EV: A First Version of the Surface Evolver with a Human-Computer Interface

Ivana Soares Bandeira[1], Antonio Elias Fabris[1*], Valério Ramos Batista[2]

[1] IME-USP, r. do Matão 1010, 05508-090, São Paulo-SP, Brazil.
[2] CMCC-UFABC, av. dos Estados 5001, 09210-580, St André-SP, Brazil.

* Corresponding author. Tel.: +55(11)30916136; email: aef@ime.usp.br.

Manuscript submitted June 13, 2016; accepted August 12, 2016.
doi: 10.17706/jsw.11.11.1073-1082

Abstract: The Surface Evolver is a general-purpose software with which one can simulate innumerous physical experiments. Since its first release in 1989, its sole developer and maintainer K. Brakke prioritised the technical part. This made Evolver applicable in many Science Areas, but its input is only through data files and command lines. Here we present EV, the first human-computer interface for Evolver totally written in Matlab. By means of eye tracker or head mouse the user can access virtual buttons and perform simulations compatible with Evolver. Although still restricted to the numerical solution of the Plateau problem, its Matlab code is at least 90% shorter than the corresponding C-code of Evolver. This will facilitate both the maintenance and the use of EV, specially if we consider physically challenged people in academia.

Key words: Surface-evolver, human-computer interface, plateau problem, matlab, accessibility design.

1. Introduction

The Surface Evolver is a powerful tool that has innumerous applications in many Areas of Knowledge. Among others we cite Aerodynamics [1], Fluid Dynamics [2]-[4] and Medicine [5]-[8]. Created in 1989 by K. Brakke, the Surface Evolver is nowadays in the version 2.70 [9] with 118 source files (extensions .c and .h) and almost 200 thousand lines of source code. It handles forces (contact, gravity, etc.), pressures, densities, n-dimensional spaces (including non-Euclidean), tracking of quantities and prescribed energies, among several other features.

Along almost 30 years the Surface Evolver became very complete due to the fact that K. Brakke prioritised its technical part. Consequently, its input is all by command lines and data files. A priory, the absence a Graphical User Interface (GUI) for the Surface Evolver seems to be of minor importance. However, one of the commonest present day facilities is the human- computer interaction. Resources like virtual buttons and voice commands have long became part of our daily tasks, even if specific to academia. Particularly, human-computer interfaces are highly desirable in the case of Physically Challenged People (PCP).

Such interfaces did not exist a few decades ago, when the Surface Evolver was created. Already in the version 1.0 it included its own method of Numerical Solution of the Plateau Problem (NSPP). This problem consists of finding the minimal surface, namely the one of minimum area that spans a given closed curve in

the 3D Euclidean space. Physically speaking, this surface corresponds to the film that forms when we dip a wire loop into soapy water. In its classical formulation the NSPP ignores gravity and thickness. Yet its part in the Surface Evolver source code takes tens of thousands of lines, if we also count all the commands of a typical simulation (acc. personal communication).

In this work we present EV, a Matlab program to the NSPP that uses the same method of the Surface Evolver. It was first introduced in [10], when still under development. The difference is that EV includes a GUI endowed with interaction through either eye tracking or head mouse. They enable the user to click on virtual buttons and perform simulations compatible with the Surface Evolver. For the time being, its program in Matlab has only 1,343 lines of source code, which is less than 90% of the part of Surface Evolver devoted to the NSPP. This makes EV easy in three important aspects: maintenance, improvements and addition of new resources.

This paper is divided as follows: Section 2 gives some brief explanation about our choice of human-computer interaction. In Section 3 we demonstrate how to use EV with an example, and explain its accessibility through eye tracker and head mouse. Some technical details of EV are presented in Section 4. Finally, our conclusions are drawn in Section 5, where we also list some future developments for EV.

2. Eye Tracking and Head Mouse

In the case of PCP that are unable to use either hands or feet when working on the computer, some good alternatives are voice commands, speech recognition, touchpad, head mouse and eye tracking. Since our work is devoted to the NSPP with a graphical interaction, this naturally includes drawing. In this case, neither voice commands nor speech recognition stand for the best choice if you must start with at most two alternatives.

The touchpad is practical only for PCP that are skilled at writing and drawing with the mouth. Hence our choices for EV were eye tracking and head mouse. The two devices are of fundamental importance for the human-computer interaction of people that have got a severe impairment. This could have been caused by a car accident, a degenerative disease, etc. For an overview, see [11-14].

