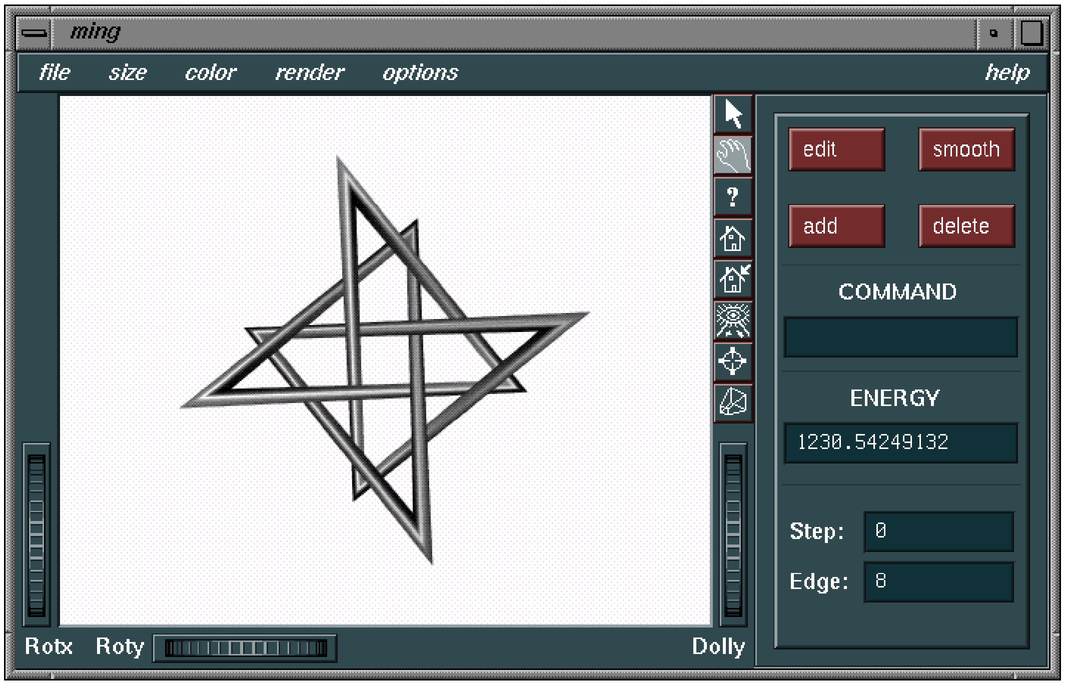
Ming – An MD Knot Energy Minimizing Program

Ying-Qing Wu

§1. The Program

Ming is a computer program used to find local minima of MD energy for knots. The program is available via anonymous ftp at ftp.math.uiowa.edu, in the directory of wu/ming. WWW users can go to *http://www.math.uiowa.edu/˜wu/min/ming.html* and follow the links to get it. Remember that you need to change the mode of the file by typing “chmod 700 ming” to make it executable. Figure 1 shows the outlook of the program.



ENERGY: 1230.54249132

Step: 0

Edge: 8

Figure 1: A picture of *ming*

Ming is written in C++, X-Motif and Open Inventor. It can handle knots with up to 1000 vertices, and can be used as a tool to draw and edit a knot, visualize it in different forms, minimize its MD energy, and find minimal edge presentation of knots. These will be discussed in detail in the following sections. Since *Ming* uses Silicon Graphics’ Open Inventor to draw the knot pictures, it can run only on SGI machines (or at least so in our building.)

The history. This program emerged from a talk given by Jonathan Simon on our topology seminar in Spring 95. He pointed out that MD energy is not a smooth function on the configuration space of a polygonal knot. As such, earlier energy minimizing programs, like *ked* of Kenny Hunt, use random perturbation methods. While this works for small knots, its performance is very poor for knots with large edge number. The first version of *ming*, called *min*, was written in the belief that the “gradient flow” method should work despite the fact that the gradient function is discontinuous on the configuration space. This turned out to be true. The program unknotted a 122 edge twisted Freedman knot within a few hours, while the earlier programs got stuck after running for several days.

Min did not have graphic output. It only showed the energy of the knot. The graphic features were added to the program during the winter vacation of 1995, and the name changed to *ming*, meaning *min* with *g*raphics. The knot editor is added in Summer of 1996, making it possible to enter the knot by clicking the mouse buttons.

Please send any comments or bugs you found to me at *wu@math.uiowa.edu.*

Acknowledgement. I would like to thank Jonathan Simon for his continuous support and many helpful conversations from the begining of this project, and thank Kenny Hunt for sending me his code for *ked*, which had been very helpful when I started writing Open Inventor code.

