Summer Internship Project
Report

[TODO] Topic

Submitted by

Yannis Sauzeau
yannis.sauzeau@etu.univ-poitiers.fr
University of Poitiers

Under the guidance of

Jiří Hladůvka
jiri@prip.tuwien.ac.at
Walter G. Kropatsch
krw@prip.tuwien.ac.at
Thierry Urruty
thierry.urruty@univ-poitiers.fr

TU Wien
Faculty of Informatics
Institute of Visual Computing and Human-Centered Technology
Pattern Recognition and Image Processing Group
Favoritenstr. 9/5. Stock/Stiege 2/E193-03
A-1040 Vienna
Austria

Abstract

Contents

Chapter 1

Objective

This internship had several personal objectives:

- Discover the world of research in the field of computer vision in a scientific way.
- Put into practice my knowledges acquired during my bachelor and my first year of master.
- Develop my english skills in order to be able to communicate with the world of research.

The technical objective of the internship was to contribute to an interdisciplinary research project bridging plant biology, 3D imaging, and computer science. My main contribution was to modelize the propagation of CO_2 in leaves cells using a diffusion equation.

Chapter 2

Introduction

2.1 PRIP Laboratory

2.1.1 Presentation

The Pattern Recognition and Image Processing Laboratory (PRIP) is a research group of the Vienna University of Technology (TU Wien) and is part of the Computer Science department. The TU Wien is Austria's largest research and educational institution in the field of technology and natural sciences. The Computer Science department is one of Europe's leading research and innovation institutions. PRIP Laboratory is focused on advanced image representations and methods that allow the structure of the image to become an essential part of recognition systems. Among the people working in the PRIP Laboratory are:

- Prof. Walter G. Kropatsch, Head of the Group.
- Dr. Jiří Hladůvka, My tutor.
- Majid Banaeyan, PhD student.
- Darshan Batavia, PhD student.

2.1.2 *Water's gateway to heaven* project

Water's gateway to heaven is an interdisciplinary research project funded through the Life Sciences programme on Multimodal Imaging of the Vienna Science and Technology Fund (WWTF) [4]. The project is a collaboration between three viennese universities:

- Plant ecophysiologists and anatomists at the University of Natural Resources and Life Sciences [6].
- Plant cell biologists at the University of Vienna [7].
- Computer scientists expert in pattern recognition and image analysis at the Vienna University of Technology (TU Wien) [8].

The project focusses on the stomata, tiny pores on the surface of plant leaves. Stomata open and close to provide CO_2 for photosynthesis and to limit water loss. This project uses novel temporal $3D$ imaging to provide a better description of stomatal movements in order to get a mechanistic understanding of how transient this movement. The goal is to answer long-standing questions about stomatal movements and to generate basic knowledge on how to improve stomatal responses under dynamic environments in order to increase net productivity and water-use efficiency [10].

2.2 Technical overview

The images use for this project are $3D$ images of leaves coming from high-resolution X-ray micro-tomography (micro-CT) and fluorescence microscopy. The size of the images is $2000 \times 2000 \times 2000$ pixels. In order to track the change of the individual cells over time, hierarchies of abstract topological cell complexes are used, which will reduce the image data to a neighboring structure of the plant cells without losing the relation to the original data. This makes it possible to verify hypotheses at any time that arise during the course of the biological analysis but may not have been known at the time the hierarchy was built. Furthermore, this structure of the plant cells is to be used to simulate dynamic processes [9].

2.2.1 Combinatorial maps

Chapter 3

Work Done

3.1 First work

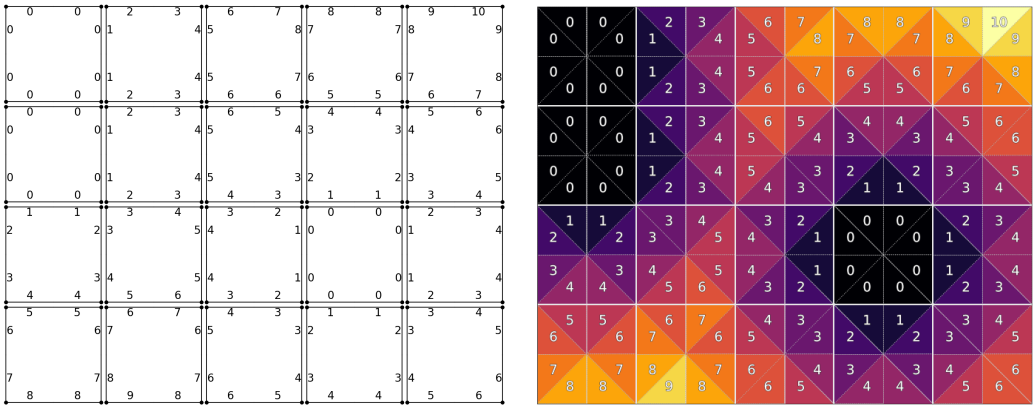
During the first month, my work was mainly focused on being familiar with the project and the data, in order to find the best way to contribute to the project.

