Tools for Visualizing Cuts in Electrical Engineering Education

Alex Stockrahm[1], Jari Kangas[2], and P. Robert Kotiuga[3]

[1]Electrical and Computer Engineering, Boston University, Boston MA, USA Email: adstock@bu.edu
[2] Electronics and Communications Engineering, Tampere University of Technology, FINLAND Email: jari.kangas@tut.fi
[3]Electrical and Computer Engineering, Boston University, Boston MA, USA Email: prk@bu.edu

We study visualization of electromagnetic (EM) phenomena from an educational viewpoint. We exploit software and 3D printing for visualization and exemplify their usage to enhance student learning. Our focus is on so called cuts which highlight important topological information in EM problems. We also discuss conceptual and historical advances on topological features of EM problems.

***Index Terms*—Computer aided instruction, Courseware, Electrical engineering education, Open source software**

I. Introduction

Visualization is an important form in gaining comprehension, e.g. [1] lists three categories: visual, auditory, and kinesthetic. Visualization of electromagnetic (EM) phenomena includes controversies and challenges. Sometimes illustrations may appear as unintuitive, yet they are correct. And, sometimes illustrations may feed incorrect understanding. A particular challenge is to depict features of 3D EM problems via 2D illustrations.

An integral part of EM problems is topological information. Students often benefit from visualizing such information to gain a better comprehension of subject matter. An example of topological information is the so called cuts which are inherent in scalar potential formulations as well as in the definition of global quantities; inductance, coupled problems, and force–free magnetic fields, to name a few [2].

Nowadays, software is commonly utilized in Electrical Engineering education to enhance student learning. In particular, software may be used as an aid to visualization. In addition, students are likely familiar with many other forms of visual material, say, artistic illustrations constructed using modern CAD tools and 3D–printing (see e.g. [3]) and science programs' video footages (of solar coronas). These remarks motivated us to study how to leverage tools currently available for visualization and enhancing student learning: algorithms, open–source software, and 3D printing.

We leverage open source software and 3D printing to highlight topological properties of EM phenomena. We describe a software that allows visualization of cuts in unknotted and knotted geometries and supports 3D printing. We provide examples of such tools' use in EE education (incl. cases that may appear unintuitive). We also include remarks on different definitions of cuts and on historical advances in topological structures, which reveal interesting controversies on learning and visualization of EM phenomena.

II. Cuts and electrical engineering education

Typically students first encounter cuts when learning about inductance L and linkage between two current-carrying loops. Perhaps it is not stressed to them that one is dealing with cuts, but the notion of L involves treating a family of surfaces (representative chains of generators of the same relative homology group) through which same magnetic flux flows; see 2a.

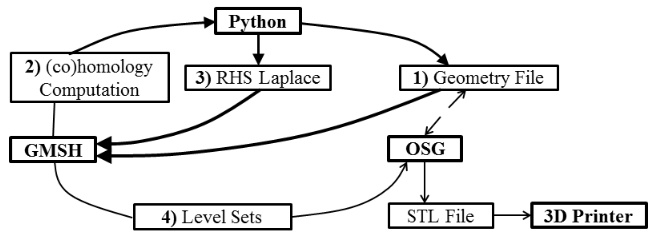


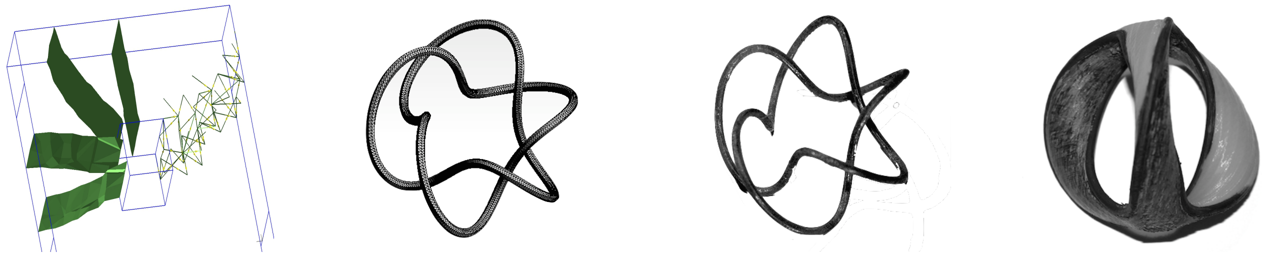
Fig. 1. Flowchart of software tools to visualize knots and cuts.

Inductance can be defined using magnetostatics where magnetic scalar potential is used in nonconducting regions; see e.g. [2]. It is essential that the nonconducting region be "cut" to have well posed problem for the scalar potential.

Another example is the analysis of a twisted pair of wires and of the EMF induced in such a loop, where visualizing "twisting" of the related oriented surfaces aids in comprehension. A third example is the knotting of current paths which appears e.g. in solutions of the near force–free magnetic fields problem[4] and in solar coronas. Thus, to allow students to see the multifacetedness of EM phenomena our code also supports knotted geometries. In all of the above examples, families of cuts highlight profound topological information whose visualization we aim to enhance.

The need for cuts became apparent in computational electromagnetism (CEM) in the 1980s with 3D problems. Since then many papers have contributed to the definition and existence of cuts and the development of related algorithms (see e.g. [2] and references therein). To highlight features of cuts, videos were used in 1990s to illustrate several key properties[5]. At that time, making the videos required special software and hardware. Presently, open source software packages allow one to start from a much higher level. This shift was one key motivation for our study. An outline of our software is given in Fig. 1; details of it are explained in section III.

