2. Theoretical Tasks: Procedural Content Generation (5 Points)

1.1 Voronoi Diagram

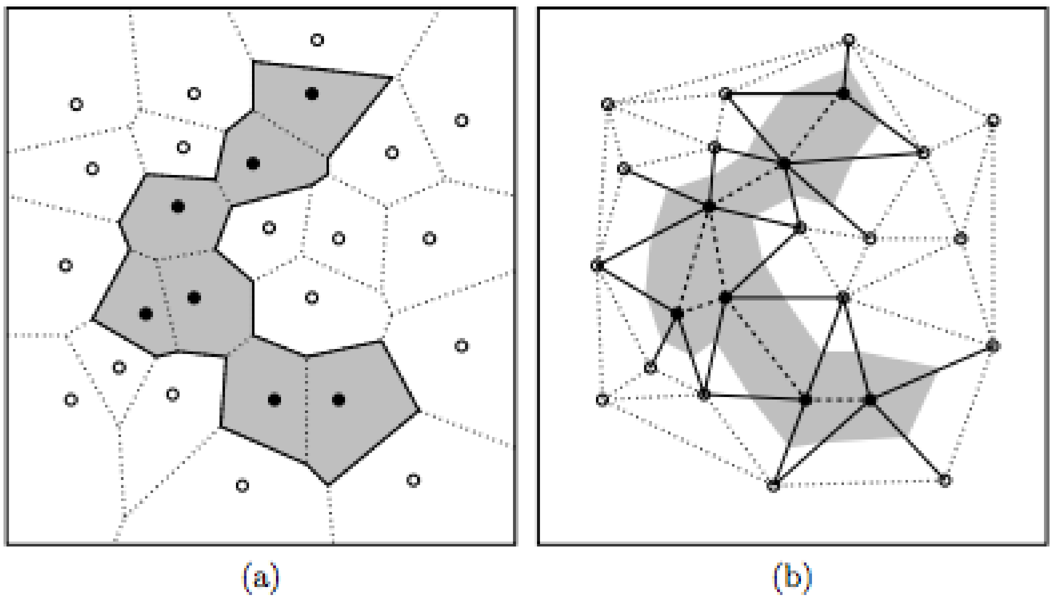
In a real-time strategy game, players can gain control of cities. The cities are specified by their x and y-positions in a plane. For this game, you want to implement a visualization that shows the area of influence of each player by displaying the border of the region they control. The input to this visualization is the set S of all cities and the subset P of cities that the player controls.

Describe how the Voronoi Diagram can be used for this task.

We are trying to find a „non-convex hull“ of the points of the cities. A regular convex hull would not approximate our mental image of a „border“ between two regions. As it turns out, since the "non-convex hull" is not uniquely defined, it is an open problem in research to define one.

When we build a Voronoi diagram of the map, we can use the cells that contain the cities and visualize those areas, for example by painting them into a texture and overlaying this texture over the map. To make the result more smooth, we can run a blur filter over this texture. One open question then would be how close two cities would have to be to each other so they are visualized with a connected region. If you are interested, you can have a look at the following paper for more information:

„What Is the Region Occupied by a Set of Points?" by Antony Galton and Matt Duckham (http://www.geosensor.net/papers/galton06.GISCIENCE.pdf)



1.2 Filter kernel

In the lecture, we presented the following kernel as carrying out an edge detection operation:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

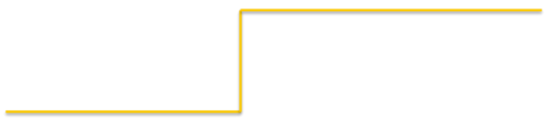
Explain in your words why this kernel detects edges. Edges are here meant to be those areas in the image where two areas with very different colors connect.

The answer is easier to see if we restrict ourselves to one row or column, reducing the problem to 1D.

An edge is a sudden change in the function that describes our image. The 1D-kernel would be:

$$\begin{bmatrix} -1 & 2 & -1 \end{bmatrix}$$

An edge in a 1D function would look like this as a diagram:



If all three pixels are constant (c):
p1 = p2 = p3 = c
Result = -1c + 2c – 1c
= 2c – 2c = 0
If the left pixel is different
p1 = c1, p2 = p3 = c2
Result = -1c1 + 2c2 –c2
= c2 – c1
→ Constant areas of the image become black, the edges remain

1.3 Gaussian blur

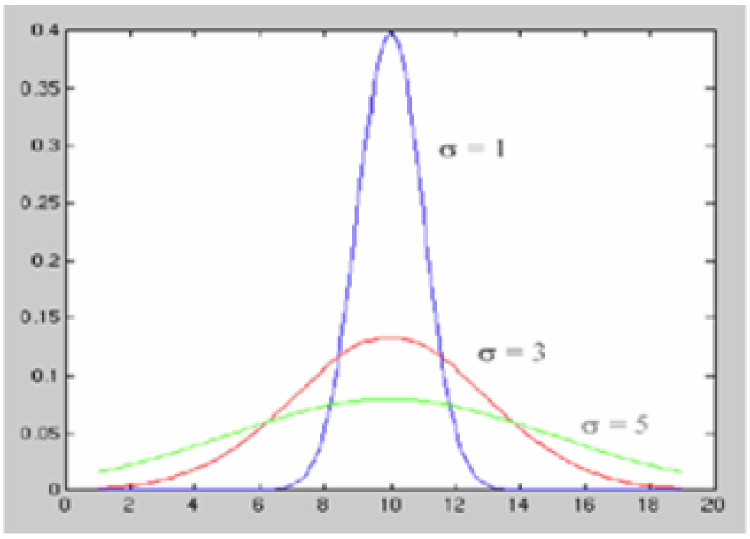
In one dimension, the Gaussian distribution can be calculated by using the formula shown here:

$$G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}$$

Draw the Gaussian for sigma = 1, 3 and 5, for example using a spreadsheet application or an online calculator such as fooplot.com.

a) Describe how the shape of the Gaussian changes.

b) If we use only the values of G(x) in the kernel (e.g. using size 7) and do not normalize them, our filter will change the brightness of the image. This effect is more noticeable the larger we choose sigma. Explain why this is the case, using your calculations from task a)



a) With larger sigma, the curve doesn't reach as high and becomes more stretched out

b) The area under the curve stays the same

→ The curve becomes wider and is therefore visibly cut off and if we use the coefficients without normalizing, they will not add up to 1

→ Without normalizing, we will change the brightness of the image