The program HeadMouse was created by the Robotics and Signal Processing Group of the Lleida University, in Spain. Like the Surface Evolver, the HeadMouse is free of charge. It works by picking up head and face movements though a simple webcam. For instance, one can adjust it to make the opening and • Š — – – ‹ • ‰ ‘ ˆ – Š ‡ • ‘ — – Š ™ ‘ ” • ƒ • ò … Ž ‹ … • ‹ • ‰ ™ ‹ – Š – Š ‡ • ‘ — • ‡ Ž ‡ ˆ – „ — – operating system, and its use is impracticable for PCP that cannot move the head any longer. See http://robotica.udl.cat/headmouse.htm for details. Therefore, we decided to program EV in such a way that it could be used even for PCP that can only move the eyes. There are many eye tracking devices, but probably the most comfortable is the Eye Tribe Tracker. Created by a Danish enterprise, this tracker consists of a case box with photosensitive cells, which is attached to a tablet or monitor. See Figure 1a for an illustration. The photosensitive cells detect eye movements and make the mouse cursor displace accordingly.

In the Surface Evolver one describes initial surfaces with a datafile. They are polyhedral surfaces that can be refined and smoothened by typical commands of the Surface Evolver. With EV one draws the boundary curve of this initial surface with a mouse, as depicted in Fig. 1b.

The reader can access the following video for a quick overview of how to operate both EV and the Eye Tribe Tracker: https://www.youtube.com/watch?v=kfOlX7V6QV0 .

As one can see in the video, Eye Tribe picks up both eye and pupil movements, and it may lose track of them in case of head movement, even if just subtle. This hypersensitivity is mentioned in the video, where one also sees that the mouse cursor jumps abruptly from region to region of the screen due to the so-called

saccades. Moreover, pupils quiver constantly and this makes the cursor tremble too. In this sense the HeadMouse is more stable.

However, the Eye Tribe was recently released in a new version that now compensates head movements. Moreover, we can train the eyes for them to move slowly, as it finally succeeds in the video. Anyway, we have not implemented EV to draw space curves with Eye Tribe yet. This is possible with HeadMouse but requires training even so. In Section 5 we shall discuss future implementations of EV that include strategies and alternatives to this problem.

In our present version of EV the user chooses virtual buttons by moving the mouse cursor with the eyes. In order to click on a certain button the cursor must hover inside the button area for at least 2.5s. One could get the click by blinking but experience shows it to be tiring if done frequently for more than 5min.

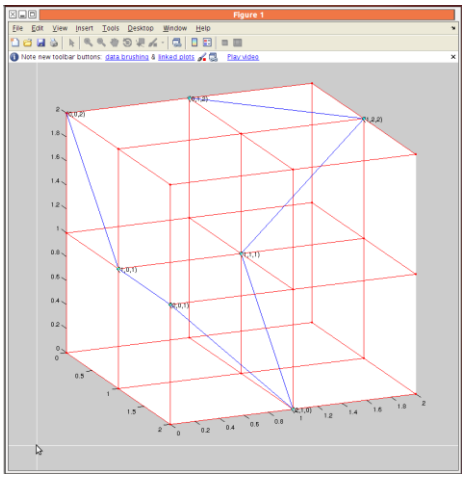


Fig1. (a) Eye Tribe Tracker. (b) Drawing a space curve in EV.
https://theeyetribe.com/products

3. Using EV

As a matter of fact, either Eye Tribe or HeadMouse are just optional to run EV. Matlab R2013a is required and any other version should be avoided, as we shall explain in Section 4. Invoke ev at the Matlab prompt and you will get our GUI as depicted in Fig. 2. Initially, only buttons DRAW, READ (from file) and EYE TR are enabled.

Pressing the EYE TR button makes mouse clicks unnecessary for any virtual button, which from then on are pressed just by letting the mouse cursor hover inside its area for 2.5s. However, the option DRAW requires using a (head) mouse conventionally: click left to mark vertices of a polygonal space curve, as illustrated in Figure 1b. The curve closes when you either mark again the same vertex or click right anywhere. This will finish drawing and all buttons are then enabled automatically. The same happens when you read a surface from file.

Now we give a brief explanation of the other buttons: REFINE subdivides each triangle into four congruent ones; ITERATE computes the overall energy and displaces vertices in order to minimise it; EQUIANG shortens edges as exemplified in Fig.3g.VERTEX AV displaces each vertex towards the weighted average of the centroids of its faces (detailed in Section 4); RM TINY EDGS removes all edges whose length is shorter that a cutoff given by the user; SAVE stores the current surface in a typical Evolver file (with .fe extension); CLOSE terminates EV. In the Surface Evolver their respective commands are: r, g, u, rawv, t, d and q.

After pressing any virtual button the cursor goes to the centre of the surface. This avoids pressing the

same button twice by accident, which is specially important when the function EYE TR is activated.

