor is asked with $[0, \infty)$ and answers 12. Ball B is asked with $[0, 12]$ and answers 7. Ball A is asked with $[0, 7]$ and answers 4.4. At the end the interval is $[0, 4.4]$, and the group returns ball A's record: $t = 4.4$, $\mathbf{n} = (0, 0.8, 0.6)$.

4b. Three times: the floor's record, then ball B's, then ball A's. The floor's and ball B's were thrown away.

4c. Ball B's own `hit`: it was asked only for hits in $[0, 12]$, and 7 is one. In the same way ball A was asked only for hits in $[0, 7]$. The interval does the comparing.

4d. Its point $(0, 2.5, -7)$ and its normal $(0, -0.6, 0.8)$ were worked out and never used. Only the nearest hit's normal is ever needed.

4e.



Ray 1 climbs away from the floor: its line meets the floor 12 steps *behind* the eye, and $t = -12$ is not in $[0, \infty)$. Both balls miss too, so the record stays a miss and the pixel shows the sky. Ray 2 runs level with the floor, $\mathbf{d} \cdot \mathbf{n} = 0$: no crossing at all. Ball B answers 7.5 and ball A misses: ball B. Ray 3: ball A. Ray 4: the floor answers 6, then both balls miss: the floor. From the top: sky, B, A, floor.

5a.

trace	order, call	t_1 after each	returned
1	P, Q, R; $[0, \infty)$	∞, 4, 4	Q's record, $t = 4$
2	R, Q, P; $[0, \infty)$	8, 4, 4	Q's record, $t = 4$
3	Y, Z, X; $[0, \infty)$	4.5, 4.5, 3	X's record, $t = 3$
4	Y, Z, X; $[0, 2]$	2, 2, 2	a miss, $t = \infty$

P is behind the eye: both its roots are negative, a miss. In trace 2, R is hit first and its record is replaced.

5b. Inside Q: its roots -1 and 4 lie on either side of 0. The smaller root is not in the interval, so Q answers with the larger one, 4 (lecture 14's case 2). The ray hits Q from inside: `front_face` is false, and the stored normal is $-\mathbf{n}_{\text{out}}$.

5c. No. To reach Y a ray must first go through X's surface, so X's first crossing is always the smaller one. Y's record comes back only for a ray that starts inside X (with $t_0 = 4$ here, X has no crossing left before Y's 4.5).

5d. Shadow rays. A shadow ray asks only whether something lies between a point and the lamp, so its interval ends at the lamp. A surface beyond the lamp cannot hide it.

6a.

order	t_1 after each	changes	returned
A, B, floor	4.4, 4.4, 4.4	1	A
A, floor, B	4.4, 4.4, 4.4	1	A
B, A, floor	7, 4.4, 4.4	2	A
B, floor, A	7, 7, 4.4	2	A
floor, A, B	12, 4.4, 4.4	2	A

Once: the two orders that start with A. A surface changes t_1 when it is nearer than every surface before it in the list.

6b. Fewest: 1, with the nearest surface first. Most: n, with the list running from the farthest to the nearest. The six rows change t_1 $3 + 1 + 1 + 2 + 2 + 2 = 11$ times: $11/6 = 1 + \frac{1}{2} + \frac{1}{3}$.

6c. For that eye, B is the nearest and A lies behind it: B first. Every ray has its own nearest surface, and one list serves every ray of the picture, so no order is cheapest for all of them. The answer is right in any order, and the unsorted list costs little.

6d. The later one's. The first sets t_1 to that t; the second one's t equals t_1, which is inside $[t_0, t_1]$, so it reports a hit and its record replaces the first. Only in such a tie does the order decide what comes back. (Ray Tracing in One Weekend asks for $t_0 < t < t_1$ instead; there the earlier one stays.)

7a.

	list B, A, floor returns	list floor, B, A returns	right for every order?
loop 1	the floor, $t = 12$	A, $t = 4.4$ (by luck)	no
loop 2	B, $t = 7$	the floor, $t = 12$	no
loop 3	A, $t = 4.4$	A, $t = 4.4$	yes

7b. Loop 1 keeps every hit, so the *last* member of the list that the ray crosses wins. Loop 2 stops at the *first* member that the ray crosses. In both, the list decides which ball is in front, not the distance: where the two balls overlap, the same ball wins everywhere, even when it is the farther one.

7c. In loop 3 every surface is asked with the whole interval, so every surface that the ray crosses builds a full record, point and normal, and then two t values are compared. In the lecture's loop ball B, asked with $[0, 4.4]$, finds its roots 7 and 9 outside the interval and answers "miss" at once.

7d. The floor, $t = 12$, in every order. After a hit at t, only crossings at t or beyond still count, so each kept hit is *farther* than the one before. The picture shows the farthest thing along every ray: the floor shows through both balls.

