Name:

Mid-term examination

Computer Graphics 1 (15-462)

Grade value: 13%

Duration: 1 hour 10 minutes

Answer all questions for full credit

Question 1: Constructive Solid Geometry - CSG (again) (27)

(a) Explain what a parametric function is. Give a practical example and say what parametric functions are useful for and why. (3)

A parametric function is a function that maps a parameter space on to a higher dimension world space.
Example, the parametric equation for a circle is:
$x(\theta) = r\cos(\theta), y(\theta) = r\sin(\theta)$. **$\theta$ is the parameter.**
They are useful for generating regular meshes of surfaces - varying the parameters in a regular manner gives the function's value at regular positions.

(b) Explain what an implicit function is. Give a practical example and say what implicit functions are useful for and why. (3)

An implicit function defines a surface by assigning a value to a given world coordinate - the surface is then defined to exist when this value is zero (typically).
Example, the implicit function of a circle is: $x^2 + y^2 - r^2 = 0$.
They are useful for calculating intersections - the intersection simply requires the solution of an equation.

(c) What are the geometric primitives of CSG? Give some examples. (2)

Simple solids are the primitives. Typical primitives are: cubes, cylinders and spheres.

(d) What are the three basic operations of CSG? (2)

The three basic operations of CSG are: adding together, intersecting and subtracting objects.

(e) What boolean operations implement the basic CSG operations? (2)

Adding is done using the union (OR) operation.
Intersecting is done using the intersect (AND) operation.
Subtraction is done using minus and intersection (unary minus and AND).

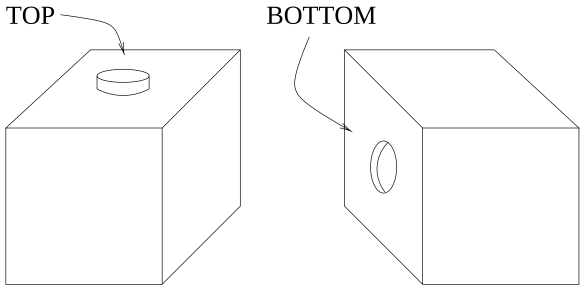
(f) What implicit functions implement the basic CSG operations? (2)

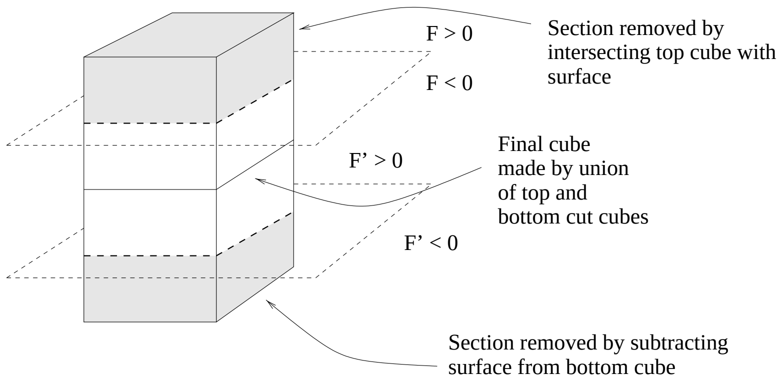
Union uses the minimum implicit function.
Intersection uses the maximum function.
Subtraction uses minus and maximum.

(g) Explain how the implicit functions achieve these operations. (3)

Minimum achieves union - because the resulting function is less than zero (inside the object) when either object's value is than zero - hence the minimum is less than zero.
Maximum achieves intersection - because the resulting function is only less that zero when both object's values are less than zero - hence the maximum of the two is less than zero.
Minus and maximum achieves subtraction because the resulting function is less than zero when the value is inside the main object and outside the removed object.

(h) Imagine a set of building blocks that interlock at the top and bottom faces. Explain how the boolean functions would be used to form a piece of this building block set - you should use all three CSG operations. Assume that your basic building blocks are cubes and that you are given the implicit form of the surface that defines the interlocking shape. (5)




F > 0
F < 0
Section removed by intersecting top cube with surface
F' > 0
F' < 0
Final cube made by union of top and bottom cut cubes
Section removed by subtracting surface from bottom cube

To produce a block that is basically a cube with an inset on the base and a matching lug on the top:
For the top of the cube - intersect the cube and the cutting surface to obtain a partial cube with a suitable lug.
For the bottom of the cube - subtract the cutting surface from the cube.
Then join these two pieces together using a union operation.