3.1.1 Python library

Carmine Carratù, the previous ERASMUS student, has been working on algorithms for the computation of distance transform into $G-maps$. My goal was to understand the code and know how to use it to create figures for future publications. The code was not documented so in reading the code I started to comment it and create documentation. I also bring some optimizations to the code. Some functions was not very helpful for the purpose of the project and the code was in several files, whereas we wanted to keep the code as simple as possible in one notebook. So I created this notebook to combine all the usefful and documented functions in one place.

New representation on a $G-map$

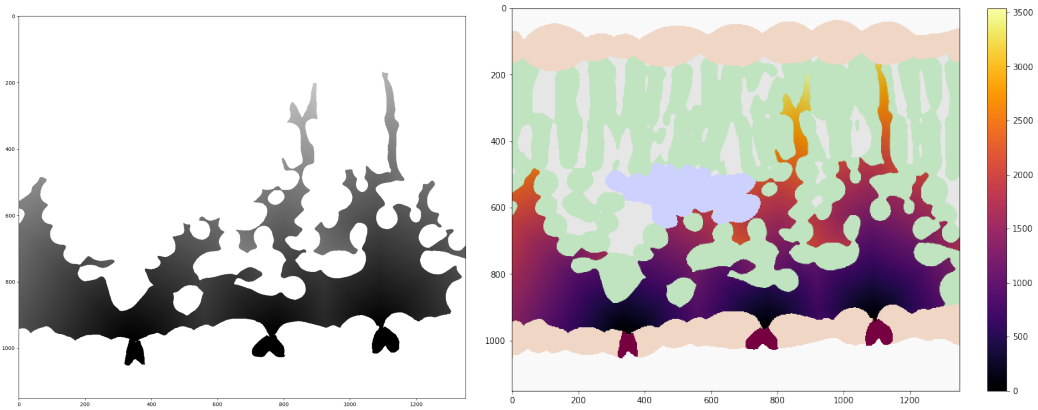
The representation of the distance transform was changed to be more intuitive. On the figure (3.1a) the distance transform is represented as a $G-map$ with the distance values on the dart. It's hard to see the variation of the distance values. The new representation is shown in the figure (3.1b). This representation uses triangles for each dart showing the distance values. In addition, we use the *inferno* color map that is perceptually uniform with monotonically increasing luminance [3].



(a) Old representation **(b)** New representation

Figure 3.1: Different representations of the distance transform on a $G - map$

New representation on a leaf image



(a) Old representation **(b)** New representation

Figure 3.2: Different representations of the distance transform on a leaf

On the leaf image, the representation was changed too. Before the distance transform was print with a black and grey color map as shown in the figure (3.2a). The new representation, shown in the figure (3.2b) uses the *inferno* color map and an overlay can be seen to show the other parts of the leaf image.

```
PS C:\Users\Yannis Sauzeau\Documents\combinatorial\combinatorial> python .\command_lines.py --help
Usage: command_lines.py [OPTIONS]

  Command line interface for plotting images.

Options:
  -p, --path FILE             Path to the image  [required]
  -o, --output FILE           Output path  [default: output.gif; required]
  -s, --seeds INTEGER         Seed label  [required]
  -v, --verbose               Verbose mode
  -f, --frame INTEGER         Frame number
  -i, --interval INTEGER      Interval between frames (ms)  [default: 100]
  -t, --triangle              Draw triangle on each label
  -pg, --propagation INTEGER  Propagation label
  -tg, --target INTEGER       Target label
  -c, --colormap TEXT         Colormap  [default: inferno]
  -ol, --overlay              Overlay mode
  -l, --legend                Display legend
  -a, --axis                  Display axis
  --title TEXT                Display title
  --help                      Show this message and exit.
```

Figure 3.3: Command line program for wave propagation animation

Wave propagation animation

The distance transform algorithm uses a wave propagation method to propagate the distance values. I used *Click* [1] Python package to create a command line program that can be used to animate the propagation of the distance transform. On the figure (3.3) the different options for the command line program are shown. Among this options, we have the required integer for the seed label, it's form the pixel with this label that the wave will start propagating. The propagation label integer corresponds to the pixel with this label that the wave will propagate to, and the target label integer is the same but for the target pixel.

3.1.2 Diffusion equation

In order to achieve the main goal of the Watergate project it's important to study the gas exchange process that occurs in plant leaves. The distance transform is used to compute the geodesic distance from stomata to mesophyll cells where the photosynthesis is performed. The gas exchange rate depends on this distance over which the diffusion occurs [5]. To study gas exchanges inside leaves, we have chosen the diffusion equation, described in 2D by this Partial Differential Equation (PDE):

$$\frac{\partial u}{\partial t} = \alpha \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) \tag{3.1}$$

where $u(x, y, t)$ is the concentration at position x, y in time t and α is the diffusion coefficient.

