Adaptive polygonal approximation of parametric curves

Luiz Henrique de Figueiredo

IMPA–Instituto de Matemática Pura e Aplicada
Estrada Dona Castorina, 110
22460–320 Rio de Janeiro, RJ, Brasil
lhf@visgraf.impa.br

Abstract. We present methods for constructing polygonal approximation of parametric curves based on adaptively sampling the parameter domain with respect to curvature. An important feature of these methods is the use of random probing for handling aliasing. We also discuss numerical applications of this method.

Introduction

Parametric curves occur frequently in geometric modeling and other applications of computer graphics to science. Models and images of parametric curves are usually created from piecewise linear approximations because these approximations can be easily computed and rendered.

To compute a piecewise linear approximation of a parametric curve, it is enough to discretize the parameter domain; we call this process **sampling**. The various methods for piecewise linear approximation differ, of course, in the way the domain is sampled. An efficient method should be able to construct precise approximations using small samples (as usual, these are conflicting goals).

The most popular sampling method is uniform sampling, which can be very inefficient if high precision is needed. In this paper, we present a class of methods for sampling the parameter domain adaptively with respect to curvature. This means that samples are more concentrated where the curve has high curvature and less concentrated where the curve is almost flat, resulting in a more efficient approximation.

Previous work on adaptive sampling either was restricted to special curves, such as Bézier curves [Farin (1990)], or used more expensive sampling criteria [Chandler (1990), Lindgren–Sanchez–Hall (1992), Wüthrich (1992)]. In this paper, we extend this work by discussing several sampling criteria; random probing play an important part in implementing these criteria and handling aliasing. We also discuss applications of this sampling method to rasterization, numerical integration, arc length parametrization and data reduction.

Polygonal approximation

Let $\gamma : U = [a,b] \to \mathbf{R}^d$ be a parametric curve. To construct a piecewise linear approximation for $\gamma(U)$, it is enough to choose sample points $a = t_0 < t_1 < \ldots < t_n = b$ in U and take the polygonal line with vertices p_0, p_1, …, p_n, where $p_i = \gamma(t_i)$.

The efficiency of approximation methods depends on the sampling done on U. The number of times the function γ is evaluated is a good measure of complexity for comparing approximation methods. Note that minimizing the size of the sample is equivalent to minimizing the number of evaluations of γ. Thus, the challenge is to choose sample points t_0, t_1, …, t_n which induce a good approximation, while keeping n small.

Aliasing

Polygonal approximation is actually a reconstruction problem: a continuous curve is to be reconstructed from a discrete sample. An intrinsic problem in reconstruction is **aliasing**: the behaviour of the curve between two consecutive sample points cannot be completely controlled, possibly resulting in a lower frequency approximation which ignores local high frequency oscillations in the original curve.

Classical work on signal processing shows that the effects of aliasing can be diminished by using fine sampling, but aliasing cannot be completely avoided, specially in the presence of high frequencies [Foley–van Dam–Feiner–Hughes (1990)]. For example, consider $\gamma(t) = (t, \sin 5t)$ on $U = [0, n\pi/2]$. Then, the sample t_0, t_1, …, t_n, given by $t_i = i\pi/2$, is not representative of γ on this interval but is representative of $\alpha(t) = (t, \sin t)$, a lower frequency curve (Fig. 1).

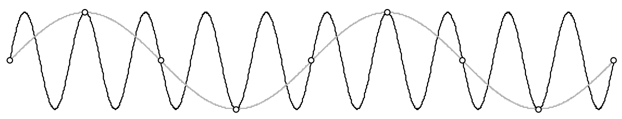

Figure 1: Aliasing.

Uniform sampling

The most popular sampling method is **uniform sampling**: choose $\delta > 0$, divide the interval U uniformly into pieces of size δ and take the extremes of these sub-intervals as the sample points, i.e., take $t_i = a + i\delta$, for $i = 0, \ldots, n$, where $n = \lceil (b-a)/\delta \rceil$ (Fig. 2). Equivalently, instead of choosing a mesh size δ, we can choose a number n of divisions of U and set $\delta = (b-a)/n$. Because it is unlikely that $b - a$ is an integral multiple of δ, some correction is needed if we choose δ instead of n: either set x_n explicitly equal to b, shortening the last sub-interval, or use $\delta' = (b-a)/n$ instead of the original δ, where $n = \lceil (b-a)/\delta \rceil$.