For finite-element (FE) algorithms, the way cuts look is irrelevant. However, to a user certain cuts may appear more



(a) Five representations of the same cut. Left: four levelsets. Right: a thick cut.

(b) Display of (5, 2) torus knot generated in Python.

(c) 3D print of (5, 2) knot geometry.

(d) 3D print of cut for a trefoil knot.

Fig. 2. Visualization tools for knotted geometries and cuts

illustrative than others. In FE computations, cuts are typically realized via representative (co)cycles[6] such as the so called thick cut; see Fig. 2a. Moreover, some users may benefit from studying a family of cuts (as in the case of L) to highlight the underlying topological and EM features. In more technical terms, there is a distinction between so called thick cuts and cuts as embedded manifolds[7]. The latter can be found via levelsets of the magnetic scalar potential[2]. Let us stress that topological information should not be considered only as visual; this viewpoint is supported e.g. by contributions of blind topologists on related mathematical structures[8].

III. Software for visualizing cuts

A flowchart of our code is shown in Fig. 1. We used the interpreted programming language Python to write our own routines, items 1)-4) Fig. 1. As external software we used GMSH[9] and OpenSceneGraph (OSG)[10]. GMSH provides us with geometric viewer, a mesh generator, and builtin topological processor[6].

OSG is built upon C++ and OpenGL, and allows users to manipulate geometries to prepare surfaces for 3D printing as well as knot geometries for FEA. We adhere to existing data formats in input/output—Standard Tessellation Language (STL) and text files in GMSH native formats.

To create geometry for the domain Ω of interest, the user may use GMSH or our routine to create parameterized (p, q) torus knot geometry, item 1). The triangulated tubular neighborhood of a knot is constructed by revolving a radial vector about the rotation axis given by the parameterized knot using a 4D transformations class[11] to generate the nodes on the tube's surface. Other simplices are then constructed with an orientation imposed for each face of the knot via an ordering of the nodes supported by GMSH and OSG. The resulting structure is exported as a GMSH geo-file. An example of a (5, 2) torus knot geometry generated and plotted is shown in Fig. 2b alongside a 3D print of the same geometry, Fig. 2c.

Our software allows users to define Ω as a knot complement which is truncated by a bounding sphere. The resulting file may then be read into GMSH, where a 3D frontal meshing algorithm is employed to mesh Ω. Then in GMSH (co)homology computations are performed to find representatives of the 2nd relative homology spaces of the knot complement, namely $H_2(\Omega, \partial\Omega)$, which is the generator of the Seifert surface, see Fig. 2d. Alternatively, representatives of 1st absolute cohomology space $H^1(\Omega)$ can be computed.

To compute for levelsets item 3), we solve the Laplace equation using the Galerkin technique and linear tetrahedral elements. Consequently, the system matrix is a standard stiffness matrix and the right hand side is set using cuts—representatives of $H_2(\Omega, \partial\Omega)$, or $H^1(\Omega)$[2].

The solution needs to be postprocessed to find surfaces of families of cuts—levelsets. This can be done as part of postprocessing in Python or in GMSH. For 3D printing the surfaces need to be thickened. The output mesh can be processed utilizing OSG where each vertex is extruded along its normal. Alternatively, user may use a CAD software to perform the extrusion. In either case, the result is written as an STL file suitable for 3D printing.

IV. Summary and conclusions

We presented tools for visualizing cuts and outlined related educational viewpoints. In the full paper we aim to include additional educational examples (incl. cases where results, though unintuitive, are correct) and also elaborate on conceptual and historical advances related to comprehending topological information and EM phenomena.

References

[1] R. M. Felder and L. K. Silverman, "Learning and teaching styles in engineering education," *Engineering education*, vol. 78, no. 7, pp. 674–681, 1988.

[2] P. W. Gross and P. R. Kotiuga, *Electromagnetic Theory and Computation*. Cambridge University Press, 2004.

[3] B. Grossman, "Bathsheba sculpture," http://www.bathsheba.com/.

[4] P. R. Kotiuga, "Topology-based inequalities and inverse problems for near force-free magnetic fields," *Magnetics, IEEE Transactions on*, vol. 40, no. 2, pp. 1108–1111, 2004.

[5] http://scv.bu.edu/visualization/gallery/knots/knots.html.

[6] M. Pellikka, S. Suuriniemi, L. Kettunen, and C. Geuzaine, "Homology and cohomology computation in finite element modeling," *SIAM Journal on Scientific Computing*, vol. 35, no. 5, pp. B1195–B1214, 2013.

[7] P. Dłotko and R. Specogna, "Cohomology in 3d magneto-quasistatics modeling," *Commun. Comput. Phys*, vol. 14, no. 1, pp. 48–76, 2013.

[8] P. R. Kotiuga, "On the topological characterization of near force-free magnetic fields, and the work of late-onset visually-impaired topologists," accepted for publication in AIMS Journals.

[9] C. Geuzaine and J.-F. Remacle, "Gmsh: A 3-d finite element mesh generator with built-in pre-and post-processing facilities," *IJNME*, vol. 79, no. 11, pp. 1309–1331, 2009.

[10] http://www.openscenegraph.org.

[11] http://www.lfd.uci.edu/~gohlke/code/transformations.py.html.