Iterative solution

With a finite-difference method, we can convert the PDE (3.1) into an explicit equation described as follows:

$$u(x,y,t+1) = u(x,y,t) + \alpha\big(u(x+1,y,t) + u(x-1,y,t) + u(x,y+1,t) + u(x,y-1,t) - 4u(x,y,t)\big) \quad (3.2)$$

where $\Gamma(x)$ is the set of neighbors of pixel x.

3.2 Modeling the Diffusion of CO_2 inside Leaves

3.2.1 1D Diffusion

The diffusion equation in 1D becomes

$$u(x,t+1) = u(x,t) + \alpha\big(u(x-1,t) + u(x+1,t) - 2u(x,t)\big) \quad (3.3)$$

where $t \geq 0$ is the iteration count and x the pixel index along the sequence. We assume that the total gas volumes (3.4) of both sequences does not change by the diffusion, e.g. for all iterations t.

$$\sum_x u(x,t) = \sum_x u(x,0) \quad (3.4)$$

Diffusion propagates in both directions (increase and decrease of index) symmetrically. Consequently we have for all iterations before reaching the end of the sequence:

$$u(x,t) + u(y,t) = L + H \quad \text{for all } x + y = 1 \quad (3.5)$$

Sequence initialization

Consider one 1D sequence of 2-connected pixels without self-intersection. Let's note x the index along the sequence and $u(x,t)$ the concentration at the time t. The first half ($x \leq 0$) should be filled with a high concentration of CO_2, H, and the second half ($x > 0$) with low concentration of CO_2, L:

x	...	-3	-2	-1	0	1	2	3	4	...
$u(x,0)$	...	H	H	H	H	L	L	L	L	...

Table 3.1: Sequence initialization

x	...	-3	-2	-1	0	1	2	3	4	...
0	...	H	H	H	H	L	L	L	L	...
1	...	H	H	H	L	H	L	L	L	...
2	...	H	H	L	$2H-L$	$2L-H$	H	L	L	...
3	...	H	L	$3H-2L$	$4L-3H$	$4H-3L$	$3L-2H$	H	L	...

Table 3.2: Result with $\alpha = 1$

Normalized formula

Setting $\alpha = 1$ we notice that $u(x,t)$ starts oscillating around the boundary between L and H already at the second iteration:

The calculated values, $2H - L > H$, $2L - H < L$, are outside the interval $[L, H]$. The reason is that the differences to the center pixels need to be normalized, or equivalently the range of $\alpha \in [0, 1/2]$ in 1D. The same result with $\alpha = 1/2$:

x	...	-3	-2	-1	0	1	2	3	4	...
0	...	H	H	H	H	L	L	L	L	...
1	...	H	H	H	$\frac{H+L}{2}$	$\frac{L+H}{2}$	L	L	L	...
2	...	H	H	$\frac{3H+L}{4}$	$\frac{3H+L}{4}$	$\frac{3L+H}{4}$	$\frac{3L+H}{4}$	L	L	...
3	...	H	$\frac{7H+L}{8}$	$\frac{7H+L}{8}$	$\frac{H+L}{2}$	$\frac{L+H}{2}$	$\frac{7L+H}{8}$	$\frac{7L+H}{8}$	L	...

Table 3.3: Result with $\alpha = 1/2$

Notice that for $\alpha \leq 1/2$ the monotonicity of the complete sequence is preserved. The diffused values are linear combinations of L and H and remain within $[L, H]$. The diffusion could be formulated with the neighborhood $\Gamma(x)$ for all indices x as follows:

$$u(x, t+1) = u(x,t) + \frac{\alpha}{|\Gamma(x)|} \sum_{n \in \Gamma(x)} (u(n,t) - u(x,t)) \tag{3.6}$$

With this formula, α needs to be normalized to the range $[0, 1]$. This formulation would also extend to higher dimensions.

Wavefront formula

We can derive direct formula (polynomials of degree t and parameters α, x, L, H) for any iteration t without the need to do the intermediate iterations within the first wave propagation, i.e. until the front reaches the end of the sequence. A simple formula for the front of the wave is:

$$u(t,t) = (1-\alpha^t)L + \alpha^t H \tag{3.7}$$

If t is the result of the distance transform, then the value of the wavefront can be computed directly without reference to intermediate results of previous iterations.

Polynomial Basis Function

More concretely, the weights of L and H for $t = 0 \ldots 2, x = -2 \ldots 3$ can be expressed as polynomials of α:

x	$u(x,t)$		
3	L	L	L
2	L	L	$(1-\alpha^2)L + \alpha^2 H$
1	L	$(1-\alpha)L + \alpha H$	$(1-2\alpha+3\alpha^2)L + (2\alpha-3\alpha^2)H$
0	H	$(1-\alpha)H + \alpha L$	$(1-2\alpha+3\alpha^2)H + (2\alpha-3\alpha^2)L$
-1	H	H	$(1-\alpha^2)H + \alpha^2 L$
-2	H	H	H
t	0	1	2

