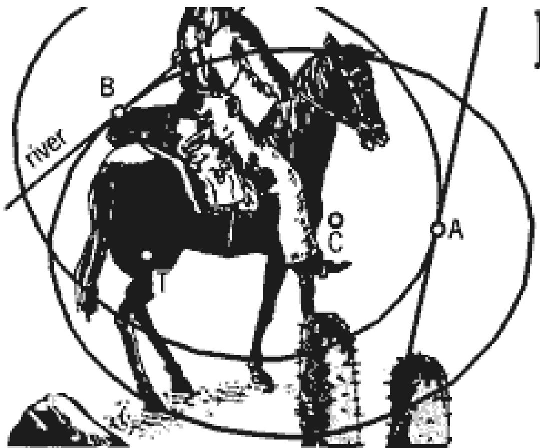


Problem solving and proof in the age of Dynamic Geometry

Daniel Scher

Since their introduction in the early 1990's, dynamic geometry software packages such as The Geometer's Sketchpad (Jackiw, 1995) have redefined the exploration process. No longer need students view a static picture of a triangle and its three altitudes. Now they can gather a host of data by dragging a triangle vertex and observing that, for seemingly every triangle on screen, the altitudes remain dutifully concurrent (figure 1).

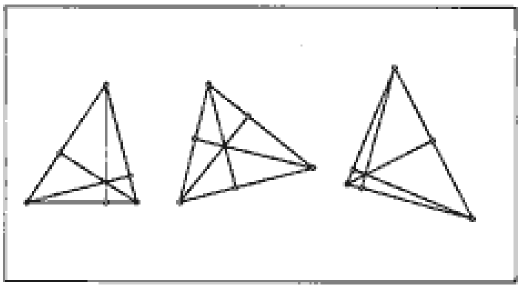


Figure 1

But saying "seemingly every" falls short of conviction: mathematics requires proof. And here dynamic geometry often fades into the background. Having used the software to convince themselves of a theorem's plausibility, mathematics classes turn to the traditional paper-and-pencil medium to ponder a proof. In the process, they create a boundary between the computer and deductive reasoning. Must this be the case?

Anecdotal evidence suggests not. Software such as Sketchpad is sufficiently rich that its capabilities reveal themselves slowly over repeated use. Newcomers to Sketchpad may view the software primarily as a demonstration tool, suitable for reproducing classic results. With time, however, there comes an almost unconscious broadening of the software's role in one's work. Users begin to notice a change in their style of experimentation and reasoning. The boundary between deductive reasoning and dynamic geometry becomes blurred; the software finds its way into the proof process.

Pinpointing this change in work style is a sticky affair. The role of dynamic geometry as a partner in problem solving and proof tends to remain elusive. In this article, I will attempt to expose this partnership by offering concrete examples, showing the ways in which dynamic geometry can provide not only *data* to feed a conjecture, but *tools* to jump-start ideas and feed a proof. Two problems will illustrate these software roles:

(1) the modelling and investigation of a mechanical linkage,

(2) a geometric approach to an optimisation problem.

Modelling a Linkage

Figure 2 shows a linkage (the so-called "Pantograph of Sylvester") consisting of six equal-length rods. Four rods comprise rhombus *OBAC*, and two rods form arms *BD* and *CE*. An actual model of the linkage contains pencils at points *D* and *E*. Point *O* is stationary, but hinges at *B*, *A*, and *C* allow all other parts to move. The angles *DBA* and *ACE* are kept equal and at a fixed measurement by the two unlabelled rods. If you were to draw a picture using the pencil at point *D*, then what would point *E* trace? Why?

