Robust Shape Reconstruction through Optimal Transportation

Guillaume MATHERON

June-August 2015

Intership supervised by Pierre ALLIEZ and David COHEN-STEINER in the TITANE team at *Inria Sophia-Antipolis*

Abstract Our objective is to use an approximation of the W_1 distance (also referred to as "earth mover distance") as an error metric for shape reconstruction. We present different new approaches to compute this distance using wavelets and Kantorovich and Rubinstein's dual formulation. We also sketch an algorithm for shape reconstruction.

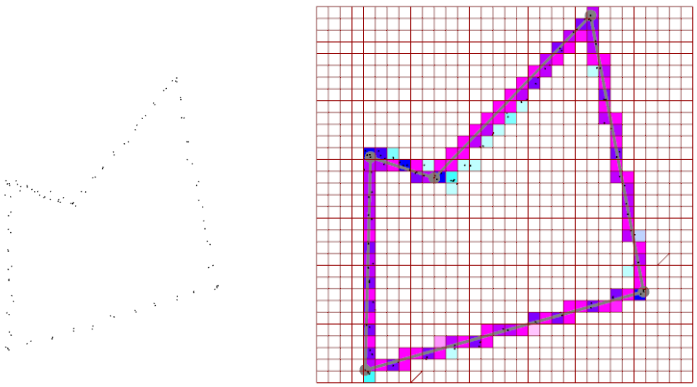


Figure 1: Point set approximated by polyline

Contents

1 Introduction **3**

2 Optimal transportation **3**

3 Dual formulation **4**
3.1 Previous approach 4
3.2 Proposal 4
3.3 2D case 5

4 Wavelet formulation **6**
4.1 How to approximate W_1 using wavelets 6
4.2 Proof 6
4.3 Experiments 8
4.4 Exhausting the optimization space 8
4.5 Adjusting normalization 9
4.6 Implementation and testing 11

5 Reconstruction algorithm **12**
5.1 Through vertex relocation 12
5.2 Reconstruction of 2D contours 12
5.3 Fine-to-coarse triangulation decimation 15
5.4 Conclusion 16

6 My internship experience **16**

7 Acknowledgments **17**

1 Introduction

Motivation Shape reconstruction is the process of converting a physical object into a digital model. This topic has received a considerable interest, and has already many applications in simulation, medicine, urban planning, art preservation and digital entertainment. Shape reconstruction is usually divided in two parts. First, a set of points is sampled on the surface of the physical object using various methods (such as depth cameras, stereo photography or laser scanning), then this set of points is converted to an appropriate surface representation, such as a triangulated surface. We are only focusing on the second step : converting a set of points into a surface. Ideally we wish to minimize the complexity distortion trade-off, but there is no universal consensus about the best way to define them.

Review The current popular method for surface reconstruction from a point set in industrial applications is Poisson reconstruction approach (see Figure 2). However it only handles smooth and closed shapes, and usually requires preprocessing steps to filter noise and outliers, which limits its applications.

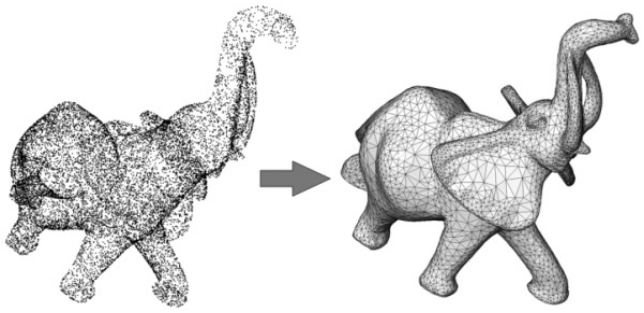


Figure 2: Surface reconstruction from point set using Poisson reconstruction [1]

The EM approach offers several advantages over other reconstruction methods, as described in [2].

- It preserves features such as sharp edges.
- It is very robust to outliers and noise.
- It allow a broad range of algorithms (both coarse-to-fine and fine-to-coarse).

Problem statement We take as input a point set sampled from a physical object. We interpret this set as a discrete measure, *i.e.*, a distribution of masses.
We wish to generate as output a simplicial complex which is the support of a continuous measure
The usual metric for computing a symmetric distance between two mass distributions is the Wasserstein distance.
We need a efficient way to compute the metric and a way to build the simplicial complex using mesh operators.

2 Optimal transportation

Wasserstein distance The Wasserstein distance between two probability measures λ and μ is :

$$W_p(\lambda,\mu) = \left(\inf_{\gamma\in\Gamma(\lambda,\mu)} \int_{M\times M} d(x,y)^p \mathrm{d}\gamma(x,y)\right)^{1/p} \tag{1}$$

where $\Gamma(\lambda,\mu)$ denotes the collection of all measures on $M\times M$ with marginals λ and μ on the first and second factors respectively.
In this internship, we only use W_1.

Transportation theory Assume a finite set M of items that we want to move ($M\subset\mathbb{R}^2$, and no two objects have the same position). If $m\in M$, $T(m)\in\mathbb{R}^2$ denotes the place where we want to move m,
then the overall cost of moving all objects to their new positions is $\sum_{m\in M}\|m-T(m)\|_p$.

Optimal transportation Assume another set $N\subset\mathbb{R}^2$, such that $|M|=|N|$ (where $|\cdot|$ denotes cardinality). We want to move all objects in M to N. We want to minimize the cost of moving all objects.
In other words, we want to find :

$$\inf_T \sum_{m\in M} \|m-T(m)\|_p$$

where T is a bijection from M to N. T is called the *optimal transport plan.*

This is called the *Min-weighted complete graph matching* problem.
In our case, each mass can be split and transported to all points of N, which makes the problem even more computationally intensive.

Computational aspect The usual way to solve for W_1 in a context where each element can have a different weight is to solve for the fraction of each element of M that goes to any point in N. Using constraints, this yields a linear system with $|M|^2$ variables. This approach was used in [2], but is very slow since it involves splitting each edge and each face into discrete bins, and solving a Linear Problem with many variables.

References For more background on optimal transportation, especially the continuous formulations, we refer to a reference book from Villani [3].
Several authors have investigated other ways to compute approximations of the optimal transport plan [4, 5].