Table 3.4: Weights of L and H expressed with polynomials

We observe that the coefficients L and H are symmetric with respect to $x = 0.5$ and the polynomial of L is $1-$ the polynomial of of H for $x > 0$. For $x \leq 0$, it's the opposite, i.e. the polynomial of H is $1-$ the pronominal of L:

$$\begin{aligned} u(x,t) &= \sum_{k=0}^{t} c(x,t,k)\alpha^k L + (1 - \sum_{k=0}^{t} c(x,t,k)\alpha^k)H \\ &= H - \left(\sum_{k=0}^{t} c(x,t,k)\alpha^k\right)(H-L) \end{aligned} \tag{3.8}$$

where $c(x,t,k)$ is the coefficient of the polynomial of degree k at iteration t and x is the index along the sequence.

t	0	1		2			3				4				
x \ k	0	0	1	0	1	2	0	1	2	3	0	1	2	3	4
4	1	1	0	1	0	0	1	0	0	0	1	0	0	0	-1
3	1	1	0	1	0	0	1	0	0	-1	1	0	0	-4	7
2	1	1	0	1	0	-1	1	0	-3	5	1	0	-6	20	-21
1	1	1	-1	1	-2	3	1	-3	9	-10	1	-4	18	-40	35
0	0	0	1	0	2	-3	0	3	-9	10	0	4	-18	40	-35
-1	0	0	0	0	0	1	0	0	3	-5	0	0	6	-20	21
-2	0	0	0	0	0	0	0	0	0	1	0	0	0	4	-7
-3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 3.5: Coefficients $c(x, t, k)$ for the very first time steps ($t = 0 \dots 4$).

Deriving the coefficients of the polynomial (Table 3.5 shows the coefficients for the first 4 time steps), we arrived at the following closed-form involving binomial coefficients for negative arguments[1] [2].

$$c(x, t, k) = (-1)^{1+x+k} \binom{t}{k} \binom{2k-1}{x+k-1} \tag{3.9}$$

With respect of property (3.5), we have this property:

$$\sum_{k=1}^{t} c(x, t, k)\alpha^k = -\sum_{k=1}^{t} c(y, t, k)\alpha^k \quad \text{for all } x + y = 1 \tag{3.10}$$

With the property above, to know all the values of $u(x, t)$ of a certain iteration t, we only have to compute $\sum_{k=0}^{t} c(x, t, k)\alpha^k$ with $x \in [1, t]$.

Result

To study the diffusion of CO_2 in the leaf from stomata to the leaf cells, we first compute the distance transform $d(x)$ of each pixel $x \in R$ in the airspace R. Afterward, we compute the coefficients $c(t, k, x)$ with $t = \max_{x \in R} d(x)$. Once we have the coefficients, we can compute the concentration $u(x, t)$ by (3.9). The result is identical to the iterative solution and is visualized in figure (3.4).

Another point of interest is to compute only the diffusion values for the pixels corresponding to the leaf cells border. Indeed, to know the concentration of the leaf cells, we don't need to compute the diffusion values for the other parts of the leaf.