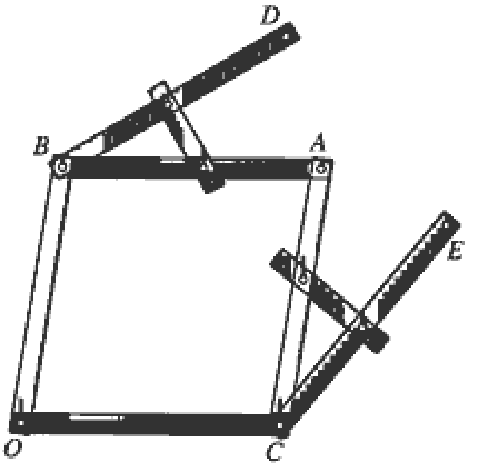


Figure 2

Setting figure 2 into motion can be accomplished by building a physical model or constructing a dynamic geometry sketch. Both methods are fine options; the below description explores the dynamic geometry route. Before reading further, try constructing a software model yourself. You'll find that you need to apply some geometric reasoning to ensure the model behaves as specified.

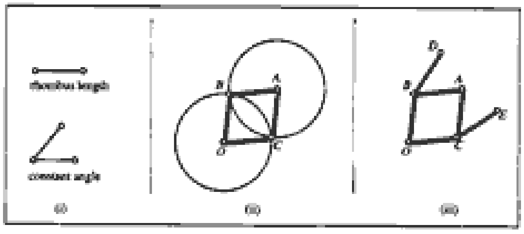


Figure 3

The Construction

The steps below, illustrated in figure 3, offer one possible method for modelling the linkage with Sketchpad. An on-line interactive Java version is available at

http://members.xoom.com/dpscher/rotator.html

(i) Draw an arbitrary segment whose length will determine the sides of rhombus *OBAC*. Draw an arbitrary angle to represent the measures of ∠DBA and ∠ACE. The length of the segment and the measure of the angle are both adjustable, so you'll be able to vary the particulars of the linkage once it's complete.

(ii) Draw two arbitrary points *O* and *A*. Construct circles, with *O* and *A* as centres, having radii equal to the segment you drew in (i). Move the points or adjust the radius if necessary to make the two circles intersect (at points *B* and *C*). Draw segments connecting *O*, *B*, *A*, and *C* to form a rhombus. Hide the two circles.

(iii) Rotate point *A* about point *B* anti-clockwise by the angle you drew in (i). You'll need to use the 'Mark Angle' command to do so. Then rotate point *A* about point *C* by this same angle, only clockwise. Draw segments *BD* and *CE*. The linkage is complete!

To experiment with the device, put a trace on points *D* and *E*. In this particular construction, the linkage is operated by dragging point *A*. Figure 4 shows the result when point *D* draws the letter 'G': point *E* traces an exact, rotated copy of the letter. Increasing the measure of ∠*DBA* and ∠*ACE* still yields duplicate images, only with a larger rotation.

What to Prove?

The construction process above describes how I began my own investigation of the linkage. Having then used Sketchpad to collect experimental data, I considered putting the software aside to begin a paper-and-pencil proof. What kind of evidence, if verifiable, could help to explain why the linkage drew rotated copies? I did not have an immediate answer to this question. Appealing to figure 4 for inspiration didn't help because without the working model on hand it was difficult to visualise its movement. Perhaps the Sketchpad model could tell me more. The linkage contained many moving parts, but it was the motion of points *D* and *E* that mattered most. Could I find a way to investigate their movement without contending simultaneously with the movements of points *B*, *A*, and *C*? Draw segments connecting stationary point *O* to points *D* and *E* (figure 5). Now when operating the linkage, imagine that it consists of only two arms, *OD* and *OE*. Observing their movement suggests some noteworthy conjectures:

(i) the lengths of arms *OD* and *OE* seem to remain equal to each other;

(ii) the angle *DOE* seems to stay constant.

By assuming these statements were true, a thought experiment helped me to explain the linkage's behaviour. Imagine viewing arm *OD* as it draws a picture. You'd see it continually change length and direction. What about arm *OE*? Its length would always be equal to *OD* (by (i)) and its direction would always adjust to remain at fixed angle *DOE* with respect to *OD* (by (ii)). So if you were to videotape arm *OD*'s movement and compare it to a video of arm *OE*, you'd see identical images by tilting your head while viewing *OE*.