3 Dual formulation

If λ and μ are two bounded distributions, the W_1 distance (also referred to as EM distance) can be obtained using this dual formula (also known as Kantorovich-Rubinstein formula) :

$$W_1(\lambda, \mu) = \sup_f \left(\left| \int f d\lambda - \int f d\mu \right|, f \text{ real cont.}, \text{Lipschitz}(f) \leq 1 \right) \tag{2}$$

3.1 Previous approach

A previous unpublished approach defined the following energy to approximate qualitatively the W_1 distance :

$$E = \frac{1}{N} \sum_{j=1}^{N} \left(\int f_j d\mu - \sum_{i=1}^{M} w_i \int f_j d\lambda_i \right)^2$$

Where f_j is a set of pre-defined functions and λ_i denotes the triangles in the reconstruction. Each triangle's density of measure is weighted by a coefficient w_i that we choose in order to minimize the energy.
The energy can be rewritten as :

$$E = \sum_{j=1}^{N} \left(b_j - \sum_{i=1}^{M} w_i a_{j,i} \right)^2$$

The minimum is reached when :

$$\forall k, 0 = \sum_{j=1}^{N} a_{j,k} \left(b_j - \sum_{i=1}^{M} w_i a_{j,i} \right)$$

Which can be found as the solution of linear system.
This approach has several degrees of freedom (that are open questions) :

- The choice of functions f_j (previous work focuses on finite support functions integrable in closed form on triangles, using [6]).
- The placement of function centers (uniform or data-dependent).

One of the initial objectives of my internship was to investigate the best way to define the functions, and I contributed an alternative approach.

3.2 Proposal

My approach is to compute explicitly the value of f in Kantorovich-Rubinstein formula on the centers of the cells of a uniform grid.
I rasterize the triangles and the samples on a uniform grid, then compute the explicit value of f on each point of the grid via linear programming (LP).
This yields an approximation of the EM distance between λ and μ

In 1D, we can immediately translate (5) into a LP program :

$$\begin{cases} \text{minimize} & f(1)(\lambda(1)-\mu(1)) + \cdots + f(N)(\lambda(N)-\mu(N)) \\ \text{with respect to} & f(1), \cdots f(N) \\ \text{under constraints} & |f(2)-f(1)| \leq 1 \\ & \vdots \\ & |f(N)-f(N-1)| \leq 1 \end{cases} \tag{3}$$

We can reformulate each constraint of the form $|f(i+1)-f(i)| \leq 1$ by $f(i+1)-f(i) \leq 1$ *and* $f(i)-f(i+1) \leq 1$ to obtain only linear constraints.
We can thus find f by solving a LP problem with N variables and $2N-2$ constraints.

3.3 2D case

Apart for the obvious problem of having N^2 variables instead of N, 2D brings its own set of problems :

1. There is no obvious way of defining a function on a lattice in 2D so that it admits a 1-Lipschitz continuation on $\mathbb{R}^2$.
2. The support of λ is a set of triangles. We need to rasterize them onto the grid.

We use a Manhattan distance on $\mathbb{R}^2$ by enforcing that f has a maximum slope of 1 along the edges of the lattice. The triangles are rasterized onto the grid using the following algorithm :

To know the value of λ at a specific cell center point :
– Build the square cell
– For each triangle that intersects this cell, compute the area of the intersection and add it to the value of λ

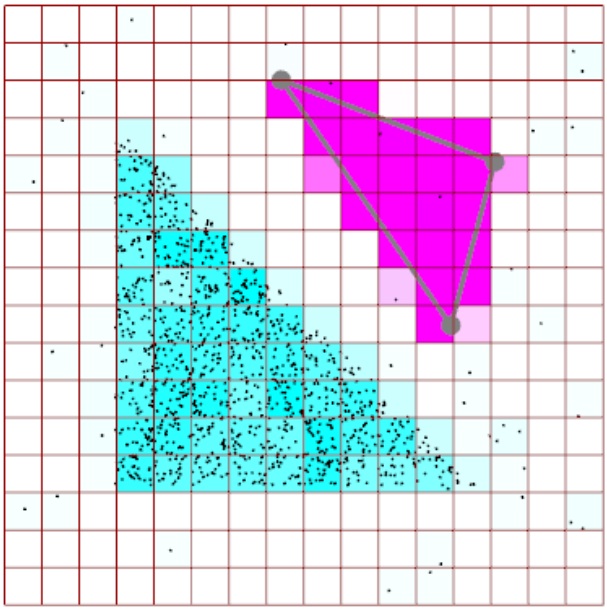


Figure 3: Rasterization of samples and triangles on the grid

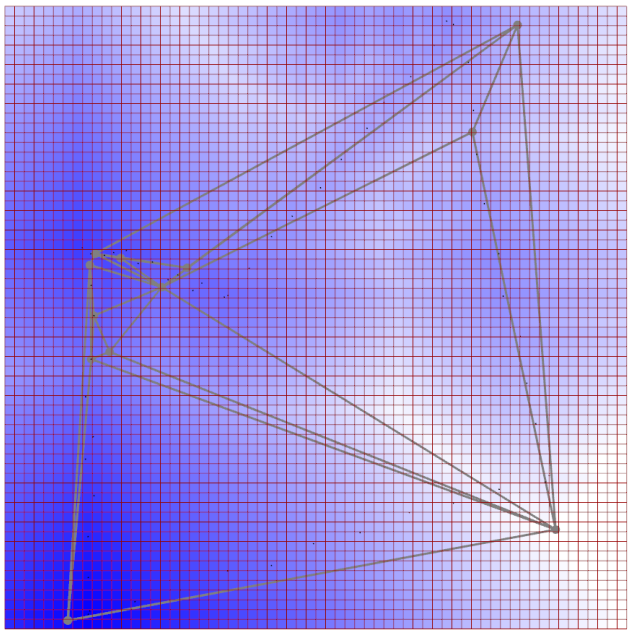


Figure 4: Function f computed by LP (edges of the triangulation are weighted with uniform linear mass)

In Figure 4, the triangulation is fitted to the samples, but extra edges are added. f is as high as possible (blue) where $\lambda > \mu$ and low (white) where $\mu < \lambda$, under the constraint that its slope does not exceed 1 in horizontal and vertical directions.

4 Wavelet formulation

Shirdhonkar and Jacobs [7] introduced an approach to approximate the Earth-Mover distance using wavelets, which is a multi-resolution approach. It takes a distribution of masses on a grid as an input (a matrix of real coefficients).

4.1 How to approximate W_1 using wavelets

We compute the coefficients of λ and μ on the grid and arrange them in 2D matrices (also called λ and μ) (as discussed in section 3.3), then compute their wavelet transform. If the histograms have a size of $n \times n$, this yields n^2 coefficients, arranged in a recursive matrix pattern. Shirdhonkar and Jacobs [7] provide the following approximation :

$$W_1 \approx \sum_i |W(\lambda)_i - W(\mu)_i| \, 2^{-2 \cdot j(i)}$$

where the sum is computed over all n^2 wavelet coefficients, $W(\lambda)_i$ and $W(\mu)_i$ are the coefficients of the wavelets transforms of λ and μ, and $j(i)$ is the *scale* of the wavelet i.
Note that since the wavelet transform is linear, we can either compute the wavelet transform of $\lambda - \mu$, or compute both transforms separately, then the difference. We will use the second approach since the distribution of samples is constant during the execution of the reconstruction algorithm, whereas the simplex changes (this way we have to parse the samples only once).
Note also that the wavelet transform of the histogram of a uniform triangle is sparse.