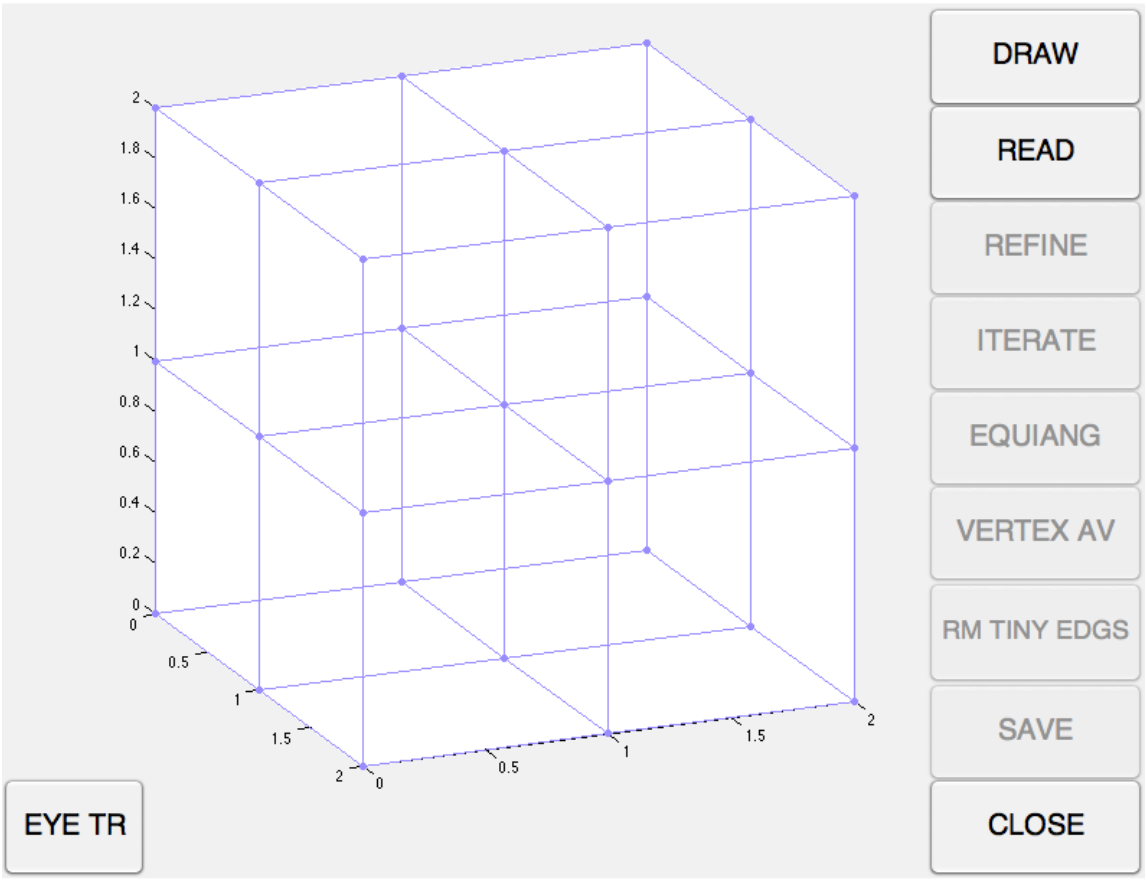


Fig. 2. Getting started with EV.

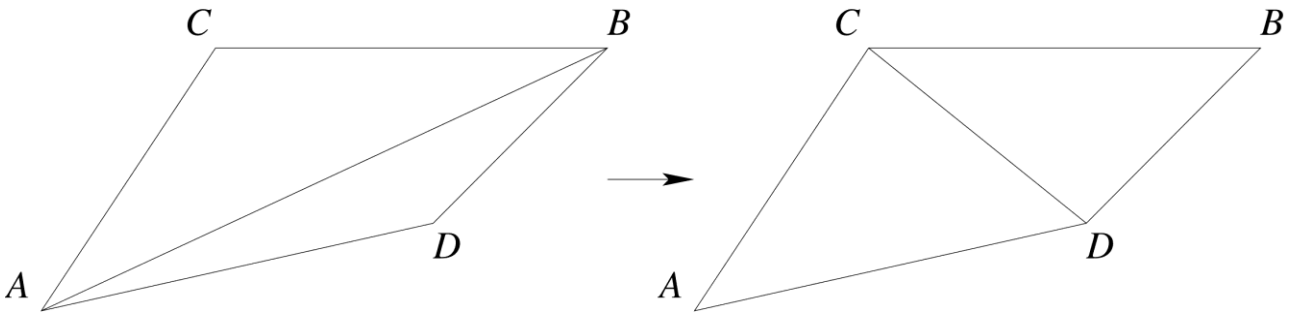


Fig. 3. Equiangulation.