[1] Explaining the colored entries in Table 3.5

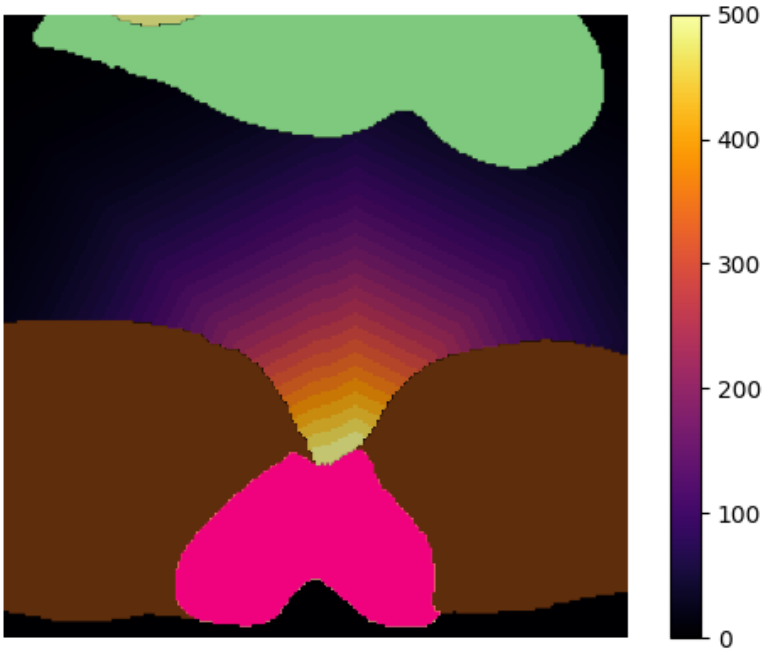


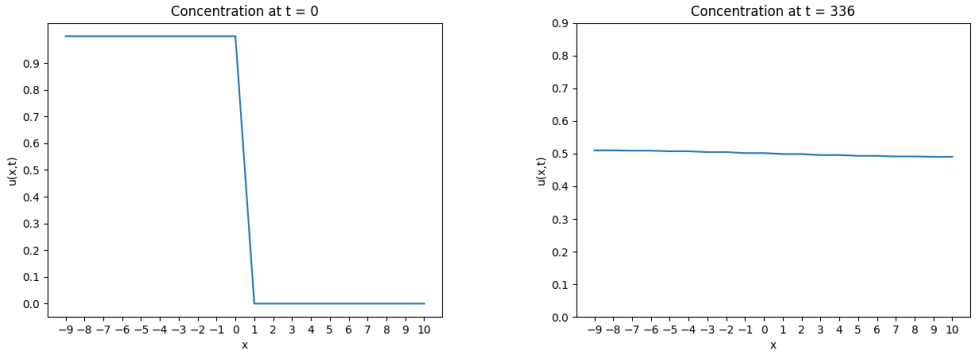
Figure 3.4: Equation (3.8) used to describe the diffusion of CO_2 from stoma.

3.2.2 Stomata's movement

In this section we will study the effects of the movement of the stomata. Let's start with a simple example. First, initialize a sequence like in Table 3.1. The size of the sequence is 20 pixels with $x \in [-9, 10]$, $H = 100\%$, $L = 0\%$ and $t = 0$ (see the figure (3.5a)). This sequence represents a closed stoma at the position $x = 0$, the pixels with negative index are the ones outside the leaf and the ones with positive index are the ones inside the leaf. When $t > 0$ the stoma is open. Using the iterative method (3.6) with $\alpha = 1$ we want to know the time t when the end of the sequence is near to be stabilized, i.e. that the value at the end of the sequence is greater than 49%. This value is reached at time $t = 336$ (see the figure (3.5b)).

Different sequence sizes

We want to know if the size of the sequence influences the time taken to reach the stabilized state. We use the same method as above but with different sizes of sequence. For 10 pixels, the stabilized state is reached at time $t = 83$, for 50 pixels at time $t = 2103$ and for 100 pixels at time $t = 8416$. With a size of 10 pixels the time taken to reach the stabilized state is much smaller than with a size of 50 pixels and with a size of 100 pixels. With the sequence



(a) $u(x, 0)$ **(b)** $u(x, 336)$

Figure 3.5: Concentration $u(x, t)$ in a sequence of size 20 at the initialization and near the stabilized state

of 100 pixels it takes 100 times longer to reach the stabilized state than with the sequence of 10 pixels.

Function to calculate the time taken to reach the stabilized state

The table below shows the time taken to reach the stabilized state for the first sequences from 2 to 10 pixels. The figure (3.6) shows this function for sequences between 2 and 100 pixels.

number of pixels in the sequence	2	3	4	5	6	7	8	9	10
time to reach the stabilized state	1	6	12	20	29	40	52	67	83

Table 3.6: Time to reach the stabilized state for first 8 sequences of sizes between 2 and 10 pixels

Let's see if sequence of time obtained from the table can be computed from the number of pixels. To know if the time to reach the stabilized state can be computed from the number of pixels we can see if it's a polynomial function. To that we can compute the difference table (3.7). We can observe that the differences didn't converge to a constant value. This is because the time to reach the stabilized state is not a polynomial function.

Stomata closing after few iterations

We want to know if the stabilized state is reached faster when the stomata close after few iterations. We start with a sequence of 100 pixels and we want to know the number of iterations needed to the end of the sequence reach the

1		6		12		20		29		36	
	5		6		8		9		7		
		1		2		1		-2			
			1		-1		-3				
				-2		-2					

Table 3.7: Difference table for the time to reach the stabilized state for the first 5 sequences of sizes between 2 and 6 pixels

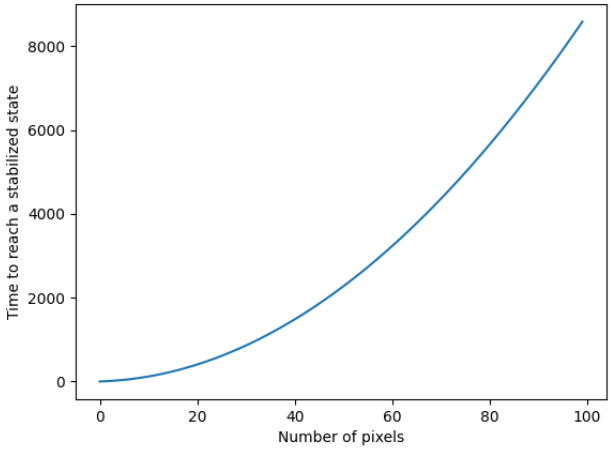


Figure 3.6: Function to reach a stabilized state from different sizes of sequence

value of 25% when the stoma is always open end when the stoma close after few iterations. When the stoma is open we need time $t = 1894$ to reach the value of 25%. On the figure (3.7), we can see that for reaching the value of 25% the stoma need to be close after time $t = 986$. When the stoma closes at time $t = 1000$, we need $t = 4000$ to reach the value of 25%, i.e. more than two times the time needed to reach the same value when the stoma is open. More the stoma takes time to close, less we need time to reach the value of 25%. When we close the stoma at time after $t = 1650$, then the time to reach the value of 25% is constant and equal to $t = 1894$.