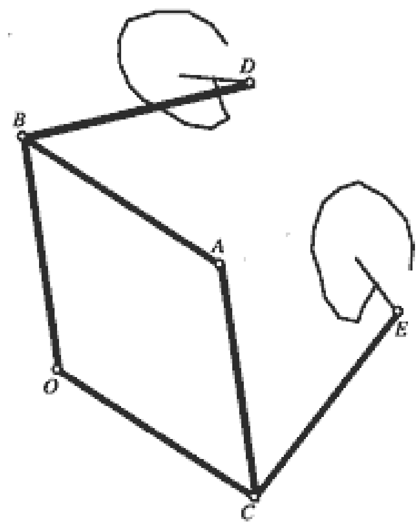


Figure 4

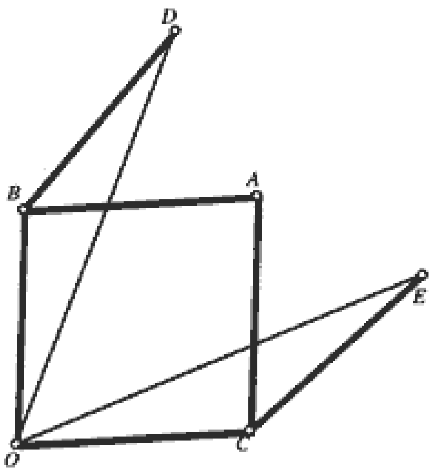


Figure 5

Notice how the reasoning process here was aided by the software. The language of the thought experiment – stretchable segments that remain at a fixed angle as they rotate – is supported by the vocabulary of a software program that makes it natural to talk about geometric objects stretching, shrinking, and moving.

How to Prove it

With an analysis in hand, it remained to justify the two conjectures from above. To prove $OD = OE$, observe in figure 5 that sides OB, BD, OC, and CE of ΔOBD and ΔOCE are all equal by construction. As rhombus angles OBA and OCA are equal and $\angle DBA$ was constructed equal to $\angle ACE$,

$$\angle OBD = \angle OBA + \angle DBA = \angle OCA + \angle ACE = \angle OCE.$$

Thus by SAS, $\Delta OBD \cong \Delta OCE$, implying $OD = OE$.

It remained to show that $\angle DOE$ stayed constant. The symmetry of figure 5 made it almost irresistible to add a line of symmetry through points O and A. Figure 6 includes that line, along with segments DE, AD, and AE. From above, DODE is isosceles, and by symmetry, OF is its altitude.

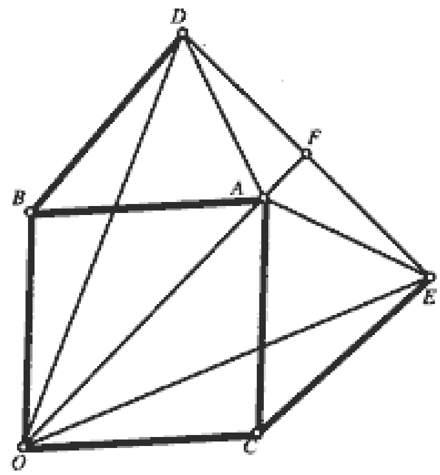


Figure 6

Having added some new segments to the construction, I turned again to the Sketchpad model to see how they would behave. Moving point A, I noticed that the shape and size of ΔBAD

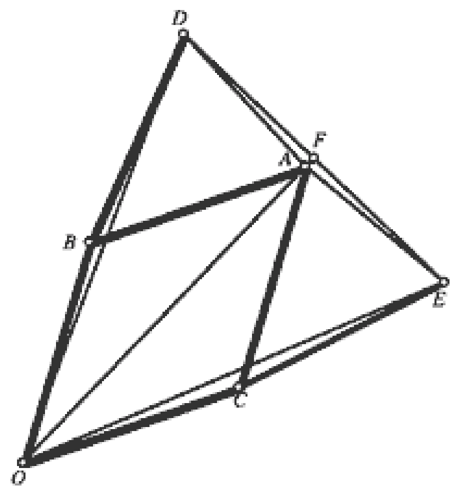


Figure 7

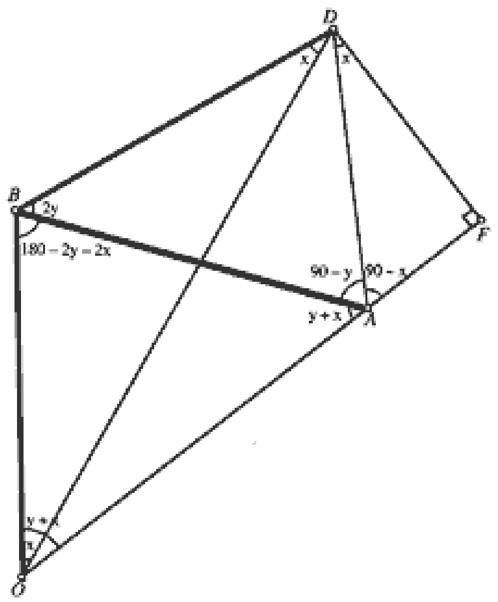


Figure 8

did not change. This was not especially surprising as its two sides, *BD* and *BA*, and their included angle, ∠*DBA*, were constructed to remain fixed.

Triangle *BAD*'s rigidity did not achieve any significance until I experimented with several "extreme" cases, moving point A towards points O and F. In figure 7 with point A near F, angle ∠*BDA* appears equal to – *ODE*. The two angles, in fact, coincide when point *A* lands on top of point *F*. This unexpected equality suggested a line of proof. The measure of ∠*BDA* is a constant for all locations of point *A* as it's a part of rigid Δ*BAD*. If I could show that ∠*BDA* = ∠*ODE* for all locations of A, then this would show that ∠*ODE* is constant, too. With base angles *ODE* and *OED* of isosceles Δ*ODE* constant, so must its third angle, ∠*DOE*, be invariant.

The steps below explain how I made liberal use of figure 8's three isosceles triangles to assign measurements to its angles:

(i) Let ∠*ADF* = x. Since Δ*ADF* is a right triangle, ∠*DAF* = 90 – *x*.
(ii) Let ∠*DBA* = 2*y*. Since Δ*DBA* is isosceles, ∠*DAB* = 90 – *y* (why is it easier to let ∠*DBA* = 2*y* rather than just *y*?).
(iii) Since the three angles meeting at point *A* total 180°, ∠*BAO* = *y* + *x*.
(iv) Since Δ*BAO* is isosceles, ∠*BOA* = *y* + *x* and ∠OBA = 180 – 2*y* – 2*x*.
(v) Since Δ*OBD* is isosceles, ∠*BOD* = ∠*BDO* = *x*.
(vi) ∠*BDA* = *x* + ∠*ODA* = ∠*ODF* (= ∠*ODE*)

Step (vi) proves ∠*BDA* = ∠*ODE*, and thus, as explained above, ∠*DOE* is constant. The proof is complete! As an added bonus, figure 8 gives *m*∠DOE = 2(*m*∠*DOF*) = 2*y* = *m*∠*DBA*. Therefore the amount by which the linkage rotates images is equal to the measure of ∠*DBA* and ∠*ACE*.

Discussion

At minimum, my Sketchpad use above led me to discover that the linkage drew rotated images. But the contributions of the software were, in fact, more substantial:

- I entered the problem unaware of the geometric behaviour governing rotations. With Sketchpad, I could analyse the movement of arms *OD* and *OE* in search of possible clues. These observations jump-started my thought experiment, explaining why equal arm lengths separated by a fixed angle should produce rotated images. How easy, though, to forget the role of the software in this process. A typical written proof might take rotation knowledge as a given and skip straight to the nuts-and-bolts calculations in figure 8.

- Deriving the angle measurements in figure 8 did require me to move away from Sketchpad to pencil-and-paper. Note, however, that the motivation for these calculations – demonstrating that $\angle BDA = \angle ODE$ – came from the intermediate results of my Sketchpad experiment (sub-conjectures), not merely from the main rotation conjecture.