Next we exemplify a simulation with EV. Consider the surface in Fig. 4a. By invoking REFINE just once we get Fig. 4b.

Now we apply the sequence ITERATE (twice), REFINE, EQUIANG (twice), VERTEX AV, ITERATE, EQUIANG and VERTEX AV. During this and any other simulation it is advisable to rotate the surface. This helps choose a short sequence of commands that leads to a smooth surface, as exemplified in Fig. 5.

Next we choose RM TINY EDGS with Cutoff Length = 0.2. In its present version EV requires the user to type it at the Matlab prompt. The cutoff can be decided through the analysis of a histogram of edge lengths, as depicted in Fig. 6a. Right afterwards we finally get Figure 6b.

In EV the vertical buttons are ordered in such a way that typical simulations with the Surface Evolver

would follow them from top to bottom, even if not in a strict sense.

4. Technical Part

Here we just give some general explanation since the algorithms of EV are taken from the Surface Evolver, more specifically from Chapter 16 of [9]. Regarding the GUI presented in Fig. 2, it was designed with the tool guide of Matlab.

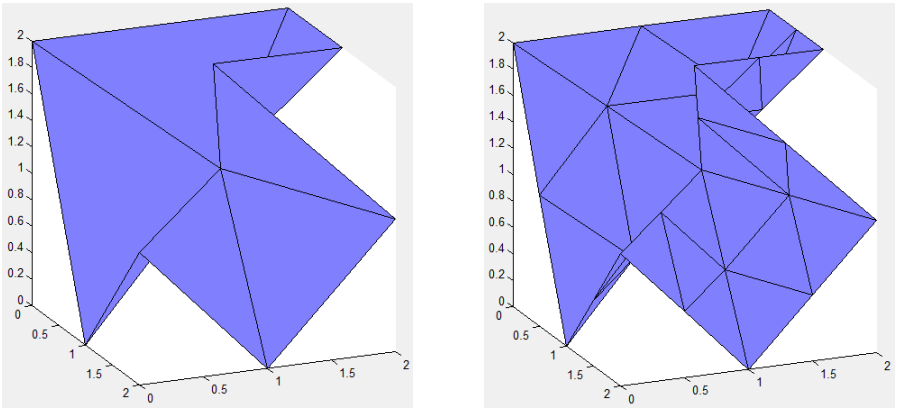


Fig. 4 (a) Initial surface. (b) Outputof REFINE.

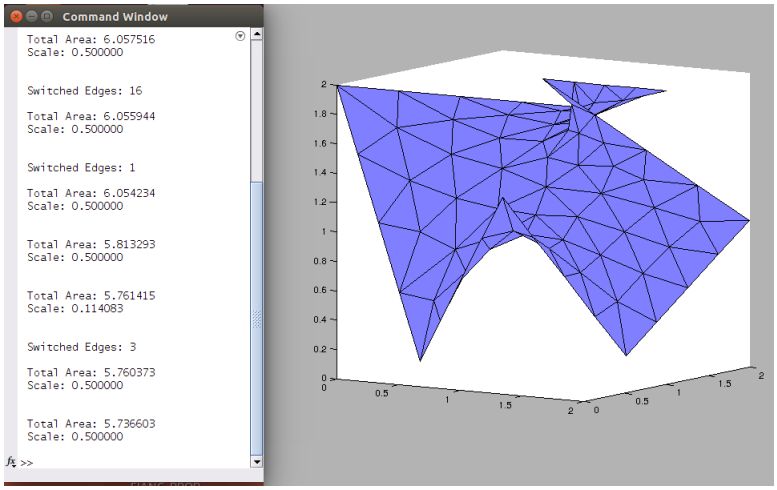


Fig. 5. Result after applying the above sequence of commands.