3.2.3 Simulate the diffusion problem for higher dimensions

Let's transform our 1D sequence into a 2D sequence. We can do this by creating a matrix with the same size as the 1D sequence and filling it with the values of the 1D sequence. Let's note x the index of the pixel in the

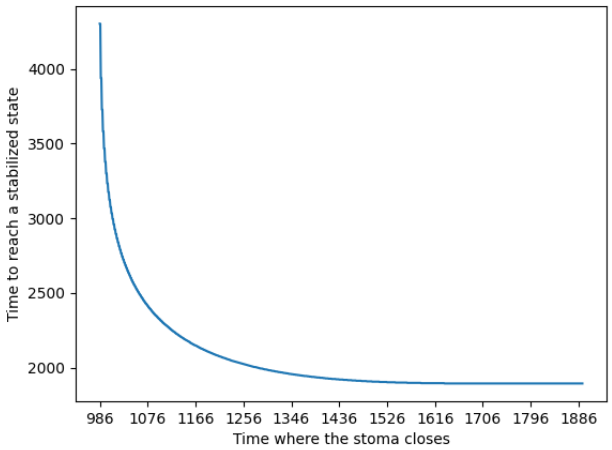


Figure 3.7: Function to reach a stabilized state when the stoma closes after x iterations

1D sequence and y the index of the pixel in the 2D sequence, and $u(x, y, t)$ the concentration at the iteration t. To differentiate the notation between the diffusion in the 1D and 2D case, we will use $u_{1D}(x, t)$ for the 1D diffusion and $u_{2D}(x, y, t)$ for the 2D diffusion. We will initialize the matrix with $u_{2D}(x, y, 0) = u_{1D}(x, 0)$ like the table below:

$u(x,0,0)$	...	H	H	H	H	L	L	L	L	...
$u(x,1,0)$	...	H	H	H	H	L	L	L	L	...
$u(x,2,0)$	...	H	H	H	H	L	L	L	L	...
...	...	...	...	...	...	...	...	...	...	...
$u(x,y,0)$	...	H	H	H	H	L	L	L	L	...
x	...	-3	-2	-1	0	1	2	3	4	...

Table 3.8: 2D sequence initialization

2D formula idealistic model

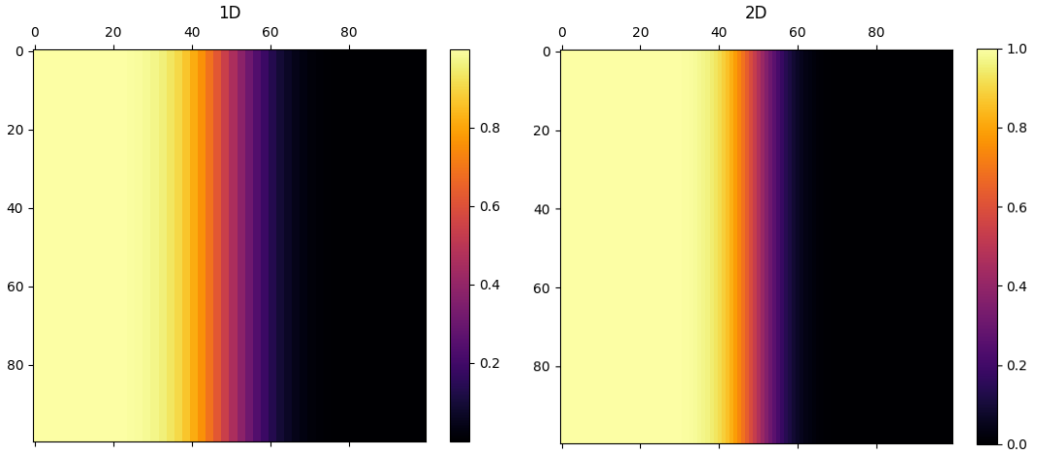
Let's adapt the formula (3.6) to the 2D diffusion problem:

$$u_{2D}(x, y, t+1) = u_{2D}(x, y, t) + \frac{\alpha}{|\Gamma(x, y)|} \sum_{n,m \in \Gamma(x,y)} (u_{2D}(n, m, t) - u_{2D}(x, y, t)) \tag{3.11}$$

With $\Gamma(x, y)$ the set of all the pixels that are connected to the pixel (x, y). Now we can compute the diffusion for the 2D sequence and compare the results with the 1D sequence. For the computation we will take $\alpha = 1$ and $t = 100$. The results (figure (3.8)) show that with the same alpha, the diffusion is two time faster in the 1D sequence than in the 2D sequence. It's understandable since the 1D diffusion only propagate in two directions and the 2D diffusion propagate in four directions. When we compare the results of the first iterations in the 1D and 2D sequences with the table (3.3) in 1D and table (3.9) in 2D, we can see similarity. For instance, $u_{2D}(1, y, 1) = u_{1D}(2, 2)$, $u_{2D}(1, y, 2) = u_{1D}(2, 4)$, or $u_{2D}(2, y, 2) = u_{1D}(4, 4)$. We can generalize by the formula:

$$u_{2D}(x, y, t) = u_{1D}(2x, 2t) \tag{3.12}$$

The formula below is working only with specific conditions, i.e. $u_{2D}(x, y, 0) = u_{1D}(x, 0)$ and $u_{2D}(x, y - 1, t) = u_{2D}(x, y, t) = u_{2D}(x, y + 1, t)$.