4.2 Proof

We now summarize of the proof contributed by [7].

Wavelet series representation A function f in $\mathbb{R}^n$ can be represented as a wavelet series [8] :

$$f(x) = \sum_k f_k \Phi(x - k) + \sum_\lambda f_\lambda \Psi_\lambda(x), \tag{4}$$

where Φ and Ψ are the scaling and wavelet functions. In our experiments we use the wavelet *Daubechies 4*, depicted in Figure 5.

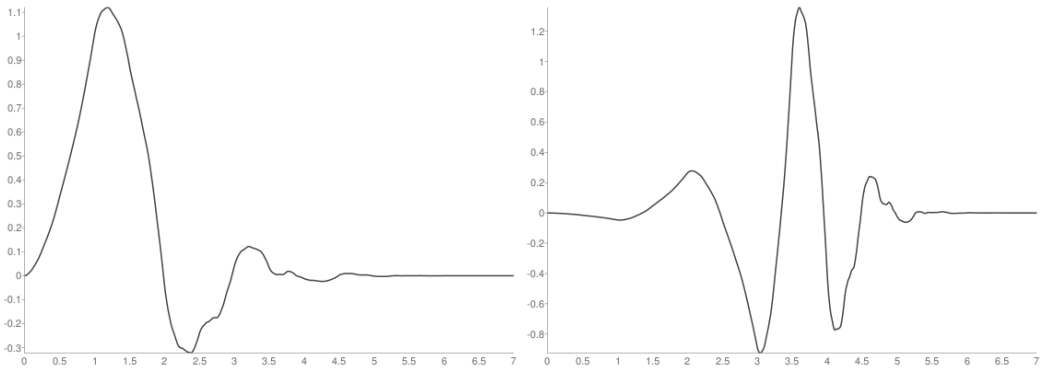


Figure 5: Daubechies 4 wavelet. Scaling function Φ (left) and wavelet function Ψ (right)

k runs in $\mathbb{Z}^n$ (where n is the dimension of f), and $\lambda = (\epsilon, j, k) \in \{1, 2, ..., 2^n - 1\} \times \mathbb{N} \times \mathbb{Z}^n = \Lambda$.
The set of all λ with a fixed j is noted Λ_j.
We define :

$$\Psi_{(\epsilon,j,k)}(x) := 2^{nj/2}\Psi^{\epsilon}(2^j x - k).$$

A wavelet Ψ is said to have regularity $r \in \mathbb{N}$ if it has derivatives up to order r and all of them (including Ψ itself) have *fast decay* (*i.e.* they decay faster than any reciprocal polynomial when $x \to \infty$).
For orthonormal wavelets, the coefficients can be computed as :

$$f_k(x) = \int f(x)\bar{\Phi}(x - k)dx$$

$$f_{(\epsilon,j,k)}(x) = \int f(x)\bar{\Psi}_{\lambda}(x)dx$$

Where $\bar{\Phi}$ and $\bar{\Psi}$ denote the complex conjugates of Φ and Ψ.
Let us remind that :

$$W_1(\lambda, \mu) = \sup\left(\left|\int f d\lambda - \int f d\mu\right|, f \text{ real cont.}, \text{Lipschitz}(f) \leq 1\right) \tag{5}$$

Let $0 < s < 1$.
Let $C_H(f) := \sup_{x \neq y} \frac{|f(x)-f(y)|}{\|x-y\|^s}$.
The constraint Lipschitz$(f) \leq 1$ can be stated as : $C_H(f) < 1$.
The following theorem [8] characterizes functions that satisfy this constraint :

Theorem 1. *A function $f \in L^1_{loc}(\mathbb{R}^n)$, (i.e. $|f|$ is integrable over all compact subsets of $\mathbb{R}^n$) belongs to Hölder class $C^s(\mathbb{R}^n)$ if and only if, in a wavelet decomposition of regularity $r \geq 1 > s$, the approximation coefficients f_k and detail cefficients f_λ satisfy : $|f_k| \leq C_0$, $k \in \mathbb{Z}^n$ and $|f_\lambda| \leq C_1 2^{-j(n/2+s)}$, $\lambda \in \Lambda_j$, $j \geq 0$ for some constants C_0 and C_1*

The authors derive the following lemma :

Theorem 2. *For $0 < s < 1$, if the wavelet series coefficients of the function f are bounded as in theorem 1, then $f \in C^s$ with $C_H(f) < C$ such that :*
$a_{12}(\Psi, s)C_1 \leq C \leq a_{12}(\Psi, s)C_0 + a_{22}(\Psi, s)C_1$
for some positive constants a_{12}, a_{21} and a_{22} that depend only on the wavelet and s. For discrete distributions, the same condition holds for $s = 1$ as well.

Main result :

Theorem 3. *Consider the Kantorovich-Rubinstein problem with the cost function $c(x, y) = \|x - y\|^s$, $s < 1$. Let p_k and p_λ be the wavelet transform coefficients (approximation and detail, resp.) of the difference density p generated by the orthonormal wavelet-scaling function pair Φ and Ψ with regularity $r \geq 1 > s$. Then for any constants C_0 and $C_1 > 0$, $\mu_c = C_0 \sum_k |p_k| + C_1 \sum_\lambda 2^{-j(s+n/2)}|p_\lambda|$ is an equivalent metric to the KR metric. For discrete distribution the same result holds with $s = 1$.*

4.3 Experiments

The authors recommend *Symlet 5* as a best wavelet, which is not provided by GSL (Gnu Scientific Library). We thus used *Daubechies 4*.
One caveat is that the GSL library only provides a few different wavelets ([7] specifies that the best wavelet is the *Symlet 5*, which is not provided by GSL, although the difference is very small.)
GSL only details the output format in 1D, but using the description from [7] solves the problem.
This algorithm is vastly faster than the LP approach (it takes 5 milliseconds to compute the EM distance on a 64×64 grid).
However, this approach revealed a weakness of our model : the EM distance was not very robust to outliers.