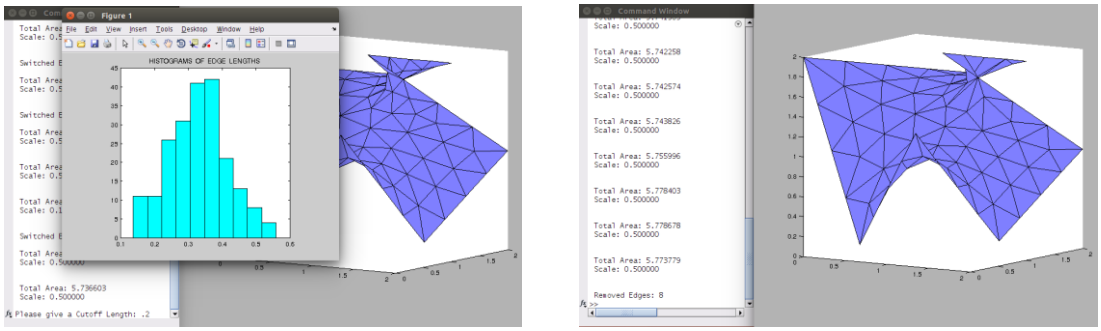


Fig. 6 (a) Histogram of edge lengths. (b) Final minimal surface.

As depicted in Fig. 1b, vertices of a 3D grid are marked on a 2D projection of this grid. This is possible because EV calls two important routines from the MathWorks File Exchange Community: plotgrid.m and

select3d.m, written by Bruno Luong and Joe Conti, respectively. An important detail is that select3d.m runs in any Matlab version from 7.8 to 8.1, but not in later versions. This is because functionalities and properties of some graphical objects were altered in recent versions of Matlab, and the ones used in select3d are now incompatible with them.

As mentioned in the Introduction, EV is easy to maintain, to improve and to acquire new resources. This assertion is now explained by Fig. 7, which shows the hierarchical diagram between ev.m and each of its sub-programs.

We conclude this section by explaining the function VERTEX AV. Here we only illustrate its functioning, since details are given in Subsection 16.27 of [9]. Consider the example depicted in Fig. 8, where one starts with a polyhedral surface S with three faces f, each in one of the fundamental planes of Oxyz.

The vertices of S are the origin and the fundamental vectors $\vec{i}, \vec{j}, \vec{k}$, with these three fixed. We can look at S as if it were a bottomless pyramid. In the case of more than three faces the "medium plane" is located in a "medium" position. In our example the mean of all vertices of S except the origin is $\vec{m} = (1,1,1)/3$. Hence the point $\vec{m} = (1,1,1)/3$ and the vector $\vec{N}$ explained in the sequel will both determine that "medium plane".

According to Equation 16.159 of [9], S has a total normal given by

$$\vec{N} = \sum_{f \in S} \vec{N}_f = \frac{1}{2}(1,1,1),$$

where $\vec{N}_f$ is the normal vector to face f with length equals to its area. On the one hand, then we must take $\vec{v} = \vec{m}/3$ to get $\vec{v} \cdot \vec{N} = 1/6$. On the other hand, by taking the centroids $\vec{x}_{centr}$ of each $f \in S$, Equation 16.157 of [9] gives

$$\vec{v} = \frac{\sum_{f \in S} \text{area}(f) \cdot \vec{x}_{centr}}{\sum_{f \in S} \text{area}(f)} = (2,2,2)/9 .$$

Hence vertex $(0,0,0)$ of S must be displaced to certain $\vec{v}_{new}$ that depends on the type of averaging. For instance, if we want to keep volume then $\vec{v}_{new} = \vec{v}_{avg} - \lambda \vec{N}$, where λ verifies $(\vec{v}_{avg} - \lambda \vec{N}) \cdot \vec{N} = \vec{v} \cdot \vec{N}$. Namely,

$$\lambda = \frac{(\vec{v}_{avg} - \vec{v})}{\vec{N} \cdot \vec{N}} \cdot \vec{N} .$$

In our example we would have $\lambda = (4/6)/3 = 2/9$, where $\vec{v}_{new} = (2,2,2)/9 - (1,1,1)/9 = \vec{v}$. Thus, in practice the vertex $(0,0,0)$ of S would not move, because the surface is already symmetric with respect to 120^o rotations around the diagonal of the 1st octant.

Surfaces with non-zero constant mean curvature are volume-preserving. In their case we ought to apply vertex averaging with the command V of the Surface Evolver. But the NSPP is like a soap film, in which case $\lambda = 0$.

Therefore, we have programmed vtxav.m for not to keep volume. This corresponds to the command rawv of the Surface Evolver. But for future developments vtxav has a variable vpr that enables volume-preserving if vpr~=0. Since we aim at the NSPP, then vpr is always zero in the present version of EV.

By invoking vtxav just once we get Figure 8b. Vertex averaging can be applied several times, and in our example we have the convergences $\vec{v} \to 0$ and $\vec{v}_{avg} \to (1,1,1)/3$. See Fig. 8c.