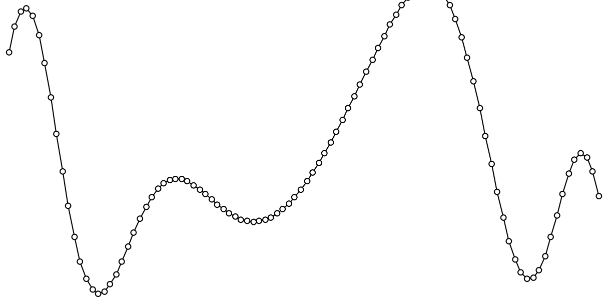
Figure 2: Uniform approximation.

This method is very simple and deservedly popular for its simplicity, but its efficiency depends on the choice of δ (or n), because γ is always evaluated $n+1$ times, regardless of the curve. Moreover, it is necessary to choose a small δ (or large n) to obtain a good approximation, but, in practice, this choice can only be made by trial and error. However, if δ is too small, and the approximation is being built for rendering, then the sample will be inefficient because many sample points will contribute a single pixel in the image. Large uniform samples are inefficient for other applications as well (e.g., modeling), because objects built from them tend to require large data structures. Thus, uniform sampling can be very inefficient if high precision is desired.

Adaptive sampling

An approximation based on uniform sampling does not have the same precision everywhere: the approximation is better where the curve is close to a straight line, i.e., where the curvature of the curve is low. In these parts, a fine sampling is not needed and we can replace many small, almost collinear, consecutive line segments by a single line segment, thus avoiding evaluating γ on several intermediate points (Fig. 3). Therefore, it should be possible to replace uniform sampling with a non-uniform sampling that is adapted to the function γ, using curvature as a criterion for sampling refinement, i.e., sampling finely only where γ has high curvature. In this section, we describe a class of methods for adaptive sampling.

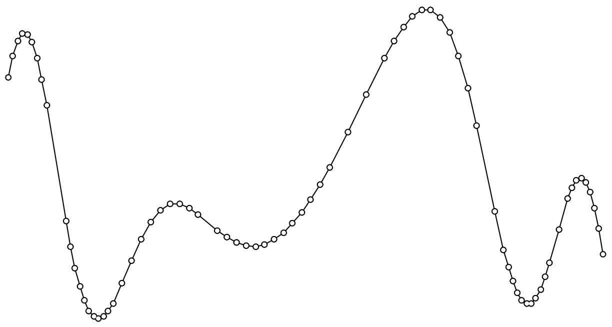
Figure 3: Adaptive approximation.

We shall adopt the following strategy for adaptive sampling of an interval:

```
· choose a criterion for refining samples.
· evaluate the criterion on the interval.
· if the curve is almost flat in the interval
      the sample is given by its two extremes;
· otherwise
      divide the interval in two parts;
      recursively sample the two parts.
```

This strategy is similar to using the de Casteljau algorithm for Bézier curves, stopping when the control polygon is almost flat [Farin (1990)]. We shall now discuss heuristic refinement criteria that are applicable to general curves.

Refinement criteria

One implementation of the sampling refinement criterion based on curvature is to choose a point w interior to the interval $[u, v]$ being considered and check whether the three corresponding points on the curve $p = \gamma(u)$, $q = \gamma(v)$ and $r = \gamma(w)$ are approximately collinear; we call this **probing**. The following numerical criteria are natural for checking local flatness:

- the area of the triangle pqr is small;
- the angle $\angle prq$ is large, close to π;
- r is close to the chord pq;
- $|p - r| + |r - q|$ is approximately equal to $|p - q|$;
- the tangents to γ at p, q, and r are approximately parallel.

Note that a tolerance needs to be chosen for evaluating these criteria. Once a tolerance is chosen, a criterion holds for p, q and r if and only if r lies in a certain region around the segment pq (Fig. 4). For rendering, the tolerance is directly related to the size of the pixels in the image; the chord distance criterion is specially useful in this case (see below for a rasterization algorithm).