8.

row 1	B	W	W	B	B	W	W	B	B	W	W	B
row 2	W	W	B	B	W	W	B	B	W	W	B	B
row 3	$\frac{1}{3}$	$\frac{2}{3}$	$\frac{2}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{2}{3}$	$\frac{2}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{2}{3}$	$\frac{2}{3}$	$\frac{1}{3}$

8a. Stripes two pixels wide. The bars are $\frac{2}{3}$ of a pixel wide, three times narrower. The row shows a coarse fence that is not there: the fine pattern comes out under a false name, an alias.

8b. One whole pixel to the *left*: three times as far as the fence itself moved, and the other way. (Three pixels to the right would look the same; neither is the fence's own $\frac{1}{3}$ to the right.) In a film the false stripes would crawl backwards, as the spokes of a turning wheel sometimes seem to do on screen.

8c. $\frac{1}{3}$, $\frac{2}{3}$, $\frac{2}{3}$, $\frac{1}{3}$, and so on: every pixel holds one and a half bars.

8d. Row 3: its values stay near $\frac{1}{2}$, where row 1 has only 0 and 1. But it is not perfect: a faint copy of the false stripes is left, with a third of the contrast. In the square every point counts the same and the square ends sharply. A weight that fades out from the centre and reaches a little into the neighbours flattens the pattern more.

8e. Every pixel would hold exactly one black and one white bar. Row 1: all centres land on the same colour, so the row is all black or all white, depending on where the fence stands. Row 3: $\frac{1}{2}$ everywhere, an even gray. That is the lecture's far checkerboard.

9a.

ξ	offset	$(i+\xi_1,\ j+\xi_2)$	$(u,\ v)$	direction
$(0.2,\ 0.6)$	$(-0.3,\ 0.1)$	$(5.2,\ 4.6)$	$(0.3,\ 0.4)$	$(0.3,\ 0.4,\ -1)$
$(0.6,\ 0.2)$	$(0.1,\ -0.3)$	$(5.6,\ 4.2)$	$(0.4,\ 0.3)$	$(0.4,\ 0.3,\ -1)$
$(0,\ 0)$	$(-0.5,\ -0.5)$	$(5,\ 4)$	$(0.25,\ 0.25)$	$(0.25,\ 0.25,\ -1)$

For example $u = -1 + 5.2 \times 0.25 = 0.3$ and $v = -0.75 + 4.6 \times 0.25 = 0.4$. The third sample sits on the pixel's lower left corner.

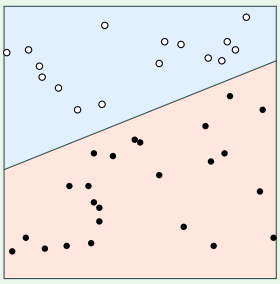
9b. $\xi = (0.5,\ 0.5)$: the offset is zero. Never on the right edge: $\xi_1 < 1$, so $i + \xi_1 < 6$ and $u < 0.5$. That edge belongs to pixel $(6,\ 4)$, as its $\xi_1 = 0$. So every point of the window belongs to exactly one pixel.

9c. $(u - l)/0.25 = 0.55/0.25 = 2.2$ and $(v - b)/0.25 = 0.85/0.25 = 3.4$: pixel $(2, 3)$ with $\xi = (0.2,\ 0.4)$.

10a. A trapezoid with sides 0.4 and 0.8 and width 1: $(0.4 + 0.8)/2 = 0.6$. The perfect colour is $0.6\,(1,\ 0.4,\ 0.2) + 0.4\,(0.6,\ 0.8,\ 1) = (0.84,\ 0.56,\ 0.52)$.

10b. At $\xi_1 = 0.5$ the line is at 0.6, above the centre: the centre is on the ball, and the pixel is all ball, $(1,\ 0.4,\ 0.2)$. That is 100% where 60% is right.

10c, d.



all 40 samples: 24 on the ball

your row	0	1	2	3	4	5	6	7	8	9
on ball A, of 4	2	2	3	3	1	4	3	2	2	2
on ball A, of 16	10	9	11	11	10	11	9	8	8	9

Colours for $1, 2, 3, 4$ of 4: $(0.7,\ 0.7,\ 0.8)$, $(0.8,\ 0.6,\ 0.6)$, $(0.9,\ 0.5,\ 0.4)$, $(1,\ 0.4,\ 0.2)$. For example three on the ball: $3\,(1,\ 0.4,\ 0.2) + (0.6,\ 0.8,\ 1) = (3.6,\ 2,\ 1.6)$, divided by 4.

Colours for $8, 9, 10, 11$ of 16: $(0.8,\ 0.6,\ 0.6)$, $(0.825,\ 0.575,\ 0.55)$, $(0.85,\ 0.55,\ 0.5)$, $(0.875,\ 0.525,\ 0.45)$: add k ball colours and $16-k$ sky colours, then divide by 16.

10e. Sixteen samples, for eight of the ten rows; rows 7 and 8 report 50% both times. Sixteen samples report 50% to 69%; four samples report anything from 25% (row 4) to 100% (row 5): more samples help on the whole, not every time. A count out of 4 can only say 0, 25, 50, 75 or 100%, never 60%; five samples could (3 of 5). All 40 samples of the table together: 24 on the ball, exactly 60%.

10f. $(0.25,\ 0.25)$, $(0.75,\ 0.25)$, $(0.25,\ 0.75)$, $(0.75,\ 0.75)$. The two lower ones are on the ball: 2 of 4, 50%, colour $(0.8,\ 0.6,\ 0.6)$. In an $n \times n$ grid the sample of cell (a, b) sits at $\big((a + 0.5)/n,\ (b + 0.5)/n\big)$, for a and b from 0 to $n - 1$.

11a.

x	0.85	0.55	1	1.25	-0.1	0.004
after the clamp	0.85	0.55	0.999	0.999	0	0.004
$256\,x$	217.6	140.8	255.744	255.744	0	1.024
byte	217	140	255	255	0	1

The pixel $(0.85,\ 0.55,\ 0.5)$ is stored as $(217,\ 140,\ 128)$.

11b. 256, one more than the largest byte, 255. Ball A's red is 1 and the sky's blue is 1: every pixel that is all ball A or all sky would ask for it. A program then stops with an error, or keeps only the lowest eight bits and writes 0: the brightest part of the picture turns dark. A random number has $\xi < 1$, so $256\,\xi < 256$ and its whole part is at most 255: the lecture's "1 never comes out" is the same safeguard.

11c. $128 \le 256\,x < 129$: from 0.5 up to $129/256 \approx 0.5039$, a range $1/256$ wide. The same for every byte: 0 takes $[0,\ 1/256)$ and 255 takes $[255/256,\ 1]$, which is why the top of the clamp is just under 1.

11d. 120,000 pixels, so 1,920,000 rays. Each ray asks all 25 surfaces: 48 million calls, 24 seconds. One sample: 3 million calls, 1.5 seconds. A hundred samples: 300 million calls, 150 seconds.

12a. $0.6^4 = 0.1296$ and $0.4^4 = 0.0256$: together 0.155. About one such pixel in six or seven still comes out all ball or all sky.

12b.

k	0	1	2	3	4
probability	0.0256	$4 \times 0.6 \times 0.064 = 0.1536$	$6 \times 0.36 \times 0.16 = 0.3456$	$4 \times 0.216 \times 0.4 = 0.3456$	0.1296

50% and 75% are equally likely, and neither is right. Of ten rows we expect about 0.3, 1.5, 3.5, 3.5, 1.3 with $k = 0, \ldots, 4$; the table gave 0, 1, 5, 3, 1.

12c. $\sqrt{0.24/N}$: 0.245, 0.122, 0.061, 0.031. Four times the rays buy half the error.

12d. $0.24/N < 1/256^2$ needs $N > 0.24 \times 65536 \approx 15{,}700$ random samples. Evenly spaced along a line: $1/(2N) \le 1/256$ needs only $N = 128$. (In a square the gain is smaller: an $n \times n$ grid can still be off by $1/(2n)$ when an edge runs along its rows.) Random samples settle slowly, which is why a picture with 100 samples per pixel still shows a little noise.

12e. The grid is 10% too low here in the same way every time, so a row of such pixels along an edge is wrong together: a regular pattern, which the eye picks out at once. Random samples make every pixel wrong in its own way: noise, which the eye forgives more easily. To keep both: cut the pixel into $n \times n$ cells and put one *random* sample inside each cell. That is called jittered sampling.

T1. With 16 samples the edge characters step through the ramp, from `.` up to `%`, by how much of each is ball. With 1 sample every character is blank or `@`: a staircase, and a ragged one, because the one sample is at a random point. Two runs draw different random numbers, so the edge changes.

T2. Every character that is all ball: its share is 1, so `int(10 * 1.0)` is 10, and the ramp has only the places 0 to 9: an `IndexError`. The same reason as $256 \times 1 = 256$: the top value must be kept just under 1.

T3. The grid: its edge is smooth and the same on both sides and in every run, where four random samples leave a ragged edge. But the grid can only report 0, $\frac{1}{4}$, $\frac{1}{2}$, $\frac{3}{4}$ or 1, and neighbouring characters make the same mistakes together, so a faint staircase is still there.