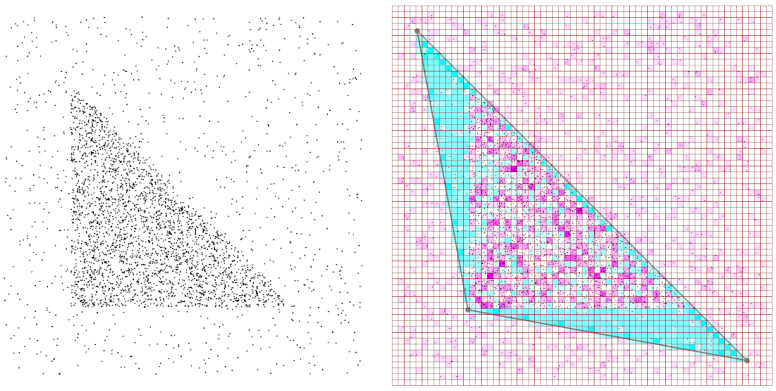


Figure 6: With many outliers, the global minimum (right) is incorrect

Indeed, the normalization is done do that the triangle has the same weight (*i.e.*, total measure) as all the samples, but if there are many outliers, when the triangle is perfectly fitted, the weight of the samples in the triangles is lower than the weight of the triangle. This induces a large transportation cost.

4.4 Exhausting the optimization space

4.4.1 Process

In order to explore the optimization space (especially to find local minimums), we performed the following experiment :
We automatically generate many possible triangles, and measure their optimal transport distance to a fixed set of points. Each of the six coordinates of the triangles are chosen in a set of 20 distinct values, which produces $20^6 = 64$ million distances.
The data is imported into a *MySQL* database.

4.4.2 Interpretation of results

This figure depicts the best W_1 distance achievable once the position of the first point of the reconstruction is fixed.

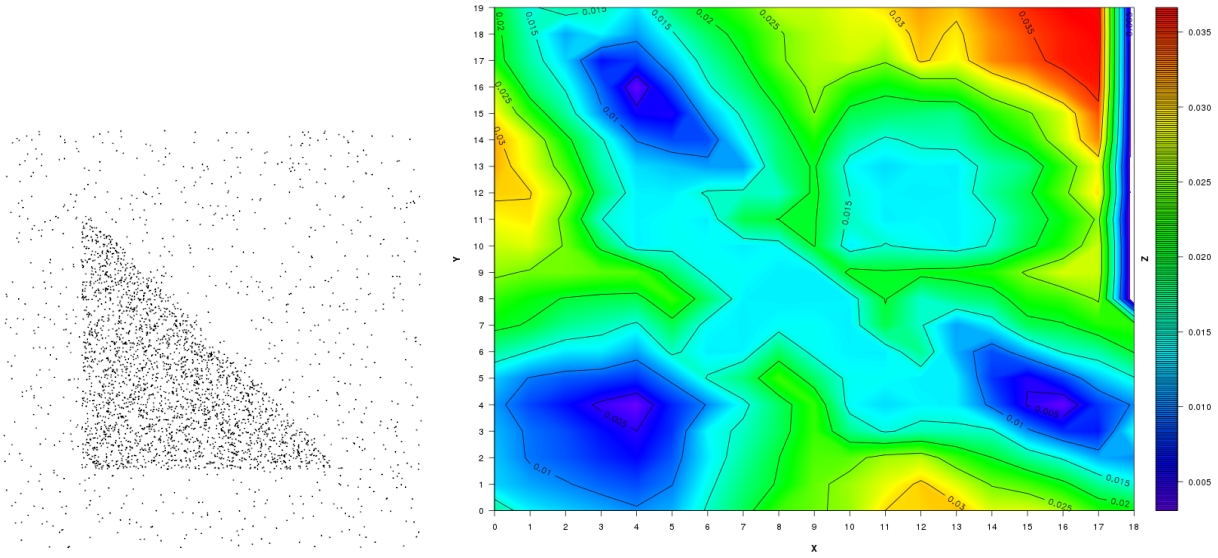


Figure 7: $color(x, y) := \inf_{x_2,y_2,x_3,y_3} W_1(samples, triangle(x, y), (x_2, y_2), (x_3, y_3))$. The artifact in the top-right hand corner is due to the fact the computation is stopped before the end

As shown in Figure 7 the three corners of the triangle corresponding to the global minimum in the energy landscape. Note that the local minimum for this projection (at the top-right), may not be a local minimum for the global optimization space.

4.5 Adjusting normalization

So far we made the assumption that all faces of the simplicial complex have the same uniform measure. This restricts the input to uniformly-sampled surfaces, which is less general than the piece-wise uniform case (in which each face can have a different weight). In [2], a linear system is solved to determine the weight of each element of the simplex.
We can mitigate the problem discussed in section 4, page 6 by using a single, adjustable weight for the whole reconstruction. This way only one scalar parameter k is added to our optimization problem. This means replacing the normalized histogram λ by $k\lambda$.
However, we would like to decouple it from the other parameters (which describe the shape of the simplex).
We wish to find k that minimizes W_1, when the shape of the simplex is fixed. There are several ways to do this.
Let $E = \{\text{wavelet } i \mid W(\lambda)_i \neq 0\}$.
Thanks to the linearity of the wavelet transform and the simple closed-form formula 4, some of the methods below can be adapted to allow a different weight for each triangle with little additional cost.

4.5.1 Dichotomy

Let $W(\lambda)$ denote the wavelet transform of the histogram λ.
We have

$$W_1 \approx \sum_i |W(k\lambda)_i - W(\mu)_i| \, 2^{-2 \cdot j(i)}.$$

The wavelet transform is linear, and we expect $\#E$ to be substantially smaller than N (see section 4.1, page 6), so

$$W_1 \approx C + \sum_{i \in E} |kW(\lambda)_i - W(\mu)_i| \, 2^{-2 \cdot j(i)},$$

where $C = \sum_{i \notin E} |W(\mu)_i| \, 2^{-2 \cdot j(i)}$ is a constant that does not depend on k, albeit it depends on λ (through E).
With C, $W(\lambda)$ and $W(\mu)$ precomputed, we can compute the estimate with complexity $O(\#E)$.
With a dichotomy process, we need to compute this estimate $-\log_2 \epsilon$ times to get $k_{optimal}$ within a tolerance of ϵ.
This method thus has a complexity of $O(\#E \log_2 \left(\frac{1}{\epsilon}\right))$

4.5.2 Convex optimization

We define the approximation we compute for a given k as $\Phi(k)$

$$\Phi(k) = C + \sum_{i \in E} |kW(\lambda)_i - W(\mu)_i| \, 2^{-2j(i)} \tag{6}$$

We define the set of functions $\Phi_i(k) = |kW(\lambda)_i - W(\mu)_i|$. All these functions are convex (see Figure 8)