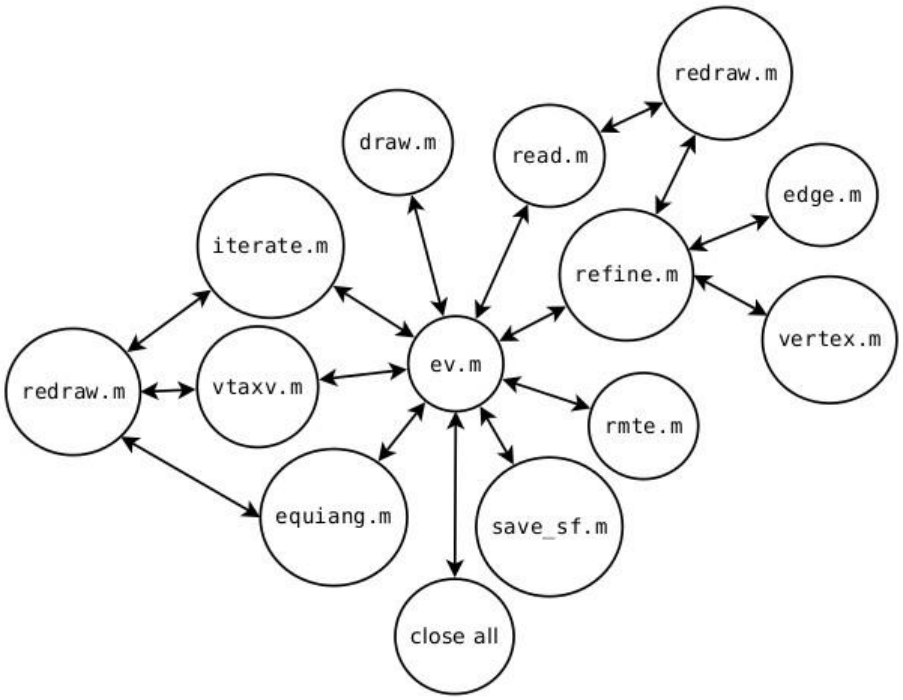
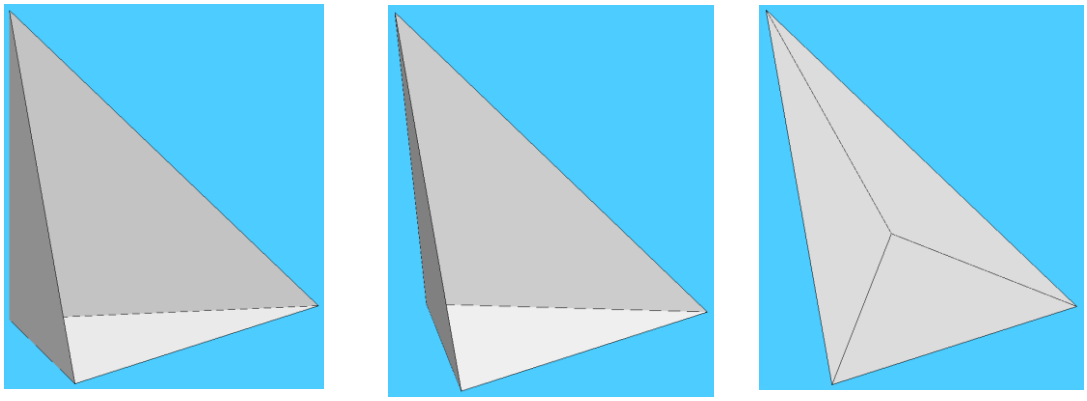


Fig. 7. Hierarchy between ev.m and its sub-programs.



(a) (b) (c)

Fig. 8. Steps of VERTEX AV.

5. Conclusions

We have obtained EV, the first version of the Surface Evolver endowed with a human-computer interface. With either the Eye Tribe Tracker or the HeadMouse one carries out simulations like done in the Surface Evolver. In its present version EV deals only with the NSPP, but it is structured to facilitate addition of resources, as well as maintenance and improvements. Its source code is fully written in Matlab and consists of only 1,343 lines, which is less than 10% of the C-code of the Surface Evolver devoted to the NSPP.

The Surface Evolver is fully open source, and in future we shall make the same to EV. Now we present some strategies and alternatives to improve it. In Section 2 we commented some limitations of the Eye Tribe, but one can get round them by implementing the following resources:

1) Division of the screen monitor in rectangular regions like the cells of a matrix. Running EV with Eye Tribe will enable more control over the mouse cursor if we make it centered inside each cell at which the user will be looking at.
2) Addition in EV of the sub-program magnify.m, from the MathWorks File Exchange Community, created by Rick Hindman. This program zooms details of a figure and makes it easier to keep the look at the magnified region. See the example in Figure 9.
3) Combination of strategies 1 and 2, which would not only make it easier to draw a polygonal curve with the eyes, but would also enable them to input some data in EV that still depend on the Matlab prompt. For instance, in Section 3 we explained the function RM TINY EDGS, which requires typing a cutoff. This could also be done with virtual buttons, like in the code of the CASIO calculator of the MathWorks File Exchange Community. See http://www.mathworks.com/matlabcentral/fileexchange/41440-simple-calculator for details.