- Having finished the proof, I retraced my steps and examined the purpose of each linkage arm. Soon, I started to wonder: with Sketchpad's formidable tool base at my disposal, did a virtual rotator linkage need to contain so many parts? No. With a little effort, I was able to construct a Sketchpad rotator consisting solely of arms *OD* and *OE*. In effect, the software put my understanding of the proof to the test. Given the chance to design a model freed from the constraints of physical parts, could I extract the key aspects of the proof and use them to build a simpler device?

The Burning Tent Problem

The rotator described above could be built either as a physical linkage or a virtual Sketchpad model. In the optimisation problems described below, Sketchpad again serves as a modelling tool. Here, however, no simple physical counterpart mimics Sketchpad's behaviour exactly. As the analysis again relies on dynamic imagery, I urge the reader to follow along by visiting

http://members.xoom.com/dpscher/river.html

and interacting with the on-line Java animations.

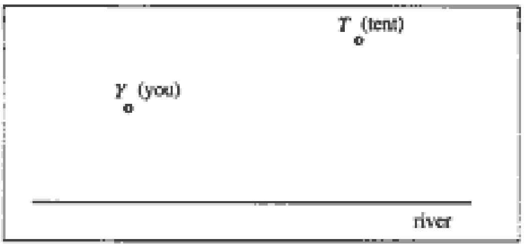


Figure 9

The so-called Burning Tent Problem states:

> You're on a camping trip. While walking back from a hike, you see that your tent is on fire. Luckily, you're holding a bucket and you're near a river. Where along the river should you fill the bucket with water to minimise the total distance to get back to your tent, *T*? (fig. 9)

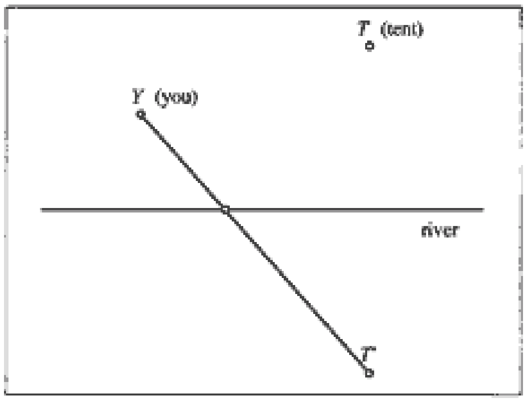


Figure 10

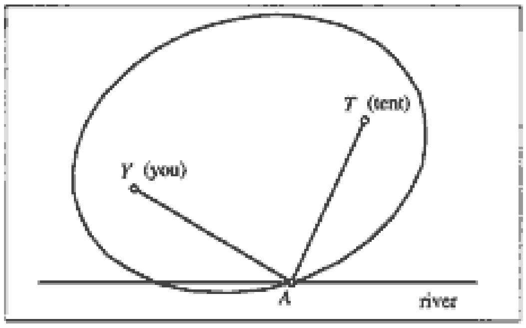


Figure 11

Typically, this problem is solved by reflecting point *T* across the river to form *T'* (fig. 10). Now, any trip from *T* to the river to *Y* is equivalent to travelling from *T'* to the same river location to *Y*. Thus a straight-line path from *T'* to *Y* intersects the river at the optimal location.

Another approach found in Connected Geometry (Education Development Center, 1999) uses dynamic geometry software to solve the same problem. Figure 11 shows an ellipse constructed with foci at *Y* and *T* passing through an arbitrary point *A* on the river. The ellipse represents all points whose combined distance to *Y* and *T* is equal in length to the trip from *Y* to *A* to *T*. This particular ellipse shows that *A* is not the optimal river location: those points on the river sitting inside it yield shorter running distances. The best spot for *A* can be found by dragging *A* until the ellipse is tangent to the river.

