Computer Modelling 2

Practical 4b

**Linear interpolation:**

***Linear interpolation*** is the process where we fill in the gap between two given points in an outline by a straight line called a ***linear interpolant***. Uses for this technique include (1) curve drawing and (2) animations – especially when we want to animate multiple properties at the same time. The animation as described in Question 1 below demonstrates an example of (2).

**Parameters:**

In maths we often use a parameter (usually called t) in conjunction with the relevant equation(s) to describe lines and curves of various types.

For example, the following equation fully describes a line where t varies from 0 to 1, and P0 and P1 are two endpoints of the line.

P = (1 – t)P0 + tP1

This equation will compute the points in between P0 and P1 because as we increase the parameter t, it moves from one endpoint to the other.

In the above equation we can think of (1-t) and t as weights, and P as a weighted average of two points.

Note that if t = 0 then P = P0, and if t = 1, then P = P1. If t is between 0 and 1, then P can be located anywhere along the line.

This technique is not just relevant to drawing lines and curves…

**For example:**

Start with a range and a value between 0 and 1 (our parameter value). A linear interpolation returns a value in that range that the given parameter value points to.

For example, if the given range is the number of days in the year, which day corresponds to 0.75 along the line?

This is very straightforward - we just multiply 0.75 by the max (of the range) to get 273.75. This means that we are 75% through the year on the 274th day.

365

0

0.75

However, what if the range does not begin at 0?

For example, what are the calculations if we wanted to draw an object on the screen within the range [100, 450] pixels on the x axis if the parameter value 0.37 represents the position where we want to draw.

0.37

450

100

Now the size of the range is 350. If we multiply 350 by 0.37 we get 129.5. This value represents an offset from the min value, so we add the min value to 129.5 to get 229.5.

The formula is: (max – min)\*parametervalue + min

**Question :**

Write the JavaScript code to generate the following animation:

A circle moves diagonally from the top left corner to the bottom right corner of the canvas. As it moves across the canvas:

* The radius gradually increases.
* It gradually becomes more transparent until it disappears (when it reaches the bottom right corner).

**Notes:**

* To draw a circle, see ‘lect5’.
* This script should include a function which returns a linear interpolation, given a range and parameter value as defined above. Call the function linterp.
* Note that there are 3 changing properties in this animation:

1. The position of the circle.
2. The size of the circle.
3. The transparency of the circle.

* To bring about each of these changes, we call the function linterp on each of these changes. Each of these function calls has 3 variables associated with them.
* Note that in the examples above, t was given a fixed value. In animation, t is constantly changing. Note that the range is [0, 1] – i.e. t starts at 0, and cannot go beyond 1.
* Let the dimensions of the canvas equal the dimensions of the window for maximum effect (not including toolbars or scrollbars).

**Programming steps:**

Set min and max values for position, transparency and size (hardcode values).

Initialise t

Define function linterp

**In animation loop:**

Set opacity so that it changes from opaque to transparent.

Draw circle with properties changing.

Deal with t (has to be in the range [0, 1]).