Besides the above strategies we also plan to add new resources to EV, not only from the Surface Evolver but also from other kinds of human-computer interaction: touchpad, voice command, etc. Finally, it is important to remark that our program does not use any Matlab toolbox, and therefore its Student Version is enough to run and develop EV.

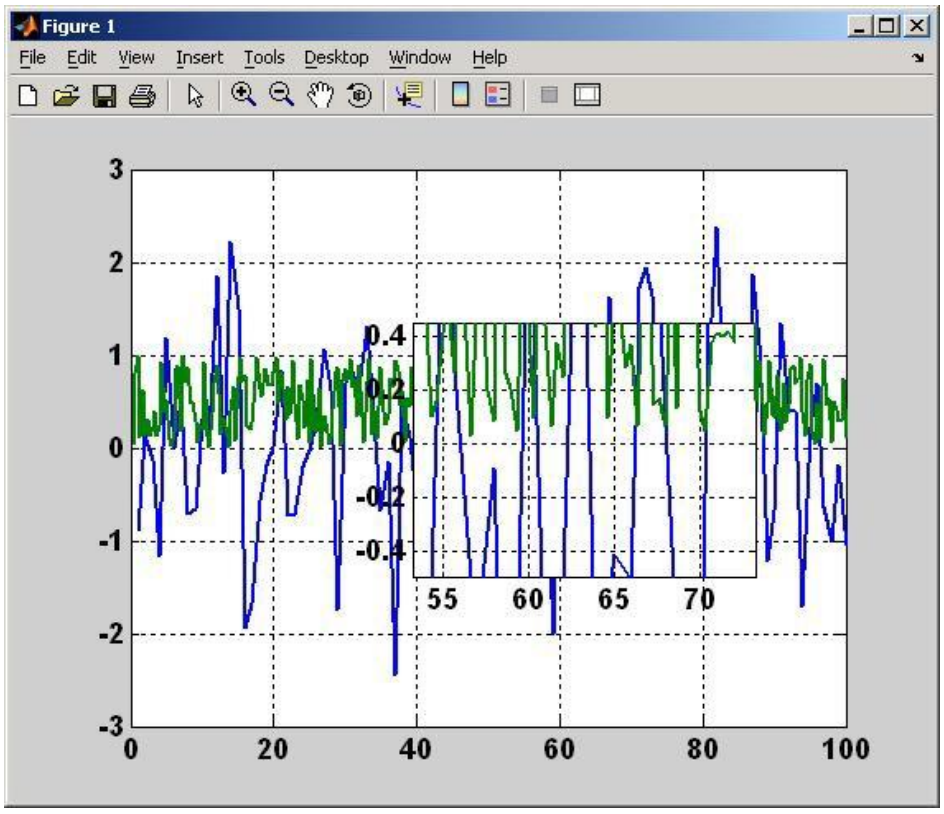


Fig. 9. Using magnify.m
Source:www.mathworks.com/matlabcentral/fileexchange/5961-magnify

References

[1] Baginski, F., & Brakke, K. A2012). Deployment analysisof pneumatic envelopes including ascending balloons and inflatable aerodynamic decelerators. *Journal of Spacecraft and Rockets*, *49*, 413 421.

[2] Chen, C., Duru, P., Prat, M., Joseph, P., & Geoffroy2015). Towards the computation of viscous flow resistance of a liquid bridge. *International Journal of Computational Methods and Experimental Measurements* , *4*, 42 49.

[3] Cunsolo, S., Baillis, D., Bianco, N., Naso, V., Oliviero, M., Lewis, R., & Massarotti, N. (2016). Effects of ligaments shape on radiative heat transfer in metal foams. *International Journal of Numerical Methods for Heat and Fluid Flow,* 26.

[4] Rondon, C. H., & Ramos, B. V. (2016). Computational modelling of a spanning drop in a wedge with variable angle. *International Journal of Undergraduate Research and Creative Activities*, *6*.

[5] Do, N. M. Z., & Ramos, B. V. (2014). First steps to virtual mammography with the surface evolver. *Recent Patents on Medical Imaging, 4*, 87 94.

[6] Do, N. M. Z., & Ramos, B. V. (2014). Simulating external compressions of the breast with the surface Evolver. *Journal of Physics: Conference Series* vol 490 (IOPPublishing) p.012146.