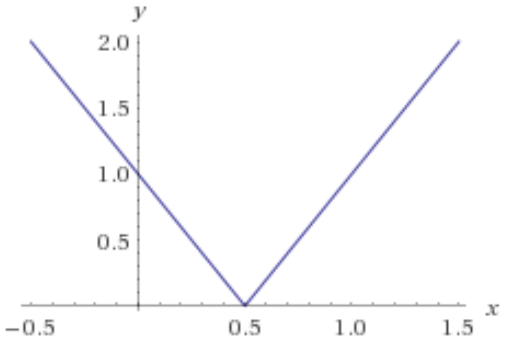


Figure 8: Curve of function Φ_i

Therefore, Φ is convex, as depicted by Figure 9

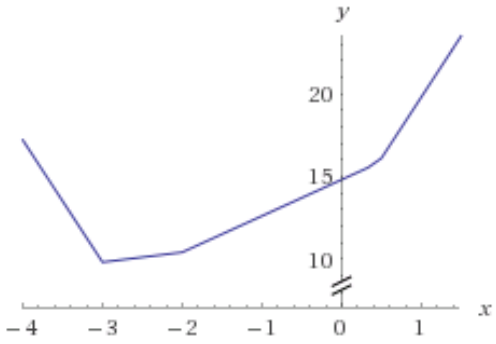


Figure 9: Curve of function Φ

In one dimension (we chose to optimize only one variable k), this method is not faster than a simple dichotomy. If we later decide to set a different weight for each triangle of the simplex, this method may become interesting.

4.5.3 Explicit enumeration

Another method is to enumerate all the singular points in the graph (see Figure 9) and find the best one.
The set of singular points is $k \in \left\{ \frac{W(\mu)_l}{W(\lambda)_l} \mid l \in E \right\}$.
Let $k_l = \frac{W(\mu)_l}{W(\lambda)_l}$.
The values at these points are $\Phi(k_l) = C + \sum_{i \in E} \left| \frac{W(\lambda)_i}{W(\lambda)_l} W(\mu)_l - W(\mu)_i \right|$.
It takes $O(\#E^2)$ to compute Φ at each singularity and find the best.
A more efficient algorithm starts by computing all the singular points, sorts them and then performs a dichotomy to find the minimum. This brings the complexity down to about $O(2\,|E|\,\log_2|E|)$.
Our experiments confirm that computation times are reduced. Most singular points k lie outside of the $[0, 1]$ range, so we can eliminate them early on from E to speed up computation.

4.5.4 Linear programming

Another look at Φ (6) reveals that it can be translated into a linear problem [1].

[1] `http://math.stackexchange.com/questions/432003/converting-absolute-value-program-into-linear-program`

$$\begin{aligned} \text{minimize} \quad & \textstyle\sum_i u_i 2^{-2j(i)} \\ \text{with respect to} \quad & k, u_1, u_2, \cdots, u_N \\ \text{subject to} \quad & \forall i, \begin{cases} kW(\lambda)_i - u_i & \leq W(\mu)_i \\ -kW(\lambda)_i - u_i & \leq -W(\mu)_i \end{cases} \end{aligned}$$

This linear problem has $N+1$ variables and $2N$ constraints, so it should have an approximate complexity of $O(N^3)$ (based on results with random matrices).
However, the constraint matrix is sparse for two reasons :

- In each constraint, only two variables (k and u_i) are involved.
- We expect most $W(\lambda)_i$ coefficients to be 0 (see section 4.1, page 6).

We can take advantage of this by examining what the constraint becomes when $W(\lambda)_i = 0$.

$$\begin{aligned} & \begin{cases} kW(\lambda)_i - u_i & \leq W(\mu)_i \\ -kW(\lambda)_i - u_i & \leq -W(\mu)_i \end{cases} \\ \Leftrightarrow \quad & -u_i \leq W(\mu)_i \leq u_i \\ \Leftrightarrow \quad & u_i \leq |W(\mu)_i| \end{aligned}$$

Since we want to minimize $\sum_i u_i 2^{-2j(i)}$, we can immediately set $u_i = |W(\mu)_i|$, and remove u_i as a variable.

We remind that $E = \{\text{wavelet } i \mid W(\lambda)_i \neq 0\}$.
The LP problem is re-written as :

$$\begin{aligned} \text{minimize} \quad & \textstyle\sum_{i \in E} u_i 2^{-2j(i)} \\ \text{with respect to} \quad & k, \{u_i | i \in E\} \\ \text{subject to} \quad & \forall i \in E, \begin{cases} kW(\lambda)_i - u_i & \leq W(\mu)_i \\ -kW(\lambda)_i - u_i & \leq -W(\mu)_i \end{cases} \end{aligned}$$

This linear problem has $1 + |E|$ variables and $2|E|$ constraints, we can thus solve it in $O(|E|^3)$.

4.5.5 Conclusion

Explicit enumeration is the best approach here in terms of speed. However, if in the future we want to give independent weights to each triangle (*i.e.*, have several k and a piecewise-constant measure), explicit enumeration will be exponential in the number of triangles, and the LP approach may become more viable.
Note that the article [7] does not specify what happens when we input the wavelet transforms of non-normalized distributions.

4.6 Implementation and testing

I implemented the explicit enumeration method using the same setup as in section 4.4, and did not see any noticeable slowdown. The first tests showed it solved perfectly the problem illustrated in Figure 6 (page 8).

The experiment showed that the expected triangle is indeed the global minimum, and that $|E|$ (the number of non-null wavelets) is about 25% of N^2.
After optimization (keeping the samples matrix μ in cache, etc...), I was able to compute the EM distance in about *30 ms* with a 512x512 grid.

Parallel programming There are two ways to speed up this algorithm using parallel processing :

1. Run the algorithm on several threads. Many of the operations that we have to perform to compute the EM distance can be computed in parallel.
2. Reconstruction requires computing a large number of EM distances against different polygons. This can be performed in parallel with each EM distance computation running on a single thread.

I first considered implementing parallel processing using CUDA on a GPGPU, but I did not have any GPGPU hardware available during my internship, so I used OpenMP to unfold the loops in the algorithm, but did not get any performance improvement, probably because of the big overhead of OpenMP, and the low number of cells in the 2D toolbox.
However, when dealing with many triangles, parallel programming greatly improves performance.