§2. Knot Editor

In the directory of *ming*, type the command “ming” to start the program. To draw a new knot, choose *New* from the *File* menu, or use the keyboard shortcut: Press the “alt” key and type “n”.

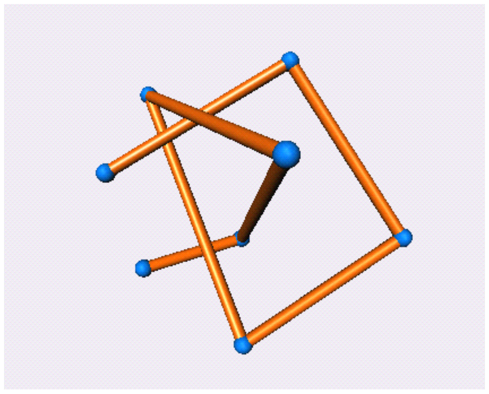


Figure 2: Drawing knot on *ming*

Click mouse button 1 to get a new vertex for the knot, and click button 2 to delete the last vertex. To move the new vertex, do not release button 1 after pressing it, and move the mouse around to move the vertex on the xy-plane (the view plane). To move the point along the z-axis, press button 1 and button 2 simultaneously, then move the mouse up or down. Figure 2 shows the picture of a knot in drawing. Knot can be opened or saved to a knot file using the *File* menu.

After making a knot, the program enters ‘edit’ mode. Click the **edit** button on the control panel to exit/enter edit mode.

In edit mode, click the ‘arrow’ box on the right hand side of the drawing area to make picking possible. Click button 1 on a vertex or an edge to highlight it. Button 2 is used to delete a highlighted vertex, or add a vertex to a highlighted edge. Moving a highlighted vertex around in the same way as when drawing a new knot. One can also use the wheels to turn the knot around, then move the vertex in the desired direction.

§3. VISUALIZATION

The drawing window, the wheels, and the little pictured boxes on the right edge of the drawing area form the "examiner viewer" of Open Inventor. This viewer has some nice built-in functions, which is the main reason we used Open Inventor instead of Open GL.

Click the "eye" box will put the knot in a good view position. Click the "hand" box to enable moving the knot. One can then drag the knot around using the mouse buttons. Click the "?" box to see detailed help information about this viewer.

Ming can visualize 3-D knot picture in different forms. The thickness of the knot can be changed using the *Size* menu items, or the command **e N**. In non-edit mode one can use the *Render* menu to choose drawing the knot as arc segments, tubes, or cylinder-balls. Use *Color* menu to specify the color schedule, or bring up color editors for knot and background, respectively. See §6 and §7 for more details about commands and menus.

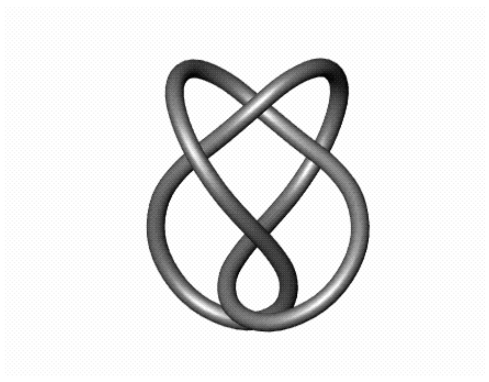


Figure 3: A smooth Figure 8 knot

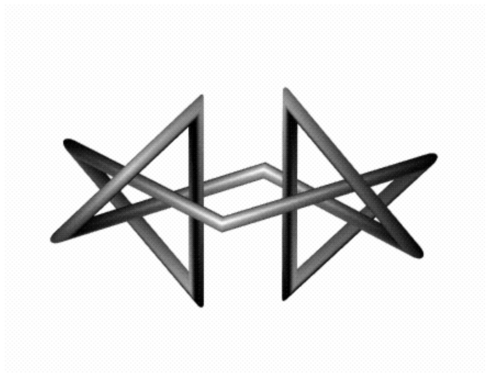


Figure 4: A polygonal square knot

One can draw a smooth knot by clicking the **smooth** button on the control panel. See Figure 3 and Figure 4 for pictures of a smooth Figure 8 knot and a polygonal square knot. The smooth mode actually draw a knot with 150 edges. To make the knot look smoother, enter the command **s N**, where N is the number of edges of the smooth knot, which should be at most 1000. Be aware that the smooth knot drawn is an approximation of the polygonal knot, so in some cases the knot type could have been changed, i.e. the knot you see may not be the knot you have behind the screen. If this happens, increase the number of edges for the original polygonal knot, run *ming* for a while, then redraw the smooth knot.

§4. The MD Energy Minimizer

The MD energy is defined by Jon Simon for polygonal knots. A polygonal knot K consists of several edges $e_1, ..., e_n$ in the Euclidean space, which form a closed knotted loop. The ends of the edges are called the vertices of the knot. The energy of K is defined as

$$E(K) = \sum \frac{L_i L_j}{D_{ij}^2}$$

where L_i is the length of e_i, D_{ij} is the minimum distance between e_i and e_j, and the sum is taken over all non adjacent edges. We refer the readers to Simon's paper [Si] for a thorough discussion of the MD energy, its relation to the other energies, and its application to the study of DNA.

A knot can be deformed to another by an "isotopy". The problem about computation of knot energy is to find the minimal energy among all knots isotopic to a given knot. *Ming* will try to find the minimal energy by pushing the knot along the direction of its "energy gradient". Actually this method does not find the minimal energy. What it approaches are local minimals of the knot energy. Such minimals are not unique. For example, the knots 4_1.8 and 4_1.8.2 in the directory "knots" are both figure 8 knots with 8 edges, and their energy can not be reduced by *ming* or any other known programs, but their energies are very different: 228 with 304! Another example is given by an unknot with 22 edges, named *triv.min* in that directory, which aparently can not be unknotted without increasing the energy. No algorithm is known to find the absolute minimum of the energy for a given knot.

To run the energy minimizer, enter the command **run N** or **r N**. The knot is moved with certain restriction to guarantee that the knot type will not be changed. There are two methods used to flow the knot. The default is to move each vertex along its gradient direction, according to its own derivative and the restriction of minimal distances between its adjacent edges and the other edges. The second method is to move all the vertices according to a universal derivative of the gradient, and subject to the universal minimal distance restriction. The first method is the default, and runs faster in practice for many knots. The second method is more accurate when approaching a local minimum. Choose *Rough Run* from the *Options* menu to toggle between these two methods.

The commands **V F N** and **v F** act like video controls. **V F N** will record the running result to the file **F** every **N** steps. Note that the data is appended to the file, not to erase the old file. Type a single **V** will stop the recording. The command **v F** will display the video file **F**. You may try the two video files *freed.v* and **brun.v** in the "knots" directory.

§5. Handling edges

The edge number of a knot K is defined as the minimal number of edges one can use to present K. See the paper of Richard Randell [Ra] for the theory of this knot invariant. *Ming* can be used as a tool in trying to find a presentation of a given knot with few edges.

To delete N edges from K, enter the command **d N**. The program will try to find some *deleting triangle*, that is, a triangle spanned by two adjacent edges and is disjoint from the other edges. If it finds such a triangle, it replaces those two edges of the knot by the third edge of the triangle, thus decrease the number of the vertices without changing the knot type.

If this fails, one can try to delete a vertex using "Pinch Force". Toggle this method on/off by choosing *Add Pinch Force* from the option menu. With this method, the program will add some force to the gradient on the two ends of an edge, and run the energy minimizer. One can see that one of the edges is shrinking while the program is running. In many cases, when the edge becomes small, a deleting triangle appears, so the program will be able to delete a vertex.

The program has successfully found knot presentations with best known edge numbers for many knots. However, there is no guarantee that the minimal number of edges found by the program is the edge number of the knot. Right now there is no known algothrim for finding the edge number of a knot, and the edge number of knots are known only for a few knots. See Randell's paper for more details.

To add vertices to the knot, simply type the command **a N**.

One can delete vertices manually using the knot editor. To avoid changing the knot type, one may want to see the triangle before deleting it. To show all triangles, enter the command **tr**. (This is the only command which is sensitive to more than one letter.) Enter **tr** again will delete them. To show a specific triangle, click at a vertex to highlight it, press button 1 and then click button 3. To delete a triangle from the picture, click button 1 on that triangle.

The edge deleting part is implemented only in the generic case. If the plane of the triangle passes through more than 3 vertices, it will not be considered as a deleting triangle, even if it does not intersect other edges. In particular, the program will not be able to detect any deleting triangles if the knot is a planar unknot.

§6. The Commands

The following is a list of all the available commands. Most of them can be obtained be either clicking on the command buttons (smooth, add, delete) or choosing from the menus. All but the **tr** ommands are sensitive only to the first letter. Thus, for example, **run: 10** and **r 10** will have the same effect: Both run the energy reducing process 10 times. In bellow, N will denote a number, which could be integer or real number.

a N: Add N vertices to the knot K. While N is greater than the number of vertices of K, the program will add one vertex for each edge. Otherwise, it adds vertices to the longest edges.