The Cowgirl Problem

Graduate students in my New York University geometry class explored both solution methods above. As follow-up homework, I assigned the Cowgirl Problem:

> A cowgirl wants to give her horse some food and water before returning to her tent. She starts at point *C* and decides to travel to the pasture first, then the river, and then back to her tent. What path should she take to minimise her riding distance? (fig. 12)

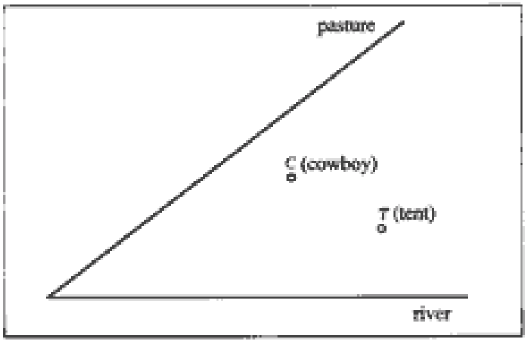


Figure 12

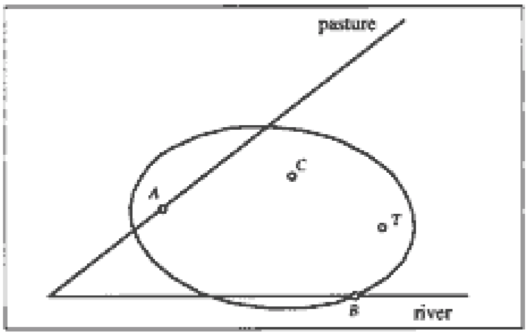


Figure 13

I anticipated that everyone would choose the reflection method to solve this problem. One student, however, surprised me by extending the dynamic geometry technique above to produce a beautiful and original ellipse proof. Here's what he did:

Place two points, *A* and *B*, at random along the pasture and river. These represent locations where the cowgirl might stop with her horse. Assume for the moment that point *A* is fixed, and consider whether *B*'s position might be improved. Construct an ellipse with focal points at *A* and *T* passing through point *B*. Figure 13 shows that *B*'s location is not optimal – the ellipse crosses the river twice. Move point *B* until the ellipse is tangent to the river (figure 14).

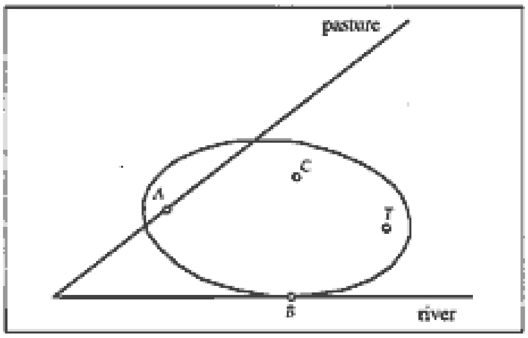


Figure 14

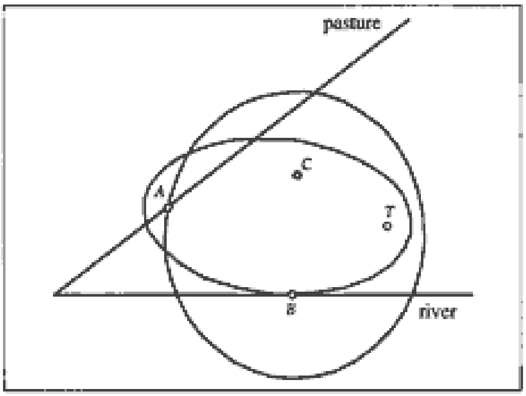


Figure 15

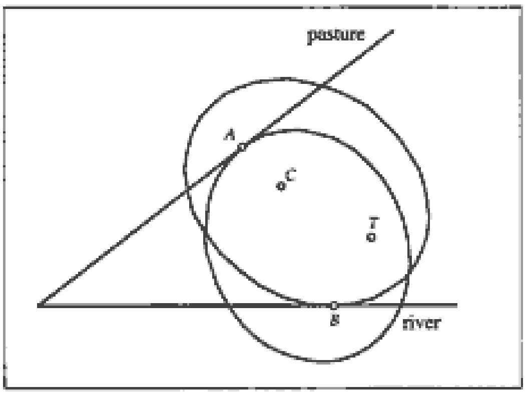


Figure 16

Now take point *B* as fixed and examine point *A*'s location. Construct an ellipse with focal points at *B* and *C* passing through *A*. Again, since the ellipse is not tangent to the pasture, point *A* needs adjusting (fig. 15). As long as either ellipse is not tangent, this process can be continued, alternating between *A* and *B*, each time shortening the overall distance to ride. In figure 16, both ellipses are tangent simultaneously, yielding the optimal spots for *A* and *B*.

Discussion

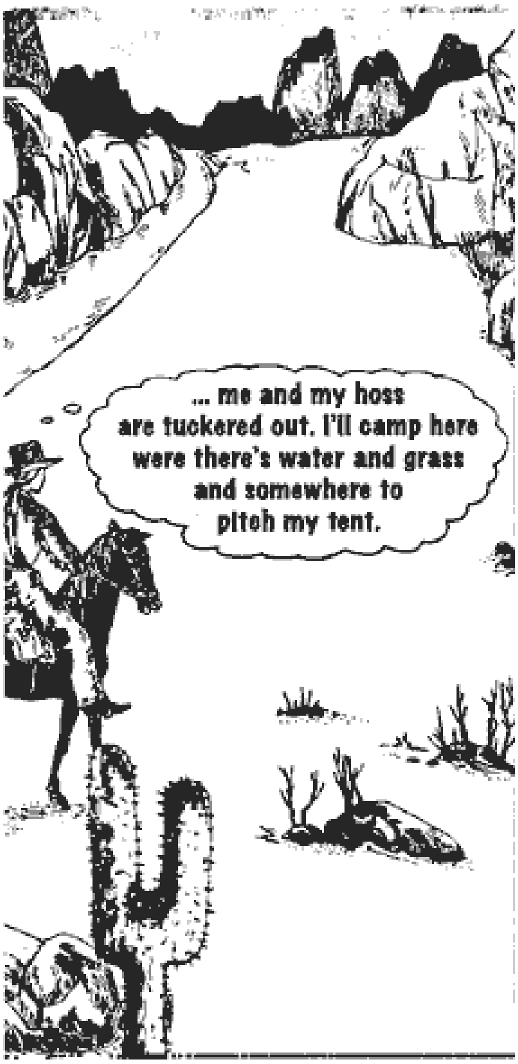
As with the rotator linkage, dynamic geometry played a pivotal role in the solution to the Cowgirl Problem. Consider the elements of my student's proof:

- When demonstrating his solution to our class, my student did not need to say, "Imagine constructing two ellipses" or "Imagine moving point *B*." The model he displayed on screen (fig. 15) was as real as a physical device, with moving parts and interdependent actions. The crucial difference: the Sketchpad model could move in ways that no physical model could. What might have remained an abstract thought experiment was converted into an interactive virtual model for the entire class to study.

- The reasoning displayed in my student's solution was quite different from the reflection technique. Pinpointing the best river location in figure 10 was a constructive process, with a method guaranteed to succeed in one shot. In contrast, my student's work with Sketchpad combined a mixture of experimentation and deduction. Picking random locations for *A* and *B* in figure 13 and then refining their placements was a "guess-and-check" technique. Yet driving the refinements was a theoretical understanding of what constituted an ideal location – two tangent ellipses.

Conclusion

As I sit now at my computer, I am typing what is the third revision of this conclusion. Does the word processor propose ideas or dictate what I write? Alas, that's still my responsibility. But its Cut and Paste tools undoubtedly have changed and contributed to my writing style. In a similar manner, dynamic geometry exerts its influence on how we reason mathematically. The responsibility of constructing an airtight proof still rests on our shoulders, but the software's collection of investigative tools can sensitise us to the power of experiments and models in contributing to the deductive process.



Daniel Scher works at the Center for Mathematical Education at EDC, Chicago.

[1]While one could draw the ellipses in figure 15 by using taut loops of string, the drawing process would need to be repeated for every movement of points A and B. As such, the smoothness and ease of operating the virtual model would be lost.

References

Education Development Center, Inc. *Connected Geometry*. Chicago, Illinois: Everyday Learning, 1999.

Jackiw, Nicholas. *The Geometer's Sketchpad*. Berkeley, Calif.: Key Curriculum Press, 1995. Software.

This article is part of the Dynamic Geometry Project funded by the National Science Foundation.