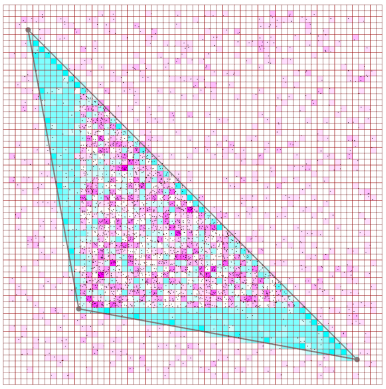
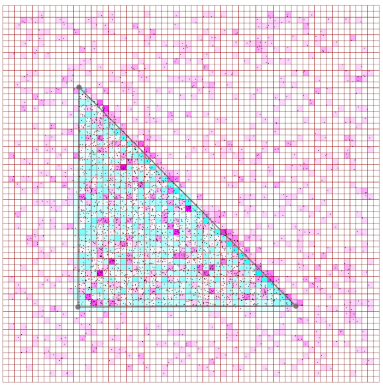
	Previous bad result	Expected best triangle
		
Value without k	0.0146 (best)	0.0231
Value with k (new method)	0.0267	0.0067 (best)

Figure 10: Global minimum found by the exhaustive exploration algorithm, performed using both metrics (normalized and adjusted)

5 Reconstruction algorithm

We wish to use the fast error metric discussed above to guide a reconstruction algorithm that is able to turn a samples set into a triangulation or set of edges.

5.1 Through vertex relocation

The simplest way to use the metric is to perform a vertex relocation.
Starting with a reconstruction that has the right topology and is not too complex, we move points of the reconstruction one at a time, improving each time the value of W_1.

Parameters : R and B (depends on the scale of the samples set)
1 – Select a random vertex from the current triangle
2 – Select a random direction $\vec{u}$ (unit vector)
3 – If $cnt \leq A$, set $\vec{d} = B\vec{u}$,
else set $\vec{d} = r\vec{u}$, with r selected randomly between 0 and R.
4 – Move the selected vertex by the displacement vector $\vec{d}$, and increment cnt.
5 – Re–compute the EM distance. If it is lower than before the relocation,
set $cnt := 0$ and move to step 4 using the same $\vec{d}$
Else move to step 1

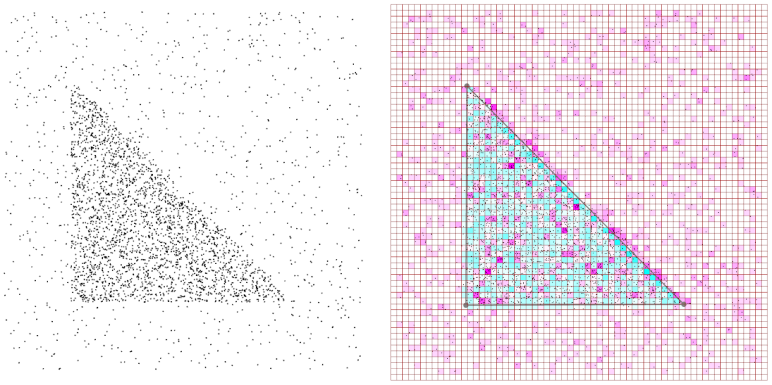


Figure 11: The relocation algorithm matches the samples even with outliers and noise, from any starting position.

5.2 Reconstruction of 2D contours

The 2D equivalent of 3D surface reconstruction is not 2D surface reconstruction but 2D contour reconstruction. We now discuss the reconstruction of 2D contours.

Instead of using samples consisting of a filled triangle, we investigate the reconstruction of a triangle outline, and instead of having a triangulation as the reconstruction mesh, we use a closed polyline.

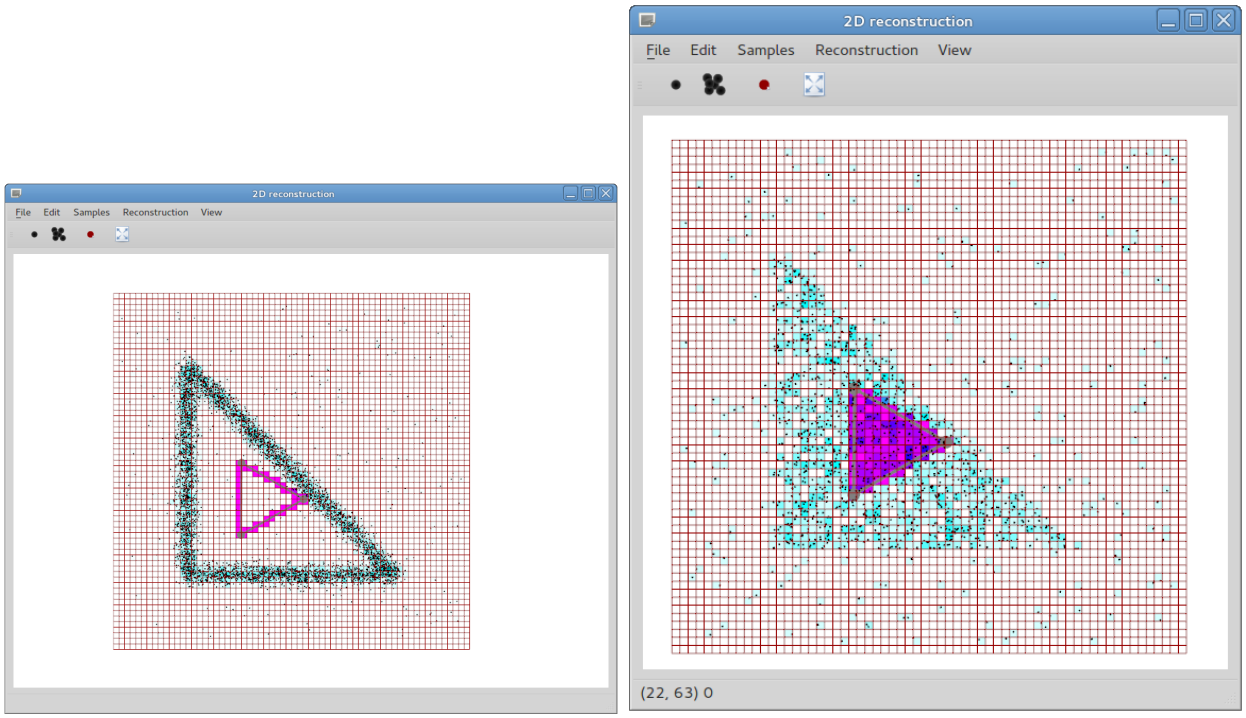


Figure 12: 2D contour reconstruction versus 2D surface reconstruction

5.2.1 Non-convex optimization space

A problem with this approach is the standard relocation algorithm (see section 5.1) gives a local minimum corresponding to a degenerate triangle on one edge of the samples set.

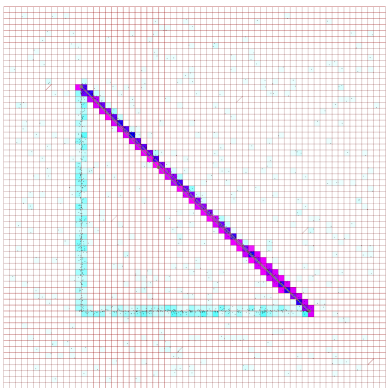


Figure 13: Degenerate reconstruction

The problem was obvious when plotting the value of w for different reconstructions, as illustrated below :

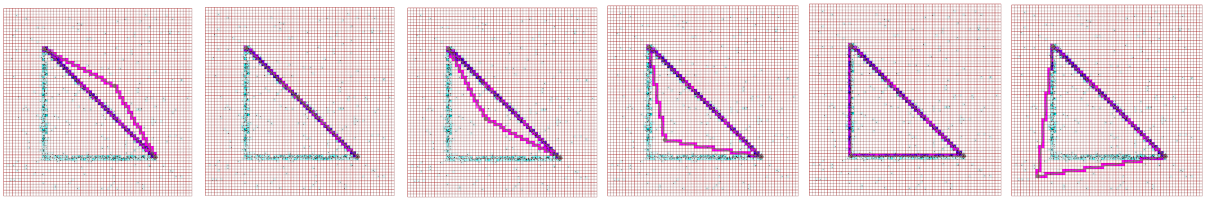


Figure 14: Reconstructions with different vertex locations to explore the optimization space (from left to right, $x = -0.3$, $x = 0.0$, $x = 0.4$, $x = 0.75$, $x = 1$, $x = 1.2$)

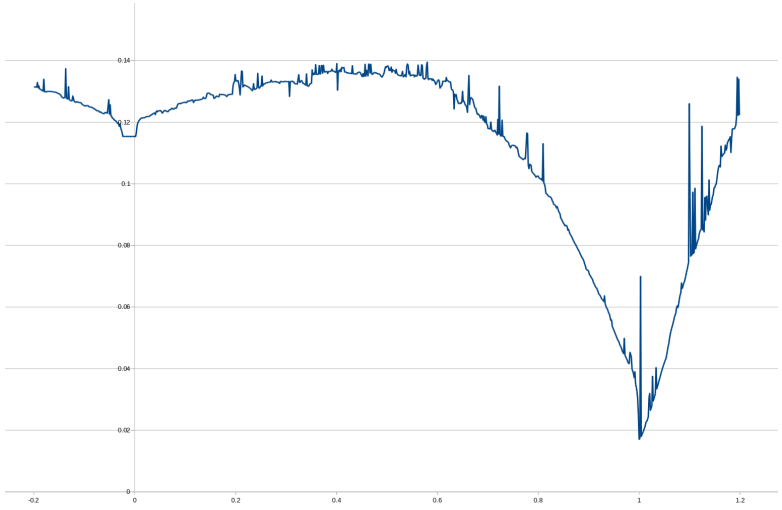


Figure 15: Projection of the optimization space. x-axis is the position of the third point of the triangle, as described in the caption of Figure 14

Notice that the curve is very noisy, which can be explained by floating-point approximation and discretisation due to rasterization on the grid. We clearly see the local minimum when $x = 0$, but the global minimum is at $x = 1$, as expected. The maximum is reached when $x = 0.5$.
This problem prohibits vertex relocation when the initial reconstruction is not close enough to the samples. Another approach could be to provide operators that can "jump" on the other side of the pass.
This curve also shows that it may be difficult to do gradient-descent algorithms efficiently, because of the noise. A possible fix could be to sample the gradient at several scales to do a weighted average and get an accurate prediction of the best relocation operation.

5.2.2 Exhausting the optimization space

We used the same process as described in section 4.4. Because no optimizations are performed, the computation is slower and is performed with only 10 values for each of the 6 dimensions of the optimization problem.

First projection This figure depicts the best W_1 distance achievable once the position of the first point of the reconstruction is fixed.

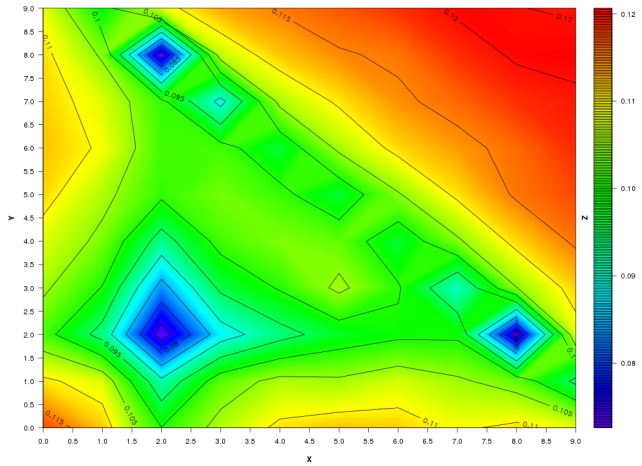


Figure 16: $color(x, y) := \inf_{x_2,y_2,x_3,y_3} W_1(samples, triangle(x, y), (x_2, y_2), (x_3, y_3))$.

We clearly see the three corners of the triangle that correspond to the global minimum.

Second projection I fixed the two lower points to their optimal position, and plotted the EM distance against the position of the third point

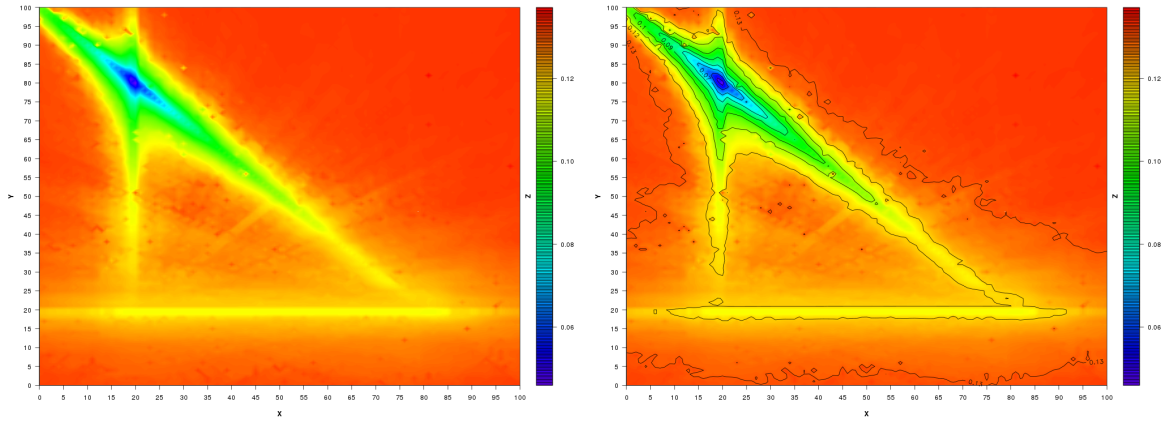


Figure 17: Value of W_1 for each possible position of the third point, with the two bottom points fixed

The figure above clearly shows the local minimum along the lower segment that is preventing our relocation algorithm from working properly.