A N: Set the level of Antialiasing. N is between 1 and 20. If N=1, antialiasing is off. Higher antialiasing level will make the picture look smoother, but the rendering will be very slow if antialiasing is on. You may want to turn it on when using "snapshot" to record knot pictures.

c N: Set the color perior. Try to choosc c -I/usr/inCC -I/usr/include -L/usr/lib -lm -le periodic from *Color* menu, and enter the command **c 3** to see how the color is changed.

d N:	Delete N vertices from the knot. The program will delete a vertex only if it does not change the knot type. When no vertices can be deleted in the obvious way, and if *add pinch force* is on (default), it will pinch one of the edges, and then try deleting again.
e N:	Set the edge radius. N can be real number.
O F:	Open the file “F”. An alternative to the *Open file* dialog.
p N:	Set the pinch edge. If pinch force is turned on, the program will try to pinch this edge to a point in order to remove a vertex. Set **p -1** to choose the edge having greatest angles to its neighbors (default).
P N:	Set Pixel per inch when saving the knot picture as a postscript file. Default is 150.
r N:	Run the energy minimizer N times.
s:	Toggle between drawing smooth/polygonal knots. Same as the **smooth** button.
s N:	Drawing smooth knots. N determines the number of vertices on the smooth knots, default = 150. WARNING: When drawing a smoothing knot, the program replaces arcs with curves, so it is possible that the knot type shown could be different from what it actually is.
S F:	Save the knot to file “F”. An alternative to the *Save file* dialog.
t N:	If *Render* menu items is set to *tube*, this command will set the number of sides for the tube. For example, **t 3** will set the cross section of the tube as triangle. If N is larger, the tube will look smoother, but it will slow down the render process. The default value is 8, and the maximum is 32.
tr:	Show/hide all triangles with two edges on the knot. Used to find deleting triangles.
V F N:	Save output to file “F” every N steps.
v F:	Continuously show knots in file “F”. Acting like showing a video.

§7. MENUS

1. FILE MENU: It contains *Open*, *Save*, *Save PS*, *New* and *Quit* items. *Save PS* will save the picture as a post script file.

2. SIZE MENU: One can choose the radius of the knot tube here. Default value is 2.

3. COLOR MENU: There are three ways to color the knot.

(i) *Periodic*: The color is changed periodically when traveling along the knot. Default period is 1, and can be changed by *Set Color Period*.

(ii) *Depth*: The color is determined by the original z-coordinate of the point.

(iii) *Single*: Use single color.

With *Choose Knot Color* one can choose several preset single colors for the knot.

The next two items will bring up color editors for the knot (with single color) and the background. When the color editor window shows up, click on the toggle buttons on the editor to bring up the color map, and move the point in the color map to choose a color.

The background color is specified only by the diffuse color on the color editor.

The last item prints the command **color_period** for setting color period. See §6 for more details.

4. RENDER MENU: Rendering by line is the fastest. Rendering by tube is the default, and is probably the best. When the vertex number is small, rendering by cylinder-ball will show round vertices, but it is slow when vertex number is large. The 4th menu item prints the command for setting the tube sides. Draw a smooth knot, and try the commands **t 1**, **t 2**, **t 3** and **t 10** to see the difference.

5. OPTIONS MENU:

(i) The first item toggles antialiasing on/off. When it is turned on, it set antialiasing level to 10. See §6 for more details.

(ii) The second item toggles *Rough Run* on/off. When rough run is on, each vertex of the knot is moved according to its own gradient. This method approaches the minimal energy faster for many knots, but it is not accurate.

(iii) *Add Pinch Force*. This item is important when one wants to find the minimum number of edges to represent a knot. If this is off, the **delete** command will only try to delete vertices without changing the other vertices. If *Add Pinch Force* is on, it add some extra force on certain edges, trying to pinch it to a point so that certain vertex can be removed.

The other two items prints commands for setting antialiasing level and pinch edge.

6. HELP MENU: It contains 4 items: *README*, *commands*, *menus* and *edit*. Each will print some help messages.

References

[Ra] R. Randell, Invariants of piecewise-linear knots, *preprint*.

[Si] J. Simon, Energy functions for knots: beginning to predict physical behavior, *preprint*.

Department of Mathematics, University of Iowa, Iowa City, IA 52242
E-mail: wu@math.uiowa.edu