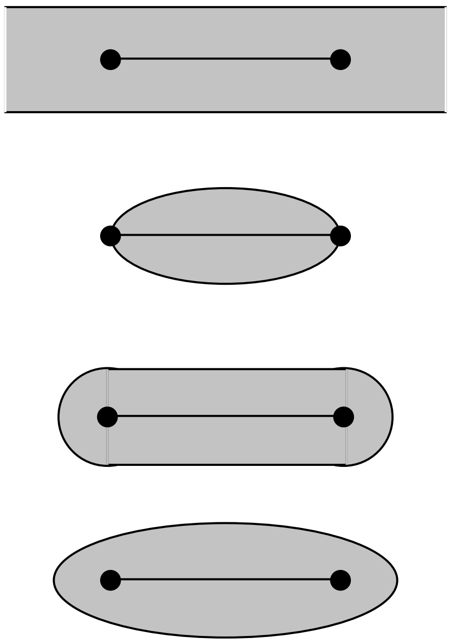
Figure 4: Tolerance zones for local flatness.

Only the angle and tangent criteria are invariant under scaling. For the others, if the scale is changed, then the tolerance must be changed accordingly, in order to obtain the same results. In practice, they all seem to be equally effective in locating pieces of low curvature. However, the tangent criterion is only useful if the derivative of γ is available (e.g., if γ is given by a formula), because a sampling adequate for numerical differentiation is actually finer than is required for rendering.

The area criterion is specially efficient to program, because, unlike the others, it does not need square roots. On the other hand, this criterion is the weakest in theory because the corresponding neighbourhood is infinite and increases in diameter when the segment pq decreases in length (i.e., if the chord pq is short, then the area of the triangle pqr can be small even if the point r is far from pq; as a consequence, the area criterion is not suited to closed curves). From this point of view, the angle criterion is the strongest because the corresponding neighbourhood is tighter than the others. Nevertheless, we have used the area criterion in our implementation because of its simplicity, with good results.

It is important to point out that these criteria are heuristic: even if r is in the neighbourhood of pq, there is no guarantee that the curve is locally flat, because other points in the curve between p and q may lie outside the neighbourhood. (If additional information on the curve is available, such as a Lipschitz bound [von Herzen–Barr (1987)], then these criteria become exact.) In other words, probing does not avoid aliasing completely. Nevertheless, the criteria are adequate in practice, specially given our choice of interior point, which we shall now discuss.

Choosing the interior point

The natural choice for the interior point w is the midpoint of the interval. However, any deterministic probing is vulnerable to aliasing. For example, consider $[u, v] = [-\pi, \pi]$ and $\gamma(t) = (t, \sin t)$. If we take w as the midpoint, then p, q and r are collinear, but γ is certainly not flat in this interval (a similar example was given by Chandler (1990)). To avoid aliasing due to such symmetries in γ, it is safer to choose a random interior point, possibly near the midpoint; we call this **random probing**. In our implementation, we have taken $w = u + \lambda(v - u)$, where λ is a random number in the interval $[0.45, 0.55]$.

Wüthrich (1992) proposed a similar adaptive method, based on the chord criterion, which chooses w satisfying the mean-value theorem for the interval $[u, v]$, i.e., such that $\gamma'(w)$ is parallel to the chord pq. The point r is thus the point on the curve farthest from pq. His method is exact (not heuristic), but requires the solution of a non-linear equation at each step to find w.

The algorithm

For the sake of efficiency, it is possible to arrange the sampling so that the interior point used for evaluating the refinement criterion is also used to divide the interval, in case it is necessary to do recursion. In this way, γ is evaluated exactly once at each sample point. An algorithm for rendering based on these ideas can then be written:

```
sample($u, v, p, q$):
    $w \leftarrow$ random point in $[u, v]$
    $r \leftarrow \gamma(w)$
    if flat($p, q, r$)
        line($p, q$)
    else
        sample($u, w, p, r$)
        sample($w, v, r, q$)
```

A sampling of the domain $U = [a, b]$ using this algorithm starts with sample($a, b, \gamma(a), \gamma(b)$). The function flat implements one of the refinement criteria suggested above. The function line is the basic rendering function: it draws the line segment pq; this function must be replaced for other applications.