Conclusion The experiments above show that reconstruction of a polygon though vertex relocation suffers from local minimums. Other methods discussed below yield better results.

5.3 Fine-to-coarse triangulation decimation

We used another algorithm, that creates a triangulation from all the samples (initially a Delaunay triangulation), and then greedily removes vertices using the metric described above to choose the best vertex at each round. Here is the result of removing vertices using the metric :

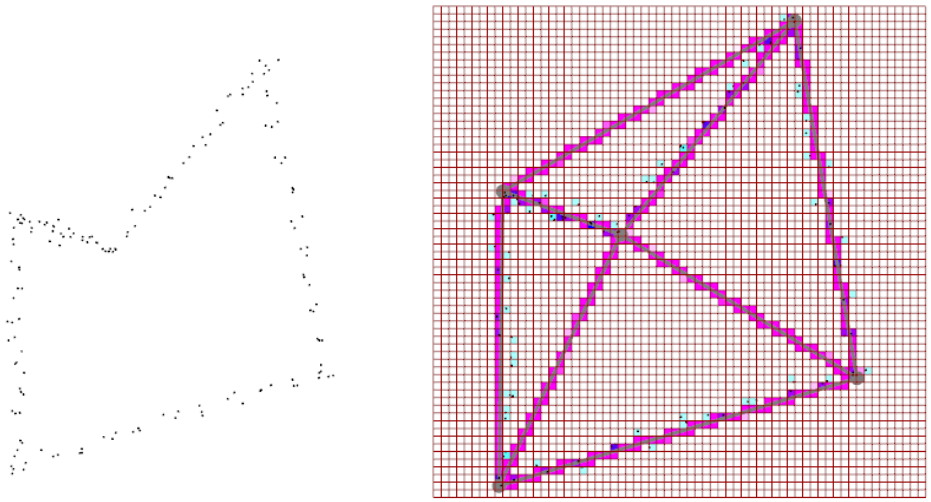


Figure 18: Samples and reconstruction using triangulation decimation. Note the inner and outer edges

We always have a convex triangulation, and inner edges are still present.
These edges can be subsequently removed greedily using the same metric, which yields the correct result :

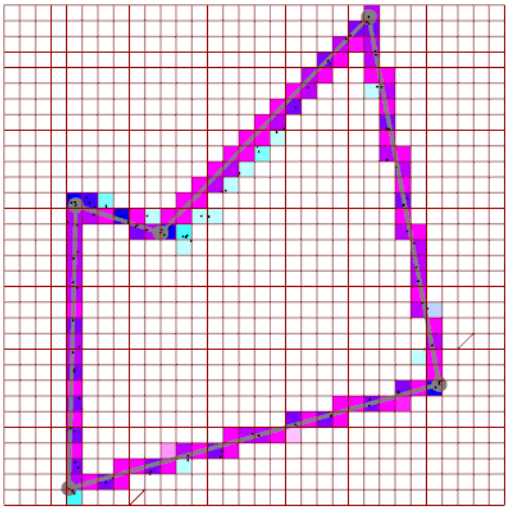


Figure 19: Inner and outer edges removed using the same algorithm

The result can be further improved using vertex relocation, now that we have the right topology and are close enough to the final result to avoid local minima.
However, the operator has to decide in advance how many edges and vertices should be kept. In addition, the presence of inner and outer edges confuses the decimation algorithm. Ideally the algorithm would be able to choose at each step any subset of the triangulation edges to hide, to avoid being greedy. However, this has an exponential complexity in the number of edges.
There is a polynomial way to achieve this using the linearity of the wavelet transform. However, this would require computing and storing the wavelet transform of each individual edge.

5.4 Conclusion

The use of wavelets provides a very fast metric for shape reconstruction.

Future work The reconstruction algorithm should have a better termination condition than simply a number of vertices. This could be achieved by defining a complexity-distortion trade-off, and minimizing it.
To improve performance further, it should be possible to re-evaluate W_1 locally using the linearity of the wavelet transform, and the fact that the wavelet transform of a small input is sparse.

6 My internship experience

Previous experiences Before this internship I already had a few experiences at *Inria Sophia-Antipolis* with Éric Madeleine and Thierry Vieville. My first internship was in 2009 and I mainly did web programming and observation. I then came back in 2011, and did web programming, Java development and project guidance for *Java's Cool*. My third experience was in 2012, and I started researching *Q-Learning* algorithms, and programmed a web-based educative toolbox.

This experience This fourth experience was a radical break from all previous internships : the background I acquired during *prépa* and *L3* allowed me to dive straight in a near-unexplored area of cutting-edge research. Since there were nearly no precedent works in this area, I was able to make critical decisions about the orientation of my research.

My contribution My supervisor and other colleagues gave me a very valuable hindsight of the domain, but I was still able to give good ideas and argue what direction seemed more promising. For instance, I chose to drop the idea of choosing a different weights for each face of the simplex, and to map the f function of Kantorovich-Rubinstein explicitly using LP.
I included my code in an existing toolbox developed by another intern.

7 Acknowledgments

I would like to thank Pierre Alliez for granting me this internship and guiding me along the way, David Cohen-Steiner for his valuable advice, his math skills and hindsight.
I would also like to thank Florence Barbara and Isabelle Delais for their help during and before the internship. Finally, thanks to all the TITANE team who welcomed me in their team. I wish you all great advances in the domain of 3D reconstruction !

References

[1] P. Alliez, L. Saboret, and G. Guennebaud, *CGAL User Manual*.

[2] J. Digne, D. Cohen-Steiner, P. Alliez, F. De Goes, and M. Desbrun, "Feature-Preserving Surface Reconstruction and Simplification from Defect-Laden Point Sets," *Journal of Mathematical Imaging and Vision*, pp. 1–14, Jan. 2013.

[3] C. Villani, *Topics in Optimal Transportation*. Graduate studies in mathematics, American Mathematical Society, 2003.

[4] D. P. Bertsekas and D. A. Castanon, "The auction algorithm for the transportation problem," 1989.

[5] Q. Mérigot, *A multiscale approach to optimal transport*, pp. 1584–1592. 2011.

[6] E. Hubert, "Convolution Surfaces based on Polygons for Infinite and Compact Support Kernels," *Graphical Models*, vol. 74, pp. 1–13, Jan. 2012.

[7] S. Shirdhonkar and D. W. Jacobs, "Approximate earth mover's distance in linear time," 2008.

[8] Y. Meyer, *Wavelets and Operators*, vol. 1. Cambridge University Press, 1993. Cambridge Books Online.