Unwrapping and Visualizing Cuneiform Tablets

Sean Eron Anderson and Marc Levoy
Stanford University

We present a semiautomatic method for unwrapping and visualizing inscribed surfaces, such as cuneiform tablets. It provides a clear visualization that can be printed on paper.

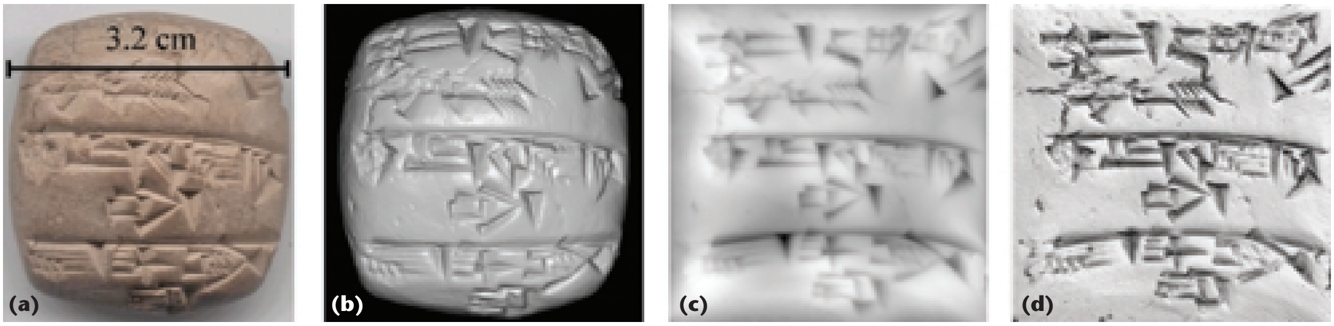
Cuneiform inscriptions, which scholars consider the first written language, were made in moist, clay tablets. People started making these tablets before 3400 BC and continued for more than 3,000 years. More than 100,000 tablets still exist today, in various states of disintegration. They range in size from two to dozens of centimeters. Typically, they hold administrative data, document historic events and commercial transactions, and narrate everyday life.

Unfortunately, the writing used on clay tablets has evolved over time, and the evolution has contributed to a loss of the meaning of the language used. Early pictograms were written quickly and gradually evolved so that related pictograms ran together. Translating these inscribed texts has thus been difficult. In addition to translation problems, scholars of ancient writing also face the challenge of depicting the text inscribed on the cuneiform tablets when creating illustrations for use in books or research papers.

We've developed a semiautomatic method for concisely displaying the tablets' inscribed writing, thereby providing a clear visualization that can be printed on paper (for other approaches, see the "Previous Work" sidebar). We first scan the tablets with 3D range scanners and use the scan data to construct a high-resolution 3D model (at a resolution of 50 microns). Next, we unwrap and warp the tablet surface to form a set of flat rectangles, one per side or edge of the tablet. This process permits all the writing to be seen at once, although necessarily slightly distorted. Finally, we apply curvature coloring and accessibility coloring to the unwrapped text, thereby replacing raking illumination with a nonphotorealistic rendering technique.

Illustration problems

The simplest solution to the illustration problem is to photograph a tablet illuminated with a raking light source, as Figure 1a shows. Although most of the writing on a cuneiform tablet is on the front (obverse) or back (reverse) faces, these faces are often curved. The curvature causes distortions from foreshortening on the periphery when looking perpendicularly at the tablet's center (see Figure 1a). This effect is exacerbated by the fact that, on many cuneiform tablets, the text wraps around one or more edges. Finally, the raking illumination poses a prob-



1 **(a) A photograph of a small Ur III dynasty cuneiform tablet (from 2100 BC). (b) A Phong-shaded rendering of a 50-micron resolution 3D model containing 3.2-million triangles. Note that the writing wraps around the tablet's edges, making it difficult to see from a single picture. (c) The writing has been unwrapped and shown as a displacement map. (d) The unwrapped inscriptions have been accessibility-colored, curvature-colored, and rendered with Phong shading to enhance readability.**

0272-1716/02/$17.00 © 2002 IEEE

Previous Work

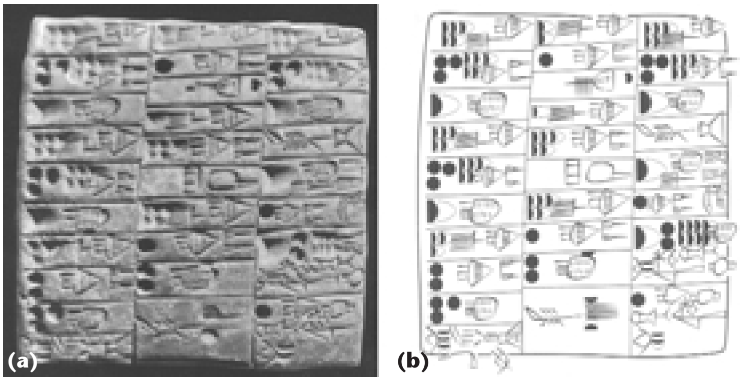
There are several major institutes of cuneiform study around the world, including the Faculty of Oriental Studies at Oxford, the Oriental Institute at the University of Chicago, the Department of Near East Languages and Cultures at UCLA, the Max Plank Institute for the History of Science in Berlin, and the Free University of Berlin. Researchers from the last three institutions in this list are collaborating on the Cuneiform Digital Library Initiative, in which tablets are scanned on a flatbed scanner and abstracted into pictographic symbols (as in Figure 2b). They use a vector-based graphics program, such as Adobe Illustrator, and human interaction.[1] In their system, a user selects a type of mark from a graphical palette. Next, a scanned tablet image is used as a reference to aid the user in accurate placement of symbolic marks, such as wedge shapes. The computer also performs a statistical analysis of the inscriptions, thereby deducing some meaning about them. Another capability of their system is the portrayal of the marks on the tablet as clear black-and-white illustrations, free of soil or the effects of weathering. However, cuneiform inscriptions are open to interpretation, so valuable information may be filtered out during this process. Also, although their system is effective on flat tablets, it doesn't work as well for rounded tablets or those whose writing extends around the tablet sides. By contrast, our method only visualizes the marks—it doesn't transcribe them.

Malzbender et al.[2] have recently visualized cuneiform tablets by taking pictures of a tablet with different light directions to compute what they call a *polynomial texture map*. These maps have a set of coefficients stored at each texel that are used for evaluating a biquadratic polynomial of the light direction. When viewing the clay artifacts, their method's effect is similar to using photometric stereo to estimate a normal map and then interactively rendering it with different materials and lighting.

In other work, Rushmeier et al.[3] published a method for computing *horizon maps* from images captured under controlled lighting. Horizon maps store self-shadowing information for a set of quantized azimuthal light directions at each point of a terrain surface. These may be rendered in real time with modern hardware. When used in conjunction with bump maps, horizon maps produce realistic renderings. In contrast to the 3D scanning required for our method, the input images may be acquired relatively quickly in both Malzbender's and Rushmeier's methods. However, their techniques don't address the unwrapping problem and aren't suitable for publishing tablets.

References

1. H.J. Nissen, P. Damerow, and R.K. Englund, *Archaic Bookkeeping*, Univ. of Chicago Press, 1993, pp. 45-46.
2. T. Malzbender, D. Gelb, and H. Wolters, "Polynomial Texture Maps," *Computer Graphics* (Proc. Siggraph 2001), ACM Press, New York, 2001, pp. 313-320.
3. H. Rushmeier, L. Balmelli, and F. Bernardini, "Horizon Map Capture," *Computer Graphics Forum*, vol. 20, no. 3, Sept. 2001, pp. C/85-94.



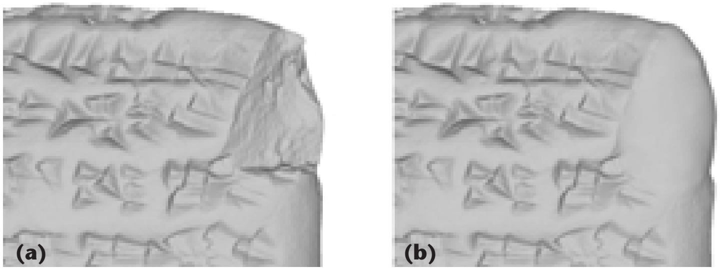
2 (a) An example of a raking-light photograph of a protocuneiform tablet (circa third millennium BC); it records an account of malt and barley groats used in beer production for four administration officials.[1] Although the front surface of this tablet is relatively flat, the text wraps around the edges in places, making it illegible from a single viewpoint. (b) A pen-and-ink style transcription of this tablet.

Photo courtesy Christie's London; illustration from Archaic Bookkeeping.

lem when parts of the tablet are in shadow, thus giving the observer no visual information about an inscription's critical portions. An additional form of obfuscation from illumination occurs when we nearly align the inscription's orientation with the incoming light direction.

An alternative to photography is to manually transcribe the tablet symbols, producing a result similar to Figure 2b. Unfortunately, this task is tedious, and it fails to portray the nuances and ambiguity of the original. Most applications require that the text be reproduced in a form that conveys the original in its entirety.

If we're willing to extend the publication medium beyond the printed book, we can consider using a computer to visualize the tablets. Digital photographs of the tablet may be taken from a variety of camera angles, processed, and viewed interactively, as in Apple's QuickTime Virtual Reality (QTVR). Another option is to scan the tablet, thereby creating a 3D model and then display this model using an interactive 3D rendering program. Such programs let users manipulate a virtual camera and light source relative to the model to view all the markings on the tablet. Yet another approach is the lightfield, which combines photographic imagery with 3D-like interactivity.[2] Lightfield viewers display images from new camera positions and orientations by interpolating from a dense array of photographic images taken from known camera positions and orientations. Here again, users can manipulate the virtual camera to



3 **(a) Tablet models may be broken (see upper right corner) with missing fragments. (b) Users may need to repair the missing portions so the displacement maps can be extracted.**

see all the tablet's inscribed characters. However, the printed page has benefits such as longevity and availability that computers don't.

Software pipeline

Figure 1 shows our software pipeline, which consists of three steps: scanning, unwrapping, and visualizing. The first step uses a combination of software and specialized hardware, while the last two involve only software. All the steps require modest amounts of human guidance to achieve optimum results.

Scanning

In the first step we scan a tablet with a 3D laser range scanner. Unfortunately, capturing the finest details on a cuneiform tablet requires a high-resolution 3D scanner. Robert Englund, professor of Near Eastern Languages and Cultures at the University of California, Los Angeles, told us that he found that the smallest details on cuneiform tablets (particularly from the small script of the second millennium BC and later) requires a resolution of 300 dots per inch (dpi), with 600 dpi preferred. After some experimentation, we confirmed his finding. This implies that we need a 3D scanner with a resolution of 50 microns. For the figures in this article, we used a laser triangulation range scanner built by the National Research Council of Canada (http://www.vit.iit.nrc.ca/) with exactly this resolution.

Because each scan of an object using a swept laser scanner shows only one side of the object, we must scan the object multiple times from different viewpoints and merge the resulting range images. Although we could merge range scans in several ways, we used a discrete volumetric intermediate representation of a cumulative weighted signed distance function (see http://graphics.stanford.edu/software/vrip/ for more information on this approach). We scan-converted each range image to a distance function and then combined it with data already acquired using a simple additive scheme. Extracting the zero-distance isosurface from the accumulated volume generates the final dense triangle mesh manifold. Applying this method to 22 range images of the tablet in Figure 1a, we created a 3.2-million-triangle model with a resolution of 50 microns (see Figure 1b).

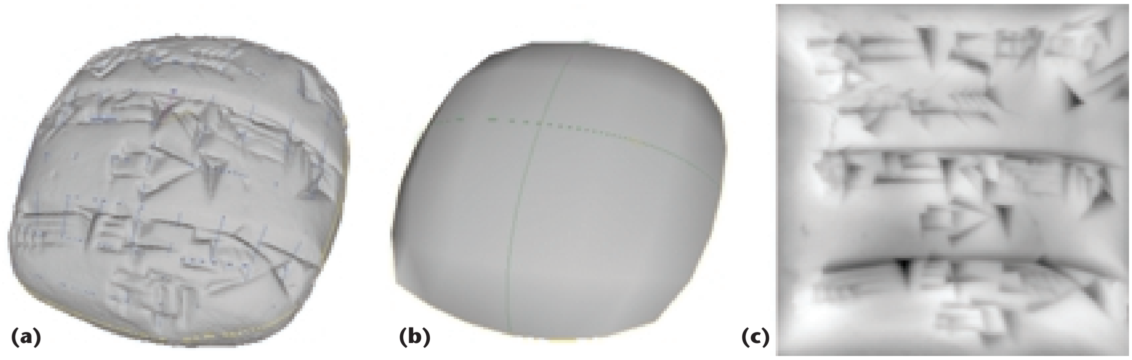
In the next step of the pipeline, we unwrap approximately rectangular patches of text, but we must first prepare the mesh for the unwrapping. Many cuneiform tablets are broken or are missing significant pieces, as Figure 3 shows. Unwrapping a rectangular patch from a (broken) tablet that's L-shaped would leave the unwrapped patch extremely distorted. Therefore, we virtually repair the tablet model by using Paraform's software to define new geometry where pieces have broken away from the original tablet. The resulting repaired areas don't have inscriptions, but approximate the tablet's shape prior to writing. In particular, Paraform's software provides a virtual spherical tool with which we can interactively sculpt the mesh's broken surfaces, gradually pushing the vertices outward to approximate the unbroken surface (see Figure 3b).

Unwrapping

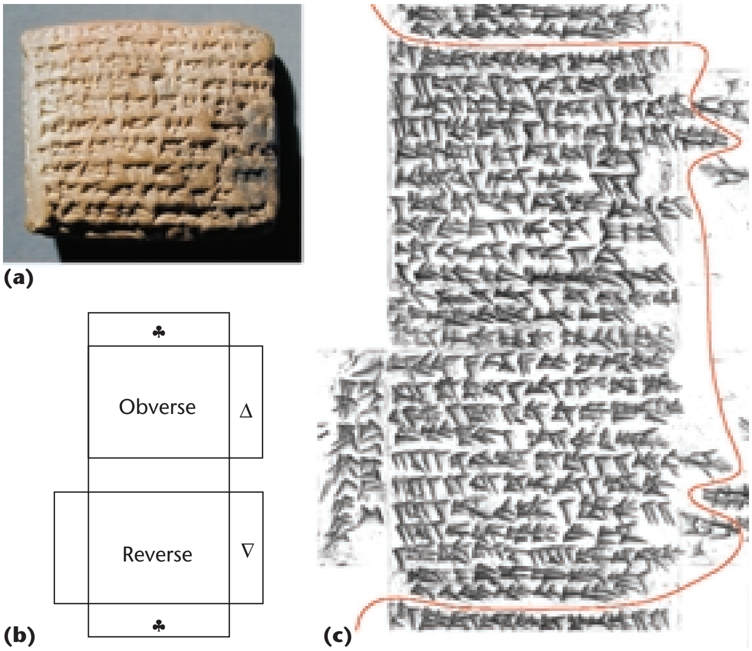
The second step in our pipeline is to unwrap the characters from the model. Once again, using Paraform's software, we partition the irregular mesh into rectangular patches.[3] Specifically, for each patch, we interactively specify the patch boundary as four connected curves on the 3D tablet model's surface. Next, we fit a grid of springs to each patch by iteratively relaxing and subdividing the spring grid. Relaxing makes the gridlines more evenly spaced across the model, while subdividing creates a better fit of the spring grid to the irregular mesh. The resulting spring grid defines a 2D parameterization over the mesh. We then fit a coarse tensor product B-spline surface to the spring grid. The error between the fitted surface and the spring grid is represented as a displacement map (see Figure 4).[4] The reason for using this hybrid B-spline and displacement map representation is that, given suitable parameters in the fitting process, the B-spline surface captures the overall tablet shape. Plus, the displacement map captures the inscribed marks.

Because we can't fit an entire tablet using a single patch, we treat the tablets as approximating a rectangular box, and we fit one patch to each side. This effectively unfolds the tablet. Sides where the text runs over from both the obverse and reverse can be repeated (see Figure 5). For example, the patch (labeled Δ in Figure 5b) on the right of the obverse is repeated but rotated 180 degrees (and labeled ∇) on the right of the reverse, because the writing spills onto it from both the obverse and reverse sides. Likewise, the topmost patch (♣) is repeated as the bottommost, without rotation. These duplicated side patches let text running over from the obverse and reverse sides be read at a glance.

By discarding the B-spline surface and rendering the displacement map, the text appears unwrapped, so that all the script is visible at once (see Figure 5c).



4 **We decompose (a) the 3D computer model into (b) a coarse B-spline, which captures the basic tablet shape, and (c) a displacement map, which encodes the inscribed detail. We discard the B-spline surface, retaining only the displacement map.**



5 **We unwrapped (a) the tablet in the photograph using (b) the diagram, which resulted in (c) the computer rendering.**

Visualizing

The third and final step of our pipeline involves visualizing the inscriptions. The visualizations involve converting the displacement maps into dense triangle meshes, estimating normals at the vertices, accessibility- and curvature-coloring the vertices, and finally, rendering with Phong shading.

We can convert a displacement map into a triangle mesh by first dicing a plane into a dense rectangular mesh, so that each pixel in the displacement map image lies on a vertex of the rectangle mesh. We then move each vertex up or down from the plane by an amount proportional to the displacement value of the corresponding pixel. We then bisect each tiny rectangle diagonally to yield two triangles.

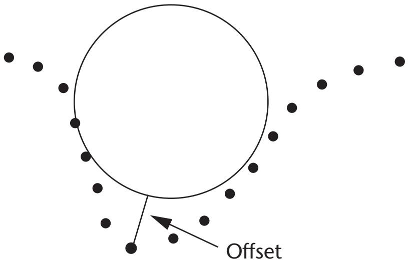
The resulting geometry may be used to calculate approximate normals at each vertex, which we need for renderings. To find a vertex's normal, we scale the normal of each triangle that incorporates that vertex by the triangle's area. We sum the area-scaled normals for all such triangles and divide by the total area of all triangles that incorporate that vertex.

At this point, we could render the newfound triangle mesh and normals with Phong lighting, but ambiguities resulting from incisions aligned with the light direction would persist. To reduce these ambiguities and visually enhance the inscriptions' shape, we artificially color the mesh's vertices, using accessibility coloring and curvature coloring, individually or in combination.

Accessibility coloring involves coloring each point on a mesh according to the maximum size of a probe sphere that can be placed at some fixed offset above the point without intersecting the mesh.[5] Using this technique, narrow crevices are darkly colored—where at most a small sphere can fit in them—while plateaus and mountaintops are lighter in color—where a much larger probe sphere meets the criterion. The offset improves the appearance of the result, because with zero offset, all vertices and edges associated with (even slightly) concave parts of the mesh would be colored black.

When dealing with dense triangle meshes, we compute accessibility only at vertices, and we approximate the accessibility at a vertex by determining if other vertices fall inside a sphere associated with that vertex. We then map this accessibility to color, and we bilinearly interpolate these colors across triangles. More formally,

```
For each vertex v ∈ verts(surface), do:
   Let real r = 0              // Radius
do:
   Let r = r + epsilon
   Let point c = v + NormalAt(v) * (r + offset)
   // Next find the closest point to c, excluding v
   Let point p = Nearest(verts(surface) – v, c)
While Distance(c, p) > r
Let r = r – epsilon
SetColor(v, power(r / maxRadius, gamma))
```



6 **To accessibility-color a vertex (shown as a dot), we must find the largest sphere that can fit at distance offset away along the normal assigned to that vertex.**

7 **(a) The unwrapped displacement map. (b) The same displacement map is directionally lit with a raking light disposed at the upper right. (c) Features only accessibility coloring. (d) Features only curvature coloring. (e) The product of the previous three images, being accessibility-colored and curvature-colored as well as illuminated with a raking light. (f) These images magnify the upper centers of Figures 7a through 7e.**

In this pseudocode, offset is the distance that the probe sphere is placed above the vertex (see Figure 6). The last line sets the color of a vertex by raising the maximum radius found to the power gamma. This nonlinear mapping of radii to colors distributes the colors in a more even and visually appealing manner.

Alternatively, we can color a surface according to curvature. To determine curvature coloring for a surface defined by a mesh, we approximate the mesh's curvature at each vertex using other vertices in its vicinity. Vertices with more absolute curvature (upward or downward), such as those in the bottoms of inscribed marks, receive darker coloring. To be precise,

```
For each vertex v on the surface, do:
    Let vector sum = (0, 0, 0)
    For each vertex w ∈ Neighborhood(v), do:
        Let sum = sum + NormalAt(w)
    Let sum = sum/CountNeighbors(v)
    Curv = 1 – dot(NormalAt(v), sum)
    RawColor = 1 – sqrt((2 – Curv) * Curv)
    SetColor(v, power(RawColor, gamma))
```

To combine the output of the two methods, we take the product of their gamma-mapped intensities.

Results and discussion

Figures 1d, 5c, 7e, 8, and 9d show examples of our pipeline. In our opinion, the combination of curvature coloring, accessibility coloring, and raking light yields the best enhancement of the inscriptions. Compared to Figures 1a, 2a, and 5a, each mark is clearly visible and readable in Figures 1d, 5c, 7e, and 8. We've shown these figures to several cuneiform scholars, who agree. The unwrapped inscriptions are visually comprehensible from a single image because darkening by accessibility coloring indicates their depths and curvature coloring enhances the 2D shapes of their troughs.

Despite these successes, our pipeline has several limitations. First, there's some geometric error introduced at each step in the process. The raw scans from the scanner are accurate to 50 microns. Provided they're resampled at a resolution of 25 microns or less when merged into a full model and also when creating the spring meshes, the values in the displacement map should be accurate to around 50 microns.

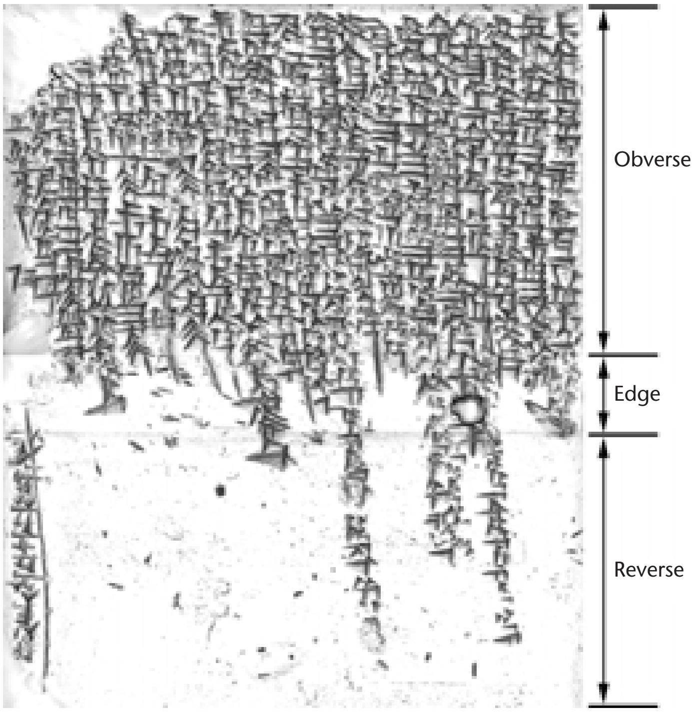
Second and most significantly, distortion is inherent in flattening curved surfaces onto planes (see Figure 9). By using tensor product surfaces to fit tablet surfaces, we assume that the tablet is well approximated by a box. To the extent this assumption is untrue, the unwrapped inscriptions appear warped, particularly around the corners of the displacement map image. This distortion may decrease the legibility of individual marks, which were made by pressing a physical stylus into a soft curved surface. We don't currently have a quantitative measure for this distortion, but in the worst case, it could conceivably transform the mark into a different word. However, we haven't seen such severe distortions in our examples.

We might employ an *interrupted* projection onto a plane, like the Interrupted Goode Homolosine Projection for mapping the Earth. As Figure 9c shows, such a projection would show the surface split apart with seams occurring along the tablet's ledger lines (the lines separating vertically adjacent rows of marks). This projection would more evenly distribute the distortion among the unwrapped characters. Achieving this projection would require a way to flatten the coarse B-spline surfaces, retaining their boundary shape. We haven't tried this.

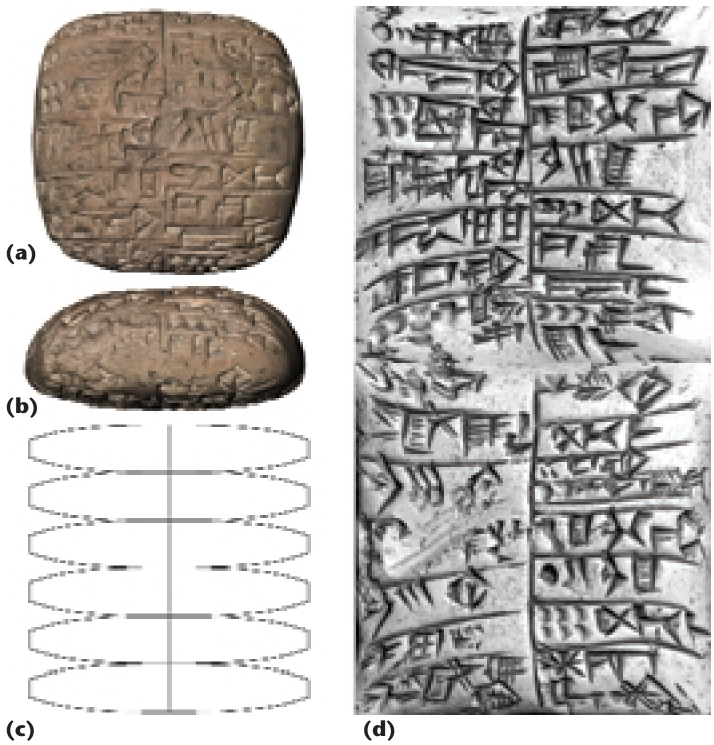
To further control distortion, we could draw Paraform feature curves along the ledger lines in the cuneiform text. These curves are treated as isoparametric lines in the fitting process, thereby forcing the ledger lines to be horizontal in the generated displacement map. This helps reduce the distortion in the output. We used feature curves in a couple places in Figure 9c to partially improve the output, but the distortion from tablet curvature is too severe to be fixed.

A third limitation of our pipeline is that large inscriptions pull the coarse B-spline surface (which is an approximation of the tablet's surface) down inside them. As a result, the measured depths of some inscriptions in the displacement map are less than they should be. We'd prefer to have the B-spline surface be the original surface of the clay, before the stylus had penetrated it. By using what's known as the alpha hull of the mesh, we may better approximate this virgin surface (see Figure 10, next page). The *alpha hull*,[6] a generalization of the convex hull, is defined as the complement of all spheres of radius alpha that don't intersect the surface. We haven't tried this either.

The fourth limitation is the inefficiency of the process of virtually repairing broken tablets by interactively adding geometry to the scanned 3D model. The process might be aided by using symmetries in a tablet's shape to approximate the missing geometry. A more comprehensive solution would permit arbitrarily shaped patches to be input and output, rather than forcing them to be rectangular. We could define an energy function for the mapping and minimize it through an optimization process.[7]



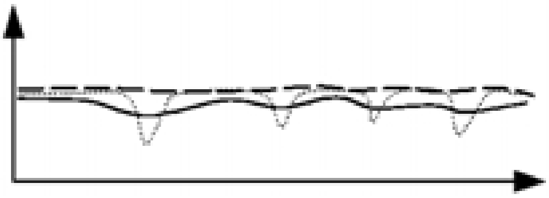
8 **The inscriptions, unwrapped and visualized using our pipeline, would be difficult to depict using photography. They would also be difficult to transcribe, because of the large number of small marks.**



9 **This example shows a highly curved tablet in top (a) and side views (b), respectively. As a result of the high curvature, the unwrapped text (d) becomes severely warped. A solution to this problem might involve cutting the rows into a number of split sections (c). In cartography, planar maps of the Earth that are split apart like this are termed *interrupted*.**

Future work

We've described methods for unwrapping and visualizing inscriptions on cuneiform tablets. These techniques should be applicable to other inscribed objects and artifacts as well. For example, the bas-reliefs on Trajan's column (in Rome, Italy), which has many small carvings over its surface, might be better visualized by employing the unwrapping and coloring approach presented here.



10 The dotted curve is the true surface; it's coarsely approximated by a B-spline surface, shown with the solid curve. But to approximate the virgin (unmarked) surface, we really want the alpha hull, drawn with the dashed curve. The difference between this curve and the true surface curves represents the inscriptions better.

We envision several directions for future work. One is toward inexpensive high-resolution 3D scanners, because the number of extant tablets is large (more than 100,000) and archiving projects aren't typically highly funded. A passive vision method, such as shape-from-stereo on a flatbed scanner,[8] might be capable of capturing the detail with sufficient resolution and it would be inexpensive enough to enable almost anyone with access to cuneiform tablets to create high-resolution models. However, we've experimented with this technique and were unable to reliably detect features, because of noise or the lack of sufficient distinguishable texture for the stereo algorithm to lock onto.

To improve the unwrapping phase, we could automatically compute the boundary curves of a patch on a 3D tablet model rather than requiring that a user draw them. However, such an automatic process would be feasible only if it included a method for minimizing distortion of the unwrapped tablet or for isolating the distortion to unimportant tablet parts.

Finally, having 3D tablet models may facilitate accurate cuneiform optical character recognition (OCR), which has been a dream of researchers. (Researchers have written several papers[9] about fast automatic cuneiform character recognition using optical correlators, but their emphasis is on speed rather than automation.) For written text, abstract symbols (see Figure 2b) derived from OCR may be superior to our nonphotorealistic rendering of displacement maps extracted from a 3D model. ■

Acknowledgments

We thank Renee Kovacs, who suggested the project and provided us with cuneiform tablets to scan. We'd also like to thank Szymon Rusinkiewicz and Matt Pharr for providing the accessibility and curvature coloring code that was used in producing the examples in this article. Also, we're grateful to Marc Rioux and Luc Cournoyer at the National Research Council of Canada for scanning a tablet for us, and to Paraform Corporation, which provided us with a version of its surfacing program to extract tablet model displacement maps. Lastly, Robert Englund provided valuable background information and was very encouraging of our research.

References

1. H.J. Nissen, P. Damerow, and R.K. Englund, *Archaic Bookkeeping*, Univ. of Chicago Press, 1993, pp. 45-46.
2. M. Levoy and P. Hanrahan, "Light Field Rendering," *Computer Graphics* (Proc. Siggraph 96), ACM Press, New York, 1996, pp. 31-41.
3. V. Krishnamurthy and M. Levoy, "Fitting Smooth Surfaces to Dense Polygon Meshes," *Computer Graphics* (Proc. Siggraph 96), ACM Press, New York, 1996, pp. 313-320.
4. R.L. Cook, "Shade Trees," *Computer Graphics* (Proc. Siggraph 84), ACM Press, New York, 1984, pp. 223-231.
5. G.S.P. Miller, "Efficient Algorithms for Local and Global Accessibility Shading," *Computer Graphics* (Proc. Siggraph 94), ACM Press, New York, 1994, pp. 319-326.
6. H. Edelsbrunner and E.P. Mücke, "Three-Dimensional Alpha Shapes," *ACM Trans. Graphics*, vol. 13, no. 1, 1994, pp. 43-72.
7. C. Bennis et al., "Piecewise Surface Flattening for Non-distorted Texture Mapping," *Computer Graphics* (Proc. Siggraph 91), ACM Press, New York, 1991, pp. 237-246.
8. R. Schubert, "Using a Flatbed Scanner as a Stereoscopic Near-Field Camera," *IEEE Computer Graphics and Applications*, vol. 20, no. 2, Mar./Apr. 2000, pp. 38-45.
9. N. Demoli et al., "Use of a Multifunctional Extended Optical Correlator for Cuneiform Inscription Analysis," *Proc. SPIE—The Int'l Soc. for Optical Eng.*, vol. 2297, SPIE Press, Bellingham, Wash., 1994, pp. 278-287.

***Sean Eron Anderson** is a PhD student in computer science at Stanford University. He earned a BS in computer science and mathematics at the University of Washington and an MS in computer science at Stanford. His interests include graphics, visualization, and gardening.*

***Marc Levoy** is an associate professor of computer science and electrical engineering at Stanford University. He holds degrees in architecture from Cornell University and a PhD in computer science from the University of North Carolina. His research interests include sensing and display technologies, image-based rendering, and applications of computer graphics in art history and archaeology. In 1996 he won the ACM Siggraph Achievement Award for his work in volume rendering.*

Readers may contact Sean Anderson and Marc Levoy at the Computer Science Dept., Stanford Univ., Stanford, CA 94305, email {seander,levoy}@cs.stanford.edu.