(i) To display these building blocks on screen without ray tracing we require the polygonal mesh of each block. Explain how to generate the polygonal mesh for a single block using the method that you explained in part (h). Assume that you are given the polygonal meshes for a cube and for the surface that defines the interlocking shape. (5)

To generate the polygonal mesh for such a building block:

We need the following definitions:

top cutting surface defines the half space $F < 0$,
bottom cutting surface defines the half space $F' < 0$,
the top cube is defined such that $C < 0$ is inside it,
similarly the bottom cube is defined $C' < 0$.

The mesh for the top part of the block:
includes the mesh for the cube inside the cutting surface ($F < 0$) and the mesh of the cutting surface inside the cube ($C < 0$).

The mesh for the bottom part of the block:
includes the mesh of the cube outside the cutting surface ($F' > 0$) and the mesh of the cutting surface inside the cube ($C' < 0$).

The complete mesh: consists of the above two meshes, excluding the parts that are inside the final result, ie those for which:
$min(max(C, F), max(C', -F')) < 0$**.**

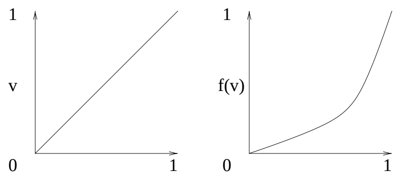
Question 2: Image processing (26)

(a) Explain what point processing is. (1)

The application of a 1-D function or look-up table to the individual pixels of an image.

(b) Given the point processing function $f(v) = v^p$, what is its effect on the image if $p > 1$? How does it work? What range of values must the pixel value v be made to lie between? (3)

For $p > 1$ this function darkens the image. v must be between 0 and 1. It darkens the image by making small (dark) values of v smaller (darker).



(c) Filtering can be applied to any kind of signal. What kind of signal is an image? (1)

A computer image is a 2-D discrete (digital) signal.

(d) What is a linear, shift-invariant filter? (2)

It is a filter that uses only linear operations (ie multiplication and addition (accumulation)) which operates with the same parameters where ever it is applied.

(e) Convolution implements a linear, shift-invariant filter. Explain what convolution is. What is a convolution matrix? (1)

Convolution is the multiplication of a set of fixed values with corresponding pixel values - the results of these multiplications are accumulated (added together) to give a single result. The convolution matrix defines the multiplier values.

(f) Blurring is a convolution. Give a convolution matrix for a 3x3 blur. (1)

$$\frac{1}{9}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

(g) The following convolution matrices form part of the edge filter calculation. What functions of the image values do they calculate? Explain how they calculate these functions. (4)

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

This convolution matrix calculates the horizontal derivative of an image (with some averaging). The derivative is the 1rate of change of the function (values), and is approximated by taking the difference between the two neighboring values. The matrix returns the difference between the pixels to the right of the current pixel and pixels to the left of the current pixel for the current line and its two neighbors. The difference for the current line is weighted twice as heavily as that for the neighbors.

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

This convolution matrix calculates the vertical derivative of an image (with some averaging). The matrix returns the difference between the pixels above and below the current pixel for the current column and its two neighbors. The difference for the current column is weighted twice as heavily as that for the neighbors.

(h) What is an image edge? Why is the full edge filter a non-linear filter? (3)

An image edge is a part of the image that has a high image gradient - ie the pixel values are changing rapidly.
The full edge filter is non-linear because it uses non-linear functions (square and square-root).

(i) Convolve the following image patch with the blurring filter and the two convolution matrices above. Apply each convolution to the original image patch (not the new convolved one). Explain what you do at the edge of the image patch. (10)

100	110	120	123
100	110	120	97
90	100	80	120
85	90	72	100

Ignore the edges of the image patch - the derivative is not easily defined there and it adds unnecessary complexity to the blurring.

Blurring:

$$\begin{array}{|c|c|}\hline \frac{930}{9} & \frac{980}{9} \\ \hline \frac{847}{9} & \frac{7}{9} \\ \hline \end{array} = \begin{array}{|c|c|}\hline 103\frac{1}{3} & 108\frac{8}{9} \\ \hline 94\frac{1}{9} & \frac{889}{9} \\ \hline \end{array}$$

Horizontal derivative:

$$\begin{array}{|c|c|}\hline 440-390 & 437-430 \\ \hline 352-365 & 437-400 \\ \hline \end{array} = \begin{array}{|c|c|}\hline 50 & 7 \\ \hline -13 & 37 \\ \hline \end{array}$$

Vertical derivative:

$$\begin{array}{|c|c|}\hline 440-370 & 473-380 \\ \hline 440-337 & 447-334 \\ \hline \end{array} = \begin{array}{|c|c|}\hline 70 & 93 \\ \hline 103 & 113 \\ \hline \end{array}$$

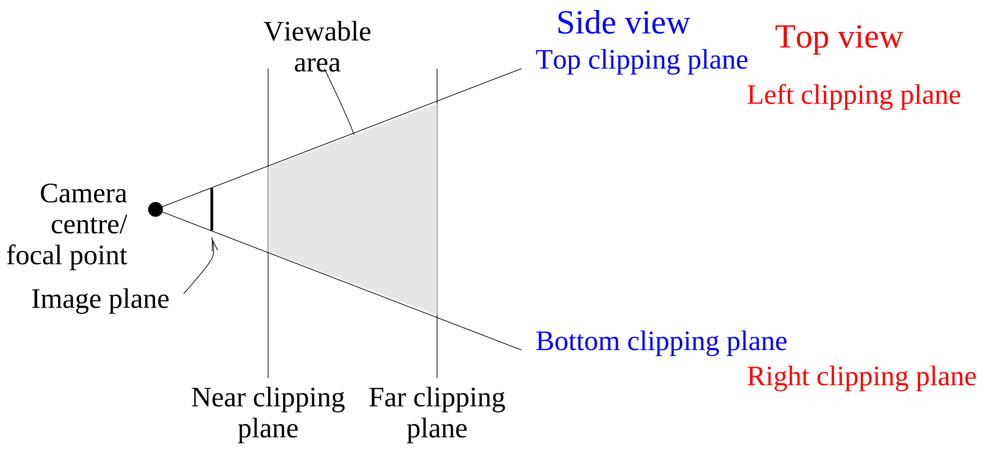
Question 3: Clipping (29)

(a) Why is the image forming process from the world to an image known as a projective transformation? (1)

It "projects" 3-D to 2-D.

(b) This projective transformation is expensive to compute, and so only the viewable world should be projected. What part of the world is visible in an image? Illustrate with a diagram. You should include 6 clipping planes in the diagram. (4)

The viewable world is what can be seen from the camera centre through the rectangle if the image in the image plane.



(c) What is the name of this viewable region of space? (1)

The Frustum

(d) There are two approaches to ensuring that only the viewable regions of space are projected. Briefly describe what they are. (2)

1. **During scan conversion check pixel by pixel, or the ends of lines.**
2. **Analytically clip the lines before scan conversion.**

(e) What are the three possible conditions for the two end points of a line when it is being clipped? What are the implications of each of these conditions? (6)

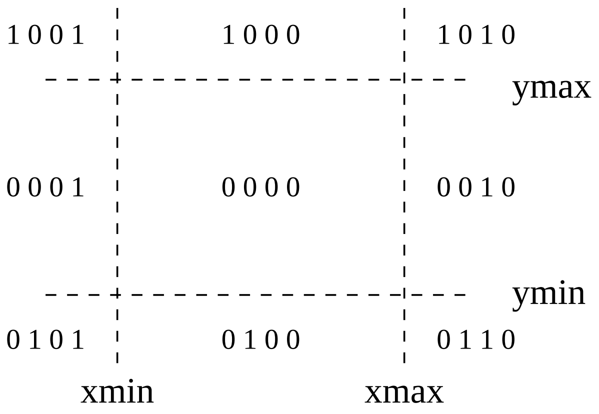
1. **Both inside frustum - trivially accept.**
2. **One inside, one outside - take inside one and intersection of line with first clipping plane crossed.**
3. **Both outside - part of the line may be visible - further analysis is required.**

(f) The Cohen-Sutherland algorithm uses the concept of *out-codes* for each end of the line. What are out-codes? (1)

Outcodes indicate whether a point is on the viewable (0) or unviewable (1) side of a clipping plane. Each vertex has an outcode for each clipping plane.

(g) Define the out-codes for a typical rectangular clipping area and show all the possible values. (6)

Outcode	
1 (MSB)	$y > ymax$
2	$y < ymin$
3	$x > xmax$
4 (LSB)	$x < xmin$



(h) Explain how out-codes are used to calculate the visible parts of lines. Describe in detail how the algorithm determines the visible section of a line that can not be trivially accepted or rejected. (8)

1. **If both outcodes are all zero - trivially accept the line.**
2. **If the outcodes bits for the same plane are both 1 (the bitwise AND of the outcodes is non-zero) - trivially reject.**
3. **Otherwise:**
   (a) **for a non-zero outcode (an outside point) select a non-zero bit (a plane that the point is outside),**
   (b) **intersect the line with that plane - replace the point under consideration with the intersection**
   (c) **recalculate the outcode for the point - ie set that bit of the outcode to zero**
   (d) **repeat from (1) until trivial accept/reject**