(a) Diffusion with $\alpha = 1$ in 1D sequence of size 100

(b) Diffusion with $\alpha = 1$ in 2D sequence of size 100×100

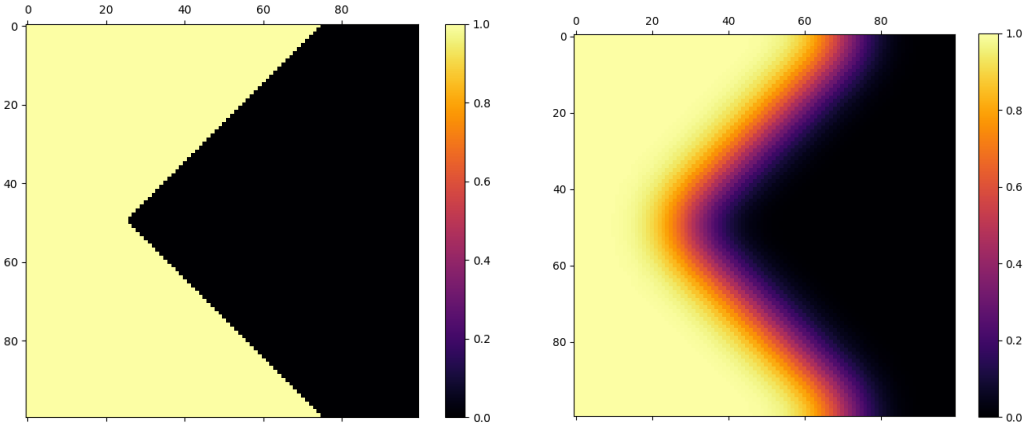
Figure 3.8: Diffusion with $\alpha = 1$ in 1D and 2D sequence of size 100×100

General 2D formula

The shape of stomata is like a funnel, so let's create a 2D sequence of size 100×100 that looks like a funnel. The figure (3.9a) shows the initial sequence. Here the formula (3.12) is not working, but we can find a formula that works for every pixel. Let's start some observations. On the first step for the 1D diffusion we can isolate two different configurations that makes changes in

x	$\dots$	-2	-1	0	1	2	3	$\dots$
$u(x,y,0)$	$\dots$	H	H	H	L	L	L	$\dots$
$u(x,y,1)$	$\dots$	H	H	$\frac{3H+L}{4}$	$\frac{3L+H}{4}$	L	L	$\dots$
$u(x,y,2)$	$\dots$	H	$\frac{15H+L}{16}$	$\frac{11H+5L}{16}$	$\frac{11L+5H}{16}$	$\frac{15L+H}{16}$	L	$\dots$

Table 3.9: Result with $\alpha = 1$ in 2D sequence



(a) 2D funnel sequence at the initialization

(b) Diffusion with $\alpha = 1$ in 2D funnel sequence at iteration 100

Figure 3.9: Diffusion in 2D sequence of size 100×100 more near the stomata shape

the sequence. Each of this configuration leads to four variations in the 2D sequence:

- First configuration when $u(x,0) = H$ (figure (3.10a)):

  In this configuration in 1D, the value at $t = 1$ is different of the value at $t = 0$, only if one of the neighbor is H and the other is L. If both neighbors are H the value at $t = 1$ is the same of the value at $t = 0$. Taking in consideration the case when $u(x,0) \neq u(x,1)$, we want to know the similarity between the 1D and 2D sequences. In the figure (3.10a) we can see that the 1D sequence can be represented as four 2D sequences whose two sequences has the same number of neighbors H and L. So we will defined the similarities between the 1D and 2D. Let's defined N_H and N_L as the number of neighbors H and L in the 2D sequence. Let's note the subtraction between N_H and N_L as $\Delta_{HL} = N_H - N_L$. Then the formula is:

$$u_{2D}(x, y, t) = u_{1D}(\Delta_{HL}x, |\Delta_{HL}|t) \tag{3.13}$$

- Second configuration when $u(x, 0) = L$ (figure (3.10b)):

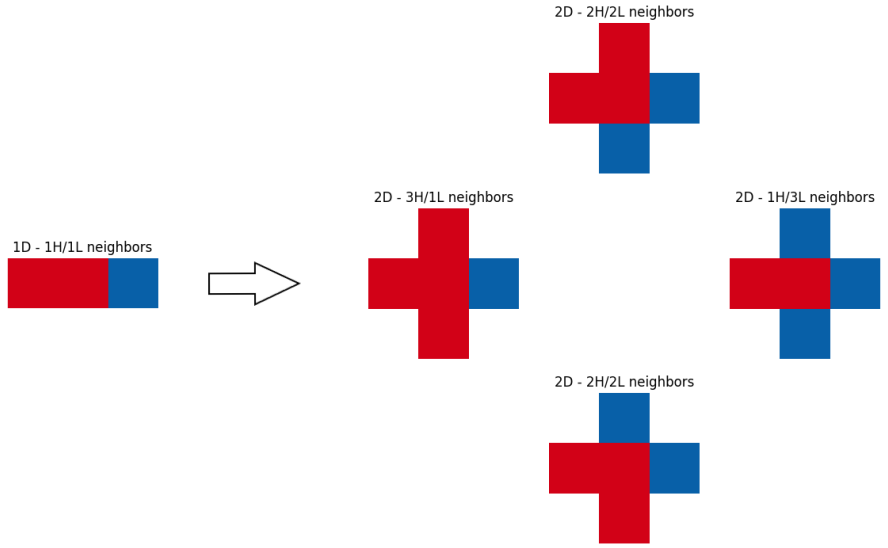
  The observations are the same as the first configuration, but we need to subtract N_L from N_H to get the correct value. Let's note this subtraction as $\Delta_{LH} = N_L - N_H$. Then the formula is:

$$u_{2D}(x, y, t) = u_{1D}(\Delta_{LH}x, |\Delta_{LH}|t) \tag{3.14}$$

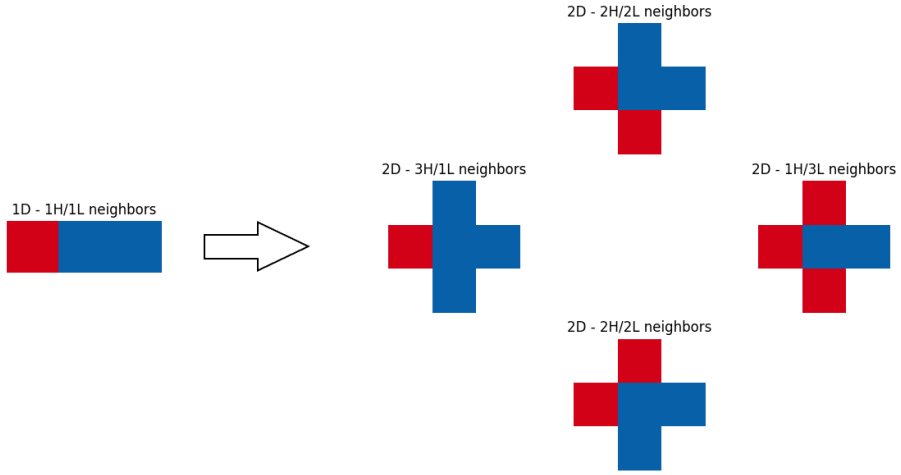
2D formula using the 1D formula

To summarize, we can find a formula to compute the value of concentration $u_{2D}(x, y, t)$ in the 2D sequence with the value of concentration $u_{1D}(x, t)$ in the 1D sequence. The formula can be written as:

$$u_{2D}(x, y, t) = u_{1D}(\Delta_N x, |\Delta_N|t) \text{ where } \Delta_N = \begin{cases} \Delta_{HL} & \text{if } u_{1D}(x, 0) = H \\ \Delta_{LH} & \text{if } u_{1D}(x, 0) = L \end{cases} \tag{3.15}$$



(a) HHL 1D sequence to 2D sequence



(b) HLL 1D sequence to 2D sequence

Figure 3.10: From initial 1D sequence to 2D sequence

Chapter 4

Future Work

4.1

4.1.1

Chapter 5

Conclusion

Acknowledgment

References

[1] Click Python Package website. https://click.palletsprojects.com/.

[2] M. J. Kronenburg. The binomial coefficient for negative arguments, 2011. doi:10.48550/ARXIV.1105.3689.

[3] K. Moreland. Why we use bad color maps and what you can do about it. *Electronic Imaging*, 2016:1–6, 02 2016. doi:10.2352/ISSN.2470-1173.2016.16.HVEI-133.

[4] Multimodal Imaging of the Vienna Science and Technology Fund website. https://www.wwtf.at/.

[5] Royal society of biology. Gas exchange. https://www.rsb.org.uk/images/12_Gas_exchange.pdf/.

[6] University of Natural Resources and Life Sciences website. https://boku.ac.at/.

[7] University of Vienna website. https://www.univie.ac.at/en/.

[8] Vienna University of Technology website. https://www.tuwien.at/.

[9] Water's gateway to heaven at PRIP website. https://www.prip.tuwien.ac.at/research/current_projects/wgh/.

[10] Water's gateway to heaven website. https://waters-gateway.boku.ac.at/.

List of Figures

List of Tables