It is interesting to note that this algorithm generates the points in the exact order that they occur on the curve, despite the recursion. In fact, it is possible to adapt this algorithm to perform a direct rasterization of the curve, instead of relying on a polygonal approximation. First, use $\bar{\gamma}$ instead of γ, where $\bar{\gamma}(t)$ is the pixel corresponding to the point $\gamma(t)$. Next, check whether p and q are neighbouring pixels, instead of checking for local flatness. Finally, plot the pixel p instead of drawing the line segment pq:

```
raster($u, v, p, q$):
    if neighbours($p, q$)
        plot($p$)
    else
        $w \leftarrow$ random point in $[u, v]$
        $r \leftarrow \bar{\gamma}(w)$
        raster($u, w, p, r$)
        raster($w, v, r, q$)
```

Note that no tolerance needs to be chosen for this algorithm; it depends solely on the zoom factor and on the resolution of the image.

Complexity

If the final sample has n points, then the algorithm sample evaluates γ exactly $2n-1$ times: once at each sample point and once at an interior point in each final sub-interval. This number of evaluations should not be naively compared with the n evaluations used in a uniform sampling with n sample points. A fair comparison is to compare $2n-1$ with the number of evaluations in a uniform sampling with mesh size δ equal to the size of the smallest sub-interval in an adaptive sampling.

Multiple probing

It is possible (and probably safer) to evaluate the refinement criterion on more than one interior point, either separately or at the same time, resulting in higher order recursion. For instance, Chandler (1990) proposes the use of $k-1$ uniformly spaced interior points and a refinement criterion based on areas, resulting in a k-ary recursion. He gives examples showing that binary recursion ($k = 2$) suffers from aliasing problems which are solved by using ternary recursion ($k = 3$). Chandler does not suggest a criterion for selecting k, but higher values of k are probably not needed in practice. Moreover, uniform probing the interior of the interval is vulnerable to aliasing; a better strategy for Chandler's method would be to use multiple random probing instead. In any case, the binary recursion scheme proposed here, based on single random probing, is simpler and appears to be adequate in practice.

Applications

Adaptive sampling is not only useful for rendering or modeling, but also for other approximation problems such as numerical integration in various flavors: quadrature of explicit functions; line integrals; rectification and arc length parametrization of curves.

For these problems, we obtain an adaptive integration method in which, unlike classical methods, refinement is not based on evaluating and comparing two numerical methods, or a single method at different resolutions [Forsythe–Malcolm–Moler (1977), Guenter–Parent (1990)]. In fact, the method implicitly constructs a good piecewise linear approximation for which the integration problem is very simple and can be solved exactly. For example, an algorithm for computing the length of a parametric curve can be written:

```
length($u, v, p, q$):
    $w \leftarrow$ random point in $[u, v]$
    $r \leftarrow \gamma(w)$
    if flat($p, q, r$)
        return $|p - q|$
    else
        return length($u, w, p, r$)+length($w, v, r, q$)
```

Arc length parametrization

Arc length parametrizations are useful in animation control [Guenter–Parent (1990)]. To compute an arc length parametrization of a parametric curve, start with an adaptive sampling t_0, t_1, …, t_n of the domain and set $s_0 = 0$ and $s_i = s_{i-1} + |\gamma(t_i) - \gamma(t_{i-1})|$, for $i = 1, \ldots, n$. Now, for each $s \in [s_0, s_n]$, find the interval $[s_i, s_{i+1}]$ containing s; this can be done using binary search in time $O(\log n)$. Next, find $\sigma(s) := t \in [t_i, t_{i+1}]$ by linear interpolation, i.e., such that:

$$\frac{s - s_i}{s_{i+1} - s_i} = \frac{t - t_i}{t_{i+1} - t_i}.$$

With this definition of σ, an arc length parametrization is given by $\alpha(s) = \gamma(\sigma(s))$.

Data reduction

Digitized polygonal lines, either manually or automatically extracted from images, usually have a large number of vertices. The ideas present in the adaptive sampling method can also be used for reducing the number of vertices defining such a polygonal line. Consider the chord defined by the two extreme vertices. If all vertices are close to this line, then the polygonal line can be replaced by the chord. Otherwise, find the point farthest from the chord, subdivide the polygonal line into two other polygonal lines and use recursion. This algorithm is actually well known in digital cartography [Visvalingam–Whyatt (1990)]; it is also the basis for a hierarchical representation of curves [Ballard (1981)].

Conclusion

We have presented an adaptive sampling method for parametric curves, based on recursive subdivision controlled by random probing. Although it is a heuristic method, it performs adequately in practice. Moreover, it is simple to implement.

The sampling method can be used for computing polygonal approximations which can then be used for building geometric models, rendering, and numerical integration and data reduction. