SHAPE — A UNIFYING CONCEPT IN DOCUMENT LAYOUT

Mike Bonham and Ian H. Witten

Department of Computer Science, University of Calgary, Canada T2N 1N4

Abstract

Text objects have traditionally been constrained to be rectangular, a constraint inherited by modern computer document preparation systems. However, a variety of tasks in document formatting benefit from a more general notion of *shape*. This paper describes such a representation, suitable for graphic composition at both low and high levels of document layout. Intermediate in generality and complexity between rectangles and arbitrary polygons, it comprises separate left and right margin functions composed piecewise of linear segments. Such shapes can be compared, combined, modified and generated using simple, economical algorithms. This notion of shape appears to provide the correct level of abstraction for elegant solutions to several knotty problems in existing systems. Software methods making use of shape have been implemented in JOT [Bonham 1985], an interactive documentation system under development as part of the University of Calgary JADE project [Witten *et al* 1983].

Introduction

Typography is an art which blends aesthetics and pragmatics. A reader's primary purpose is extracting a document's information content, and attention to aesthetics in the document design pays off in increased legibility and readability by making text smoother for the eye to scan, and by facilitating access to material through its layout [Duchastel 1982; Southall 1984]. The basis of graphic design is the arrangement of objects in relation to each other according to a host of aesthetic criteria such as balance, unity, simplicity, and rhythm. An attribute of graphic objects which is very important in creating designs is that of *shape*.

Shape is a perceptual phenomenon determined mainly by physical dimensions and boundary outlines, but also affected by axis orientation, texture, colour, density distribution, proximity to other objects, etc. Throughout the history of printing, text objects — ie. type slugs, words, lines, paragraph blocks — have been constrained to be rectangular, and this constraint is inherited by computer document preparation systems. Human typographers offset the mechanical limitations of the medium by making adjustments to letter and line spacing, and overall placement and layout. Computer-composed text has been less pleasing to the eye, primarily because the underlying graphic primitives are inadequate to represent design aesthetics. Dealing with shape as a basic primitive greatly expands our capabilities in computer-aided graphic layout.

The idea of shape is applicable at both low and high levels in document composition; this paper describes a general shape representation and its application to

† This paper was presented at the PROTEXT II conference in Dublin Ireland, October 1985.

three tasks: paragraph layout, word assembly, and interactive pointing.

Shapes

Our motivation for generalizing the usual rectangular shape representation derives from considerations such as:

- economical methods for character kerning [Kindersley 1976; Naiman 1984]
- fancy dropped initials in paragraphs [Sterken 1983]
- arbitrary cutouts in paragraphs [Knuth 1984, p.101])
- odd paragraph shapes [Witten *et al* 1982].

These point to a representation intermediate in generality and complexity between rectangles and arbitrary polygons. Accordingly, we represent shapes by functions for computing the left and right extents or "margins", based on vertical position within an object†. This representation is well suited to calculations involving abutting or comparing irregular outlines along rectilinear (left-right or up-down) axes. General rotations are impossible, but this does not appear to be a serious drawback because text objects have a preferred orientation.

Margins are assembled piecewise from simpler "segments". We limit segments to "constant" ($y = b$) and "linear" ($y = ax+b$) ones, but it is clear how to extend these if necessary. Segments are maintained in sorted order of ordinates of the endpoints, and never overlap adjacent ones. For example, Figure 1(b) shows the list data structures defining the shape of the zone depicted in Figure 1(a). Shapes of characters and other primitive graphic objects can be obtained by analyzing their underlying bitmaps or vector coordinates. In principle, the same spline curve outlines, used by systems such as POSTSCRIPT [Adobe Systems 1984] and METAFONT [Knuth 1979] for defining letter forms, could be used directly.

Operations on Shapes. Some of the operations which can be performed on shape functions are:

> *Evaluating the function value at a given ordinate* — used to determine left and right endpoints of a line of text from a paragraph's boundary shape and the vertical baseline position; or to determine whether a given coordinate point falls inside a object's boundaries for an interactive pick operation.
>
> *Adding or subtracting shapes* — for preparing paragraphs and running text with deviations from the rectangular form.
>
> *Computing the distance between two shapes* — to determine the minimum offset between two shapes without overlapping (kerning).
>
> *Computing area, centre of gravity and axis orientation* — to derive visually balanced compositions in two dimensions.

†We speak mainly of left and right shape boundaries because this is the orientation most frequently required during horizontal layout of text objects. However, the discussion applies equally to top and bottom margins used to compose things one above the another.

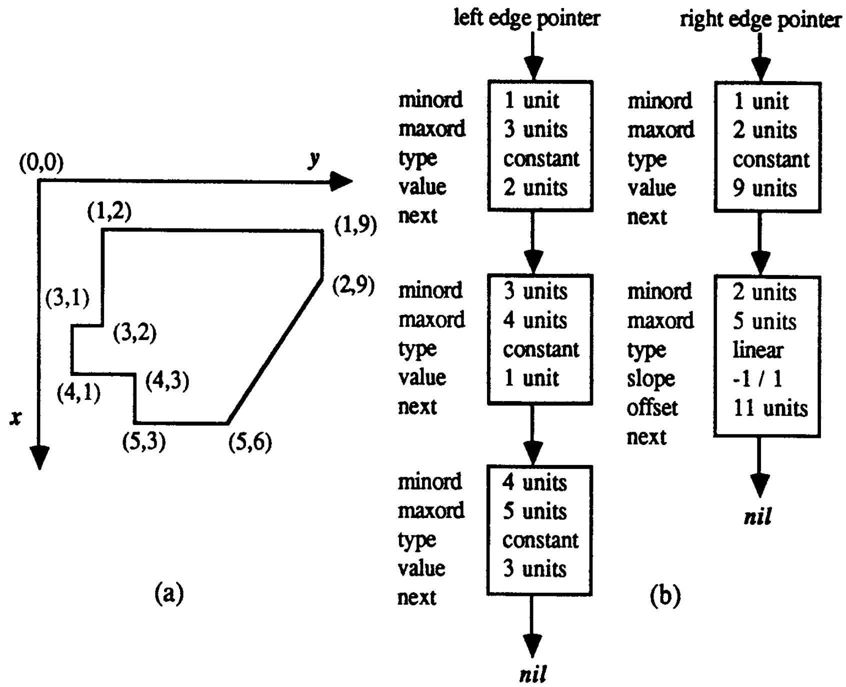

Figure 1. Irregular object (a) and Shape data structure (b)

Combining or comparing two shapes is done using a simple "sorted merge", which scans the lists of segments performing the intersection, maximum/minimum difference or superposition operation on individual segment pairs. Successive elements from each of the two sorted lists are compared or otherwise operated upon to produce a sorted output list. At each iteration the list "most in arrears" is advanced until one or both lists have been consumed. Such algorithms have linear time complexity in the number of edge segments — a most attractive property.

Haloes. Rather than tight boundary shapes, many applications require "haloes" or buffer zones which enclose objects at some distance. Defined as the locus of points at constant distance from the inner shape, a halo broadens bumps and fills in dents of the inner shape in the same way that a country's offshore sovereignty zone is based on its coastline.

Haloes can be computed piecewise from an original shape function by offsetting each segment's line equation by a constant and adding (using shape addition) a characteristically-shaped extension at both endpoints. This produces a canonical halo shape of non-overlapping segments. Several approximations to the halo are desirable under different circumstances. For example, Figure 2 shows, for the letter 'R', (a) the tight boundary shape; (b) a canonical or circular-pen halo; (c) a square halo which adds a square extension at each endpoint; and (d) a beveled halo which adds a diamond shape to each endpoint. More sophisticated algorithms could simulate the continuous path of an arbitrary-sized and shaped pen around the character's perimeter.

Applications of Shapes and Haloes

Software methods making use of shape have been developed for a variety of tasks in document formatting, and implemented in JOT [Bonham 1985], an interactive documentation system under development as part of the University of Calgary JADE project [Witten *et al* 1983]. Here we discuss representative problems which shape functions solve in a way that improves on or rationalizes methods used in other text manipulation systems.

Irregular Paragraph Margins. A variety of effects involving non-rectangular text shapes are often sought in documents. Whether exotic or prosaic, the shape abstraction simplifies and unifies their specification. For example, a very ordinary requirement is the **indented first line** of a paragraph. Though easy to achieve using conventional methods, a conceptually "cleaner" method is to subtract a small rectangle from the upper left corner of the paragraph shape. This has the signal advantage that it generalizes directly to fancy paragraphing styles which distinguish leading paragraphs, perhaps by a large **initial capital** letter around which the first few text lines are cut in. It is trivial to mould the text indentation shape to the shape of the letter by subtracting the halo of the initial capital from the paragraph shape at the upper left corner. The point is not that this is a particularly necessary typographical construction, but that it is specified at the right level of abstraction through the use of shape.

More exotic are **special shapes** of body text, seen in annotated maps and drawings, advertising displays, and other works which integrate text and graphics. The ease with which such effects can be achieved is an indication of the power and flexibility of the layout primitives in a formatting system [Witten *et al* 1982]. **Side-notes** and **graphic cut-ins** (such as are seen in dictionaries with small figures and graphics set into the text) provide another application of paragraph shaping. Before configuring the main text, the length and shape of the side-note must be known.

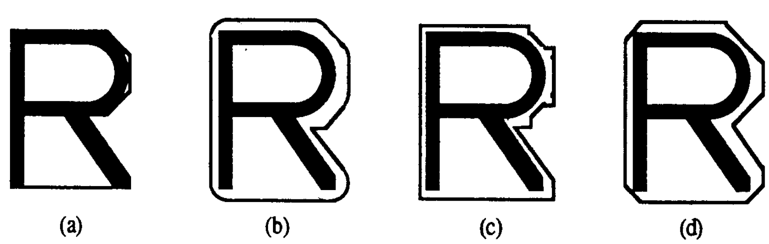

Figure 2. (a) Shape (b) Round halo (c) Square halo (d) Beveled halo

Because a cut-in is usually centred around the line containing its callout some look-ahead is required, and it this makes it complicated to coordinate side-note with body text formatting. Without the high-level view afforded by shape functions, the problem would be quite intractable.

Figure 3 shows an example paragraph exhibiting a dropped initial capital, shaped left margin, and cut-in text block. What is remarkable is that, using *shape*, these three separate effects can be combined quite naturally.

Kerning and Word Assembly. On a display device with proportionally-spaced characters, "kerning" and "letter spacing" are techniques for adjusting inter-character gaps to account for visual properties of each combination of letters. Once reserved for titles, headings and other "display text" because of its time-consuming nature, kerning is common nowadays in ordinary computer-typeset text. Automatic kerning usually involves consulting a "kerning table" containing spacing adjustments for each pairwise combination of letters. The table is determined empirically and does not take

```
~~|~~ was  brillig, and the slithy toves did gyre and gimbol
  | in  the  wabe.   All  mimsy  were  the  borogoves!   And
  | the    mome-raths  outgrabe.   Il  brilgue:   les   toves
   lubricilleux    se  gyrent  en  vrillant  dans  le  guave.
    Enmimes  sont  les gougebosqueux;  et le momerade  hors-
     grave.   "Garde-toi du Jaseroque, mon fils!  La  gueule
      qui  mord;   la giffe  qui  prend!
       Garde-toi de l'oiseau Jube, evite   Here  is a quite  silly
        le  frumieux Band-a-prend!"  Son   comment  cut-in to  the
         glaive  vorpal en main, il  va-   right margin of a  very
          T-a   la  recherche  du  fauve   oddly-shaped paragraph.
           manscant;    puis  arrive   a
            l'arbre   Te-te,  il  y  reste,   reflechissant.
             Pendant qu'il pense, tout uffuse, le jaseroque,
              a  l'oeil flambant, vient siblant par le  bois
               tullegeais,  et burbule en venant.  En  deux,
                un  deux,  par le milieu, le  glaive  vorpal
                 fait  pat-a-pan!  La bete defaite, avec  sa
                  tete,  il rentre gallomphant.  "As-tu  tue
                   le  Jaseroque?   Viens a mon coeur,  fils
                    rayonnais!  O jour frabbejais!  Calleau!
                     Callai!"  Il  cortule das sa joie.   Il
                      brilgue:   les  toves lubricilleux  se
                       gyrent  en  vrillant dans  le  guave.
                        Enmimes  sont les gougebosqueux;  et
                         le momerade hors-grave.
```

Figure 3. Shaped paragraph with dropped initial and cut-in.

into account the wider context of other letters in the word; neither does the method handle unanticipated situations where adjacent characters are from different fonts or are offset vertically. Consequently it is advantageous to kern automatically, based on character shape. For example, bitmap encodings of the left and right profiles of letter shapes have been used to compute letter spacing and kerning [Kindersley 1976; Naiman 1984].

Haloes are ideal for this purpose. Though our piecewise encoding occupies more storage and requires more computation to evaluate than a bitmap, it is more easily specified and handles a wider class of situations. Words are assembled by computing distances between, and combining, the shapes of successive pairs of letters to produce relative offsets and an overall word boundary shape (for later use in line and paragraph assembly). Adjacent shapes are composed so that the haloes exactly touch but do not overlap. If particularly sensitive kerning is desired, haloes can be adjusted manually, retaining the advantages of the uniform shape representation and allowing unusual combinations to be kerned automatically.

A variety of hybrid mechanisms can be imagined for reducing computation. For example, a conventional kerning table could serve as a cache, being updated as each new letter pair is encountered and kerned.

Interactive Pointing and Feedback. Increasingly, techniques of typography are being applied to interactive information display systems utilizing (relatively) high-resolution bitmap displays and pointing devices such as the "mouse". Determining which graphical object is being picked is a standard problem in mainstream interactive graphics. Text displays mercifully do not usually contain overlapping objects, which complicate the picking problem in an obvious way. However, one text object may protrude into another, so that rectangular bounding boxes are inadequate for accurate picking. The abstraction of shape, through the use of haloes, supports interactive selection of objects in two ways:

- by defining a pick area for each object
- as a suitable candidate for outlining, flashing, or highlighting to inform the user which object has been selected.

We are also intrigued by the idea of using haloes to stand in for the actual objects during fast screen update. Until computers can display text as fast as people flip pages, visual scanning will be limited by screen update speed. A document's structure, as manifested by its formatted appearance, is a major cue guiding a reader's search for information. Showing the outline of text on a page — ie. paragraph halo shapes — may be faster than rendering the text in full. Rapid scrolling of on-line documents, showing only the outlines of headings, paragraphs and figures, may be a useful way of expediting information access. The full text would be displayed in detail when the user stops scanning.

Conclusions

While all text manipulation systems deal with shaped objects, most often shape is not recognized and represented explicitly but is treated in an *ad hoc* fashion at each level of document structure. Adopting a unified representation results in simple and elegant algorithms, consistent overall design, and economic implementation, for a number of text formatting and interaction operations.

Using highly-programmable systems (such as TROFF and TEX) it is possible to achieve almost any special effect desired. However, considerable difficulty is encountered when integrating one effect with another, for unanticipated interactions appear when constructions are combined. The use of a common abstraction which encompasses many independent functions gives them a much greater chance of working in combination. We believe *shape* to be just such a complexity-reducing abstraction.

Acknowledgement. Support from the Natural Sciences and Engineering Research Council of Canada is gratefully acknowledged.

References

Adobe Systems, Inc. (September 1984) "POSTSCRIPT Language Manual, First Edition, Revised" Palo Alto, California.

Bonham, M.H. (1985) "Viewing and Formatting Documents On-Line" MSc Thesis, research report no. 85/195/8, Department of Computer Science, University of Calgary.

Duchastel, P.C. (1982) "Textual Display Techniques" in *The Technology of Text,* edited by Jonassen, D.H., pp 167-191. Educational Technology Publications, Englewood Cliffs, New Jersey.

Kindersley, D. (1976) *Optical Letter Spacing for New Printing Systems.* Sandstone Press, New York.

Knuth, D.E. (1979) "METAFONT: a system for alphabet design" in *TEX and METAFONT, New Directions in Typesetting.* Digital Press.

Knuth, D.E. (1984) *The TEXbook.* American Mathematical Society, Addison-Wesley Publishing Company.

Naiman, A. (May 1984) "Some New Ingredients for the Cookbook Approach to Anti-Aliased Text" *Proceedings of Graphics Interface '84,* 99-108, National Computer Graphics Association of Canada, Ottawa, Ontario.

Southall, R. (1984) "First principles of typographic design for document production" *TUGBOAT, 5* (2) 79-90.

Sterken, J. (1983) "Problems: the first lines of a paragraph" *TUGBOAT (TEX Users Group Newsletter), 4* (2) 80.

Witten, I.H., Bonham, M.H., and Strong, E. (1982) "On the power of traps and diversions in a document preparation language" *Software — Practice and Experience, 12,* 1119-1131.

Witten, I.H., Birtwistle, G.M., Cleary, J., Hill, D.R., Levinson, D., Lomow, G., Neal, R., Peterson, M., Unger, B.W., and Wyvill, B. (1983) "JADE: a distributed software prototyping environment" *ACM Operating Systems Review, 17* (3) 10-23, July.