[7] Do, N. M. Z., & Ramos, B. V. (2015). Simulating CC and MLO compressions with the surface evolver. *Journal of Physics: Conference Series* vol 574 (IOPPublishing) p. 01207.

[8] Fabris, A.E., Do, N. M. Z., & Ramos, B. V. (2015). A software tool based on the surface evolver for precise location of tumours as a preoperative procedure to partial mastectomy. *Journal of Physics: Conference Series* .

[9] Brakke, K. (2013). The surface evolver. Susquehanna University. Retrieved July 2013, from the website http://www.susqu.edu/brakke/evolver/evolver.html .

[10] Fabris, A. E., Bandeira, I., & Ramos, B. V. (2015). Human-computer interfaces applied to numerical solution of the Plateau problem. *Journal of Physics: Conference Series*.

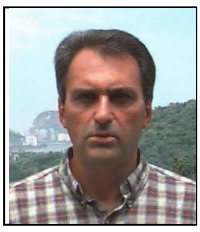
[11] Hornof, A., Cavender, A., & Hoselton, R. (2004). Eyedraw: A system for drawing pictures with eye movements. *Proceedings of the Conference on Computers and Accessibility,* 86 93.

[12] Perini, E., Soria, S., Prati, A., & Cucchiara, R. (2006). FaceMouse: A human-computer Interface for tetraplegic people, *Proceedings of the 2006 International Conference on Computer Vision in Human-Computer Interaction* ï r x

[13] Chau, T., Memarian, N., Leung, B., Treherne, D., Hobbs, D., Worthington-Eyre, B., Lamont, A., Pla-Mobarak, M. (2012). Home-based computer vision access technologies for individuals with severe motor impairments. *Handbook of Ambient Assisted Living,* 581 597.

[14] Jian, Y., & Jin, J. (2012). An interactive interface between human and computer based on pattern and speech recognition, *Proceedings of the 2012 International Conference on Systems and Informatics* (pp 505 509).

Ivana Soares Bandeira has „ *f* … Š ‡ degree in mathematics at Amazonas Federal University, Brazil, in 2009. She completed her master course in mathematics from ï Amazonas Federal University, Brazil, in 2012. At the moment, doctorate course in progress at the University of São Paulo (IME-USP). Her recent publications are: "A human-computer interface and an analysis on the drawing of curves with a face tracker mouse," presented at the International Conference on Universal Access in Human-Computer Interaction, Springer International Publishing, 2016; "Human-computer interfaces applied to numerical solution of the plateau problem," *Journal of Physics, Conference Series*, 2015.



Antonio Elias Fabris works as a full professor of computer science and applied mathematics at the University of São Paulo (IME-USP), Brazil. His main research interests are in computer graphics, applied discrete geometry and software development.

He obtained a bachelor's degree in applied mathematics in 1978 and a M.Sc. in applied mathematics in 1986 both from the University of São Paulo. In 1986 he took a position at University of São Paulo (IME-USP). He holds a Ph.D. in computer science from the University of East Anglia (UK, 1995), a post-doctoral degree in computer science from IMPA-CNPq

(Vision and Graphics Laboratory, 1998). He has been a visiting professor at the School of Computing Sciences (UEA) and the Visgraf: Vision and Graphics Laboratory (IMPA).

In 1996, Prof. Fabris formed the CGCAP, Computer Graphics and Applied Computational Geometry Group, at the IME-USP. CGCAP is the second Brazilian lab to have its research published and presented at ACM-SIGGRAPH (Los Angeles, Aug 1997).

Valério Ramos Batista graduated in computer engineering at the Technological Institute of Aeronautics in SJ Campos, Brazil, Dec 1993. He concluded his master's in mathematics at the University of São Paulo, Brazil, Sep 1996, and his PhD in mathematics at the University of Bonn, Germany, Jul 2000. Nowadays he works as a full professor in computer modelling at the Federal University of ABC, St André, Brazil.

Some recent publications as a co-author include: "LBP operators on curvelet coefficients as an algorithm to describe texture in breast cancer tissues," Expert Systems with Applications, 2016; "A software tool based on the surface evolver for precise location of tumours as a preoperative procedure to partial mastectomy," *Journal of Physics: Conference Series*, 2015; "Programming plantation lines on driverless tractors," Revista SODEBRAS, 2014. His main research areas are image processing and computer modelling applied to medicine, agriculture and human-computer interfaces.

Prof. Ramos Batista was awarded the scholarship in research productivity by the Brazi- lian National Council for Scientific and Technological Development, from Mar 2008 to Feb 2011.