A Fast Algorithm for Computing Reeb Graph of 2-Manifold

JEAN-BAPTISTE DEBARD ♠
Tsinghua University
and
HU SHI MIN ♣
Tsinghua University

This paper presents an algorithm to compute the Reeb graph of a given Morse function over a compact connected oriented 2-manifold without boundary. After introducing the motivations for such a new algorithm, we will give the details of the algorithm for simplicial complex, prove its correctness and discuss its complexity and its running time.

Categories and Subject Descriptors: []:

1. INTRODUCTION

The notion of Reeb graph is part of the Morse theory which provides an analysis of the relationship between the topology and the geometrical information of a space given by a suitable function, a Morse function. Until recently, the main drawback for using Reeb graph analysis was the incapacity of constructing adequate Morse functions. Before the work of Ni et al. [2004], the used Morse functions gave rise to many unnecessary critical points, making the construction and the analysis of the Reeb graph expensive. The work presented in [X. Ni and Hart 2004] now allows us to precompute Morse functions with just the required number of critical points in time linear relative to the size of the model. An algorithm which computes a Reeb graph in a time that depends on the number of critical points would thus be appreciated. This paper presents such an algorithm, which in most cases runs in $O(n + m.\log(m))$ time, where n is the number of vertices and m the number of critical points.

1.1 Related Work

Reeb graphs have already been put into practice many times for surface compression and reconstruction [S. Biasotti and Spagnuolo 2000b], and for surface understanding [Y. Shinagawa and Kergosian 1991] [S. Biasotti and Spagnuolo 2000a] [M. Hilaga and Kunii 2001]. Reeb graphs are also an interesting tool for the visualization of space-time scientific data [H. Edelsbrunner and Pascucci 2004]. It is above all the work of Wood [2003] in the domain of topological simplification which motivated our research: in this work, Wood constructs a graph similar to a Reeb graph to find the handles in a compact connected oriented 2-manifold without boundary and remove them. She uses two kinds of functions (not necessarily Morse functions), a height

Author's address: ♠ yangf@cg.cs.tsinghua.edu.cn, ♣ shimin@tsinghua.edu.cn

function and a distance function, and uses some tricks in order to avoid handling unnecessary critical points. This does not always work well and can be very slow because of the computation of the distance function[1] even if done on a limited part of the mesh. Now, if we use a suitable Morse function that introduces very few critical points and a fast algorithm to compute the Reeb graph, we can do the same in a much shorter time. Besides, the only work which focuses on the properties and the computation of a Reeb graph has been done by Cole-McLaughlin et al. [2004]. The authors, after stating some major theoretical results, describe an algorithm for computing Reeb graph which theoretically runs in $O(n.\log(n))$ time, where n is the number of edges in the simplicial complex.

1.2 Contribution

The algorithm introduced in [K. Cole-McLaughlin 2004] is very fast in practice (as we are going to show in this paper) because the *a priori* slowest part is a sort of all the vertices of the simplicial complex. However, in the event of a low density of critical points, situation which arises when using a well-controled Morse function, we describe an algorithm faster than the above mentioned.

2. MORSE THEORY

For details about Morse Theory, we refer the reader to the work of Milnor [1963]. The theory of Reeb graphs of smooth functions on 2-manifolds has been introduced by George Reeb [1946]. This section reviews some basics of Morse theory for smooth functions on a 2-manifold and their adaptation to piecewise linear functions on a simplicial-complex.

2.1 Smooth Theory

For all the following, let $\mathscr{M}$ be a connected smooth oriented 2-manifold without boundary.

Definition 2.1. (*Critical Point*) Let $h : \mathscr{M} \to \mathbb{R}$ be differentiable on $\mathscr{M}$, $p \in \mathscr{M}$ is a critical point if $dh(p) = 0$

Definition 2.2. (*Hessian Matrix:*) Let $h : \mathscr{M} \to \mathbb{R}$ be 2-differentiable on $\mathscr{M}$, $p \in \mathscr{M}$ and $(U, (x, y))$ a chart of $\mathscr{M}$ s.t. $p \in \mathscr{M}$. The Hessian matrix of h at p is:

$$H(p) = \begin{pmatrix} \frac{\partial^2 h}{\partial x^2}(p) & \frac{\partial^2 h}{\partial x \partial y}(p) \\ \frac{\partial^2 h}{\partial x \partial y}(p) & \frac{\partial^2 h}{\partial x \partial y}(p) \end{pmatrix}$$

Remark 2.3. The Hessian matrix does not depend on the local parametrization.

Definition 2.4. (*Degenerate Point*) Let $h : M \to \mathbb{R}$ be 2-differentiable on $\mathscr{M}$ and p a critical point of h. p is degenerate if the Hessian matrix of h at p is singular.

Definition 2.5. (*Morse Function*) Let $h : M \to \mathbb{R}$ be 2-differentiable on $\mathscr{M}$, h is a Morse function if all its critical points are non-degenerate and if for two distinct critical points $x \neq y$ then $h(x) \neq h(y)$

[1] Dijkstra algorithm, $O(n.\log(n))$

Definition 2.6. (*Level Set*) A *level set* of the Morse function f is the preimage of a constant value $f^{-1}(\{t\})$.

Definition 2.7. (*Reeb Graph*) The *Reeb graph* $\mathbb{RG}$ of f is the quotient space of $\mathscr{M}$, with the usual quotient topology, defined by the equivalence relation: $x \simeq y$ **iff** $f(x) = f(y) = t$ **and** x and y belong to the same component of $f^{-1}(t)$.

(The example of Figure 1 may help to visualize)

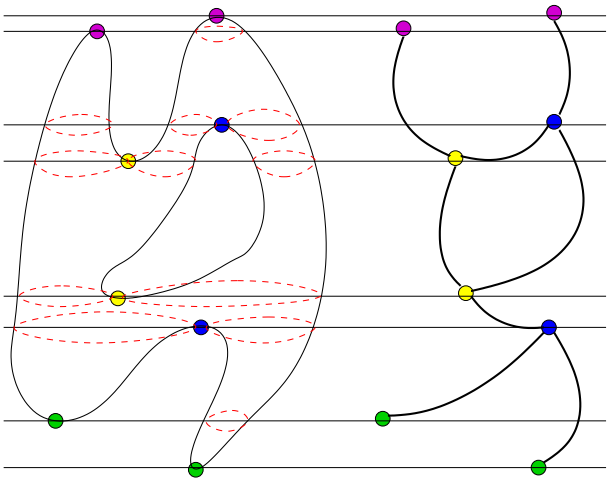

Fig. 1. A 2-manifold (left) and its Reeb Graph according to a height function along the z-axis

The proofs of the following two lemmata can be found in [Milnor 1963].

LEMMA 2.8. *(**Morse Lemma**) It is possible to choose local coordinates (x, y) at a critical point $p \in \mathscr{M}$, so that h takes the form: $h(x, y) = \pm x^2 \pm y^2$*

LEMMA 2.9. *Let $f : \mathscr{M} \to \mathbb{R}$ be a smooth function. Let $a < b$ and suppose that the compact set $f^{-1}([a, b])$ contains no critical points. Then $f^{-1}(\{a\})$ is diffeomorphic to $f^{-1}(\{b\})$, furthermore $f^{-1}(\{a\})$ is a deformation retract of $f^{-1}(\{b\})$, so that the inclusion map $f^{-1}(\{a\}) \to f^{-1}(\{b\})$ is a homotopy equivalence.*

Although based on the two preceding lemmata, the following theorem is not a direct corollary of them, but we do not deem it necessary to introduce the proof as the theorem is quite intuitive and its proof rather tiresome. The proof of the theorem 2.10 follows the proof given by Milnor in [Milnor 1963][theorem 3.2].

THEOREM 2.10. *(**Topology of the Level Set**) Let $h : \mathscr{M} \to \mathbb{R}$ Morse function on $\mathscr{M}$. Sweeping the level set of h from $+\infty$ to $-\infty$, the changes of topology of the level set occur only at critical points in the following way (cf Figure 2):*

—*sweeping through a Maximum induces a new component for the level set*

—*sweeping through a Top Saddle point makes the concerned component split into two components*

—*sweeping through a Bottom Saddle makes two concerned components merge into one*

—*sweeping through a Minimum makes the concerned component collapse*

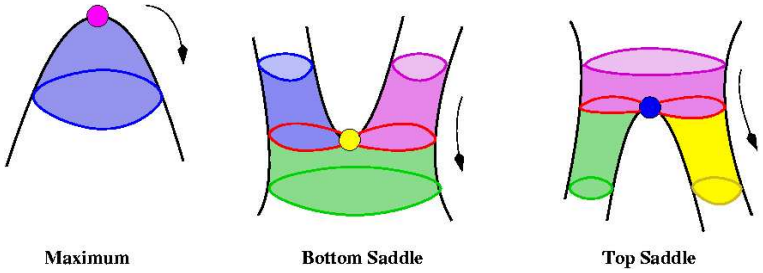

Fig. 2. Evolution of the level set passing through a critical point

We can now understand the structure of a Reeb graph as a graph whose nods are the level set components that contains a critical point and whose arcs give the connectivity of the level-set. We end the part about smooth theory with a theorem which depicts one of the great interests of a Reeb graph (its proof can be found in [K. Cole-McLaughlin 2004]).

THEOREM 2.11. *The Reeb graph of a Morse function over a connected orientable 2-manifold of genus g without boundary has g loops*

2.2 Piecewise Linear (PL) Theory

In the PL theory, the 2-manifold is still connected without boundary, but it is no longer a differential manifold. The manifold is described by a *simplicial-complex*. With appropriate definitions of a critical point and of a Morse function, the notion of Reeb graph still makes sense and the two above theorems hold in this case.

Definition 2.12. A *k-Simplex* is the convex hull of $k+1$ affinely independent points $S = \{v_0, v_1, ..., v_k\}$. The points in S are the vertices of the simplex. In $\mathbb{R}^3$ a simplex can be a point, a segment, a triangle or a tetrahedron.

Definition 2.13. (*Face, Coface*) Let σ be a k-simplex defined by $S = \{v_0, v_1, ..., v_k\}$. A simplex θ defined by $T \subseteq S$ is a face and has σ as a coface. The relationship is denoted with $\sigma \geq \theta$ and $\theta \leq \sigma$

Definition 2.14. A *Simplicial-Complex* $\mathscr{K}$ is a finite set of simplices such that $\sigma \in \mathscr{K}$, $\theta \leq \sigma \Rightarrow \theta \in \mathscr{K}$ and $\sigma, \sigma' \in \mathscr{K} \Rightarrow \sigma \bigcap \sigma' \leq \sigma, \sigma'$. The dimension of $\mathscr{K}$ is $dim\mathscr{K} = max\{dim\sigma | \sigma \in \mathscr{K}\}$

Definition 2.15. The *Underlying Space* $|\mathscr{K}|$ of a simplicial complex $\mathscr{K}$ is $|\mathscr{K}| = \bigcup_{\sigma \in \mathscr{K}} \sigma$

Remark 2.16. $|\mathscr{K}|$ is a topological space.

Definition 2.17. The *Star* of a vertex is the set of its cofaces in $\mathscr{K}$

$$St(\sigma) = \{\eta \in \mathscr{K} | \sigma \leq \eta\}$$

Definition 2.18. The *Closure* $\bar{A}$ of a set A of simplices in $\mathscr{K}$ is the smallest sub-complex of $\mathscr{K}$ which contains A

Definition 2.19. A *Function* over a simplicial-complex $\mathscr{K}$ is a piecewise linear function of $|\mathscr{K}|$ defined by assigning a value to each vertex of $\mathscr{K}$.

The following definitions are given for $\mathscr{K}$, a 2-simplical-complex whose underlying space $|\mathscr{K}|$ is an oriented connected compact 2-manifold.

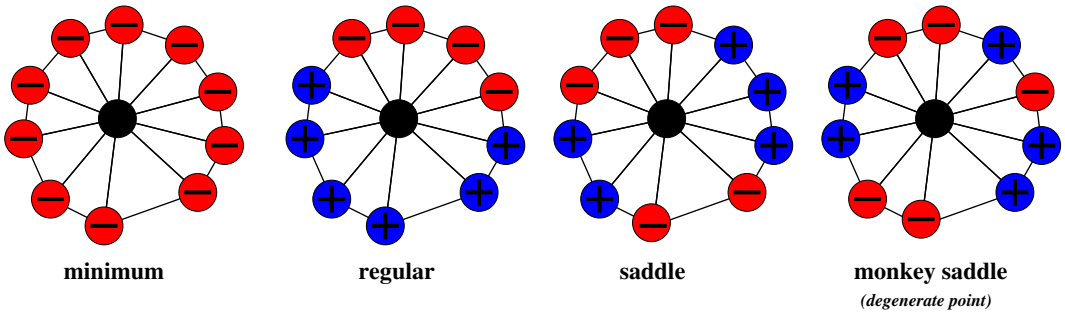

Fig. 3. The four types of critical points

Definition 2.20. (*Critical Point*) Let σ be a vertex of $\mathscr{K}$, we define 3 subsets of $\mathrm{Lk}(\sigma) = \overline{St(\sigma)} - St(\sigma)$:

$$\begin{aligned} Lk^+(\sigma) &= \{\eta \in \mathrm{Lk} | \forall \mu \in \eta,\, h(\mu) > h(\sigma)\} \\ Lk^-(\sigma) &= \{\eta \in \mathrm{Lk} | \forall \mu \in \eta,\, h(\mu) < h(\sigma)\} \\ Lk^=(\sigma) &= \{\eta \in \mathrm{Lk} | \forall \mu \in \eta,\, h(\mu) = h(\sigma)\} \end{aligned}$$

Then, σ is Critical *if* the number of components of $Lk^+(\sigma)$ is different from 1 *or if* the number of components of $Lk^-(\sigma)$ is different from 1 *or if* $Lk^=(\sigma) \neq \emptyset$
cf Figure 3

Definition 2.21. (*Degenerate Point*) A critical point is degenerate if $Lk^=(\sigma) \neq \emptyset$ or if the number of components of $Lk^+(\sigma)$ and of $Lk^-(\sigma)$ is different from 2 (cf Figure 3 -monkey saddle-)

Derived from the smooth theory, we now have exactly the same notion of Morse function and Reeb graph on $\mathscr{K}$. Furthermore, as proved in [Banchoff 1970], lemma 1 and lemma 2 still hold and with them, theorem 2.10 and theorem 2.11.

3. ALGORITHM

3.1 On a Smooth Manifold

In this part, let $\mathscr{M}$ be a smooth connected compact oriented 2-manifold without boundary. The algorithm works on the simple observation that we do not need to maintain the level set while sweeping from $+\infty$ to $-\infty$, as we know when and how the changes in the topology of the level set occur (cf Theorem 2.10), we simply need to maintain the connectivity of the level set between two following critical points.

3.1.1 *Pseudo-Code.* .

define type Region
begin: The critical point which starts the Region
end: The critical point which ends the Region
surface: The surface spanned by the Region
end define

Graph **ReebGraph**(*Manifold* $\mathscr{M}$, *MorseFunction* h)
let RG a Graph whose nodes are indexed by the critical points
let T be the table of all the critical points
sorted by decreasing values of h
let s be the size of T
let $Q = \emptyset$ a queue of Regions
let c be a counter
for $i = 1$ **to** $s - 1$ **do**
if $T[i]$ is a Maximum
then
put a new *Region* in Q which starts with $T[i]$
and whose *surface* is the region stricltly between $h(T[i])$ and $h(T[i+1])$ which is connected to $T[i]$
end if
if $T[i]$ is a *Top Saddle*
then
put 2 new *Region* in Q which both start with $T[i]$
and whose respective *surfaces* are the two disconnected regions stricltly between $h(T[i])$ and $h(T[i+1])$ which are connected to $T[i]$
end if
if $T[i]$ is a *Bottom Saddle*
then
put a new *Region* in Q which starts with $T[i]$
and whose *surface* is the region stricltly between $h(T[i])$ and $h(T[i+1])$ which is connected to $T[i]$
end if
$c \leftarrow 0$
for each *Region* $R \in Q$ **do**
expand the *surface* of R with the connected surface stricltly between $R.begin$ and $h(T[i+1])$ and connected to the previous *surface* of R
if R is connected to $T[i+1]$
then
link R.start and $T[i+1]$ by an *Edge* in the *Reeb Graph* RG
delete R from the queue Q
increment c
end if
end do
if $T[i+1]$ is a *Saddle*

```
      then
      if c = 1
        then T[i + 1] is a Top Saddle
      end if
      if c = 2
        then T[i + 1] is a Bottom Saddle
      end if
    end if
  end do
  return RG
```

3.1.2 *Proof of the Algorithm*

Definition 3.1. (*Region*) Let h be a Morse function on $\mathscr{M}$, a Region $\mathscr{R}$ of h on $\mathscr{M}$ is a maximal connected subset of $\mathscr{M}$ which verifies the following conditions: its image by h lays strictly between the image by h of two critical points x and y, it is connected to x and y and does not contain any critical point.
In this case, we say that the Region connects x and y and if $h(x) > h(y)$, we say that $\mathscr{R}$ begins with x and finishes with y. (cf. Figure 4)

Remark 3.2. According to the properties of a Region, $\{x\} \bigcup \mathscr{R} \bigcup \{y\}$ is connected in $\mathscr{M}$.

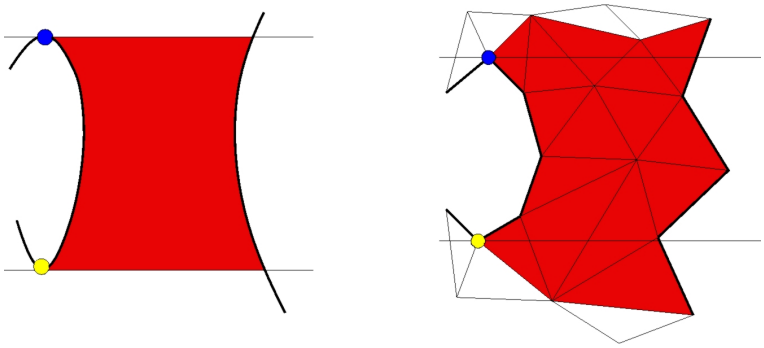

Fig. 4. Example of a region (red) between a Top Saddle and a Bottom Saddle on a smooth manifold (left) on a triangular mesh (right)

It is intuitive that a one-to-one correspondence between the Regions of $\mathscr{M}$ and the arcs of $\mathbb{RG}$ exists. Furthermore, if there is an arc connecting two nodes corresponding to the critical points x, y, then there exists a Region which connects x and y. From this statement, it is now easy to prove the algorithm.

Theorem 3.3 Loop Invariant. *At each step of the main loop:*

—*For each Region which finishes with cp_2 an edge is created in the graph.*
—*All the Regions which begin with x such that $h(x) \geq cp_1$ and which finish with y such that $h(y) < cp_2$ are in the queue Q*

PROOF OF THE LOOP INVARIANT. It is sufficient to prove that all the Regions which begin with cp_1 do not exist before the current loop and that they are all created during this loop.
As a Region is only connected to two critical points (the one which begins it and the one which finishes it), then no other critical point before cp_1 in the table T can begin the Region beginning with cp_1. Thus, at the beginning of the loop, the Regions which begin with cp_1 are not in the queue Q.
By Theorem 2.10, all the Regions begining with cp_1 are put in the queue Q during the current loop. □

PROOF OF THE ALGORITHM. As the number of critical points is finite, the algorithm necessarily terminates. At the end of the algorithm, according to the Loop Invariant, all the Regions which finishes with a critical point have been handled except for the Regions which finish with the first critical point. But, as the first critical point is necessarily a Maximum, there is no Region finishing with it. Thus, at the end of the algorithm, all the Region of the manifold $\mathscr{M}$ have been considered and all the associated edges in the graph have been created. As there is a one-to-one correspondence between the Regions of $\mathscr{M}$ and the Edges of $\mathbb{RG}$, all the Edges of $\mathbb{RG}$ appear in the created graph. Lastly, as $\mathbb{RG} = \{[x] | x \in \mathscr{C}\} \bigcup \{\mathscr{E} \in \mathbb{E}\}$, the created graph is the true Reeb Graph of $\mathscr{M}$. □

3.2 On a Simplicial-Complex

On a 2-simplicial-complex $\mathscr{K}$, whose underlying space $|\mathscr{K}|$ is a compact oriented connected 2-manifold, the algorithm is exactly the same except that the definition of a Region has to be modified.

Definition 3.4. A *Region* between two critical points x and y ($h(x) < h(y)$) is the maximal set of open simplices which intersect $h^{-1}(]h(x), h(y)[)$, does not contain any critical point and whose spanned surface is connected and connected to x and y. cf Figure 4

Now, it is no longer true that a region is connected with only two critical point, it can be connected with more than two. Furthermore, two Regions can overlap and the collection containing the Regions together with the level sets of the critical points is no more a partition of $|\mathscr{K}|$.
However, it is still true that there is a one-to-one correspondence between Regions of $\mathscr{K}$ and Edges of $\mathbb{RG}$ and we only need one last lemma to prove the rightness of the Loop Invariant.

LEMMA 3.5 END OF A REGION. *The critical point which finishes a Region $\mathscr{R}$ beginning with y is the critical point x such that:*
$h(x) < h(y)$ *and* $\forall z \in \mathscr{C}$ *s.t.* $\{z\} \bigcup \mathscr{R}$ *is connected, then* $h(z) < h(x)$

PROOF. Let us assume there exists a critical point $z \in \mathscr{C}$, such that $\{z\} \bigcup \mathscr{R}$ is connected and $h(x) < h(z) < h(y)$. Then, the vertex z is a simplex connected to $\mathscr{R}$ and one which belongs to $h^{-1}]h(x), h(y)[$. As $\mathscr{R}$ is maximal, then $z \in \mathscr{R}$ which leads to a contradiction since $\mathscr{R}$ does not contain any critical point. □

PROOF OF THE ALGORITHM. As the critical point which begins a Region is always known, and by the Lemma 3, the Loop Invariant given for the smooth case

is still true for the PL case. The reasoning of the proof for the algorithm in the smooth case still holds in the PL case. □

3.3 Complexity

3.3.1 *Worst Case.* Finding the m critical points takes $O(n)$ time (n is the number of vertices), sorting them takes $O(m.\log(m))$. Growing a Region takes the size of a Region in time and as a face could belong to every Region, then growing Regions takes $O(n.m)$ time in the worst case. The whole algorithm thus takes $O(n.m + m.\log(m))$ worst case in time and obviously $O(m + n)$ in size.
This worst case can eventually come about as shown in Figure 5 (infinitely many blue triangles can be "glued" to the mesh and, with a height function as given Morse function, the blue triangles then belong to more than $m/2$ Regions)

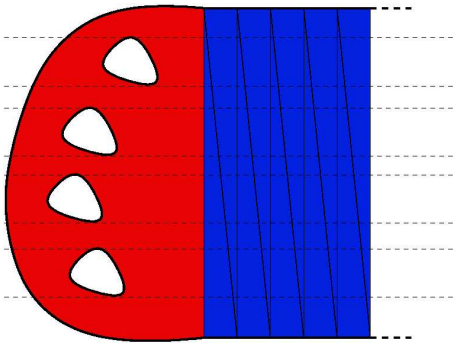

Fig. 5. Example of a worst case situation

3.3.2 *Is it that bad in practice?.* The worst case complexity is pessimistic and the table 6 presents the results when we experimentally count the average number of Regions a face belongs to.

The average number of Regions for a face is still reasonable and far smaller than the number of critical points. Furthermore, this number increases with the density

Mesh	nb faces	nb cp	mean
eight	25334	6	1.050
chetha	9996	74	2.290
chetha'	9996	36	1.459
budha 10K	21152	304	8.997
Buddha' 10K	21152	138	2.287
Buddha 30K	62224	450	8.086
Buddha' 30K	62224	198	2.082
David head	63618	260	3.127
David head'	63618	336	4.571
dragon leg	95410	224	3.032

Fig. 6. Experiment on the mean number of Regions a face of a mesh belongs to (the prime ' means the model has been tested with two different Morse functions, nb cp is the number of critical points for a given model and a given Morse function)

Mesh	nb v	nb cp	running time of GR	running time of SLS
eight	20K	6	CP: 20ms GR: 40ms	SORT: 30ms SLS: 40ms
chetha	5K	74	CP: 10ms GR: 20ms	SORT: 10ms SLS: 20ms
buddha	10K	138	CP: 20ms GR: 60ms	SORT: 30ms SLS: 40ms
	30K	198	CP: 80ms GR: 150ms	SORT: 70ms SLS: 150ms
	60K	230	CP: 190ms GR: 240ms	SORT: 140ms SLS: 270ms
David	40K	336	CP: 50ms GR: 260ms	SORT: 70ms SLS: 170ms
Dragon Leg	40K	172	CP: 70ms GR: 190ms	SORT: 100ms SLS: 200ms

Fig. 7. Comparison of the running time for the GR-algorithm and the SLS-algorithm

of critical points, and highly depends on the choice of the Morse function. Thus, when precomputing a good Morse function, we can hope to greatly reduce the two numbers.

3.3.3 *Comparisons.* The fastest algorithm provided by [K. Cole-McLaughlin 2004] is proved to run in $n.\log(n)$ and is extremely fast in practice: the algorithm maintains the level set of the given Morse function while sweeping the level from $-\infty$ to $+\infty$. Thus, it relies on an pre-ordering of all the vertices of the mesh according to the given Morse function which can be done with a quick sort. The implementation of the data structure for the level set components can be done using chained lists. This allows querying, inserting and removing an edge in constant time and merging and splitting in time the size of the list. But as these two last operations happen only at saddle points, we can hope them to be relatively rare and thus not so costly (this is verified in practice). We thus obtain an algorithm which performs a sort in $n.\log(n)$ expected time and the sweep line in (almost always) linear time. In practice, this algorithm is very fast because the *a priori* slowest part is the sorting of all the vertices which can be done with a quick sort.
The table 7 presents the comparison of the running time for the 'growing region' (GR) algorithm and the 'sweeping level set' (SLS) algorithm, CP is the time to compute critical points and sort them, SORT is the time for sorting all the vertices of the mesh. In the tests, our algorithm is hardly faster than the one of [K. Cole-McLaughlin 2004] and even slower when the density of critical points is high. Nevertheless, it is worth considering two remarks: first, no other code optimization can be expected for the sweep algorithm as the slowest part of the algorithm relies on a sorting and sorting algorithm have been optimized up to the highest degree. Second, it is advisable to look at the evolution of the GR-algorithm running time when the size of the mesh increases while the density of critical points does not. What is shown in Figure 8 is the evolution of the two compared algorithms when the Morse function gives the minimal number of critical points (one maximum, one minimum and $2.g$ saddle points where g is the genus of the model) on meshes of increasing size. The models used for the experiment pictured in the graph of Figure 8 are all buddha models of different sizes and the Morse function is computed using the method described in [X. Ni and Hart 2004].

3.3.4 *Discussion.* This algorithm is limited to oriented 2-manifolds without boundaries, but it is possible to extend it to oriented 2-manifolds with boundaries and even to non oriented 2-manifolds. The main problem comes from the fact that it is no more possible to predict the configuration of critical points only from

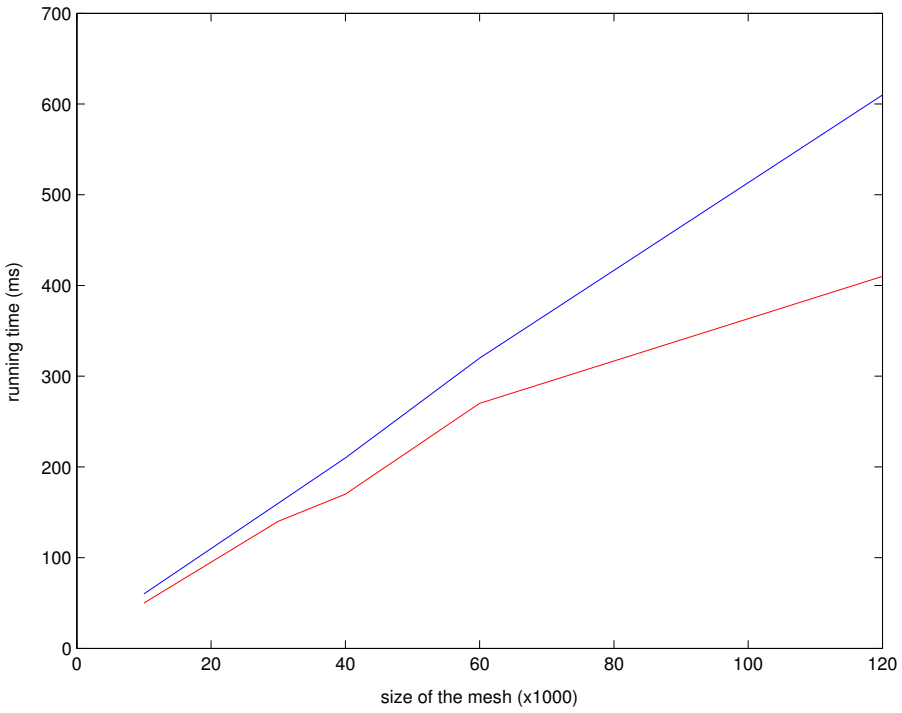

Fig. 8. compared running times of GR (red) and SLS (blue)

Regions that connect it from 'above': when the 2-manifold has boundaries, a Morse function can give rise to saddle points connected with 4 Regions (cf Figure 9). When the 2-manifold is non-oriented, a Morse function can give rise to saddle points connected with only 2 Regions (cf [K. Cole-McLaughlin 2004]). However, we did not feel the need to implement such an algorithm as the property of Reeb graphs given by theorem 2.11, which motivated us holds neither for non-oriented 2-manifolds nor for 2-manifolds with boundaries as shown in [K. Cole-McLaughlin 2004].

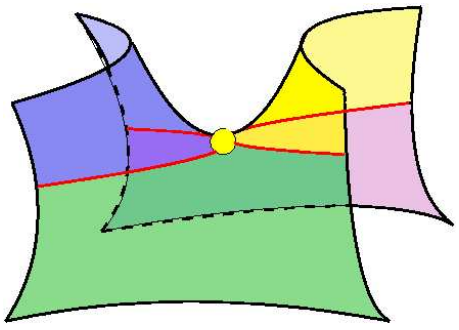

Fig. 9. Example of a saddle with four connected Regions on a 2-manifold with boundary

4. CONCLUSION

We have presented a new algorithm for computing the Reeb graph of an oriented 2-manifold without boundary. We have shown that this algorithm does not achieve the best complexity in the worst case but that a situation in which it is worth using a Reeb graph, this algorithm is much faster than previous algorithms. The original purpose of this algorithm was to capture the topology of scanned models (which are or can easily be connected compact oriented 2-manifolds) to retrieve handles and tunnels as did Wood in [Wood 2003]. One drawback of the Wood's method is that it computes a kind of graph whose "Morse" function[2] does not depend on the model. This can lead to many useless critical points as shown in [X. Ni and Hart 2004] and Wood consciously chose not to compute a true Reeb graph because of this problem. Now, if the GR-algorithm is used with a Morse function which minimizes the number of critical points (namely, for a connected compact oriented 2-manifold, one maximum, one minimum and $2.g$ saddle points), it can be extremely fast (much faster than the Sweep-algorithm as shown in Figure 8). Such a good Morse function can be computed in linear time, which depends on the size of the mesh [X. Ni and Hart 2004]. The whole algorithm can then run in linear time in most cases ($g << n$) and in $O(g.n + g.log(g))$ time in the worst case.
In conclusion, as our algorithm does not provide an improvement of the worst case complexity, the complexity of $O(n.\log(n))$ achieved by the algorithm of [K. Cole-McLaughlin 2004] is still the best in the worst case and proving wether or not this complexity is optimal remains an open question.

ACKNOWLEDGMENTS

We wish to thank BLABLABLA

REFERENCES

Banchoff, T. 1970. Critical points and curvature for embedded polyedral surfaces. *American Mathematical Monthly 77*.

H. Edelsbrunner, J. Harer, A. M. and Pascucci, V. 2004. Time-varying reeb graph for continuous space-time data. *Annual Symposium on Computational Geometry*.

K. Cole-McLaughlin, H. E. 2004. Loops in reeb graph of 2-manifold. *Discrete Computational Geometry*.

M. Hilaga, Y. Shinagawa, T. K. and Kunii, T. 2001. Topology matching for fully automatic similarity estimation of 3d shapes. *Proceedings of SIGGRAPH*.

Milnor, J. 1963. *Morse Theory*. Princeton Univ.Press.

Reeb, G. 1946. Sur les points singuliers d'une forme de pfaff complètement intégrable ou d'une fonction numerique. *Comptes Rendus de l'Academie des Sciences, Paris 222*.

S. Biasotti, B. F. and Spagnuolo. 2000a. Extended reeb graphs for surface understanding and description. *Proc. of 9th Discrete Geometry for Computer Imagery Conference*.

S. Biasotti, M. M. and Spagnuolo. 2000b. Surface compression and reconstruction using reeb graphs and shape analysis. *Proc. of Spring Conference on Computer Graphics*.

Wood, Z. 2003. Computational topology algorithm for discrete 2-manifolds. Ph.D. thesis, California Institute of Technology, Pasadena, California.

X. Ni, M. G. and Hart, J. C. 2004. Fair morse functions for extracting the topological structure of a surface mesh. *Proceedings of SIGGRAPH*.

[2]here, it is also not necessarily a true Morse function

Y. Shinagawa, T. K. and Kergosian, Y. 1991. Surface coding based on morse theory. *IEEE Comput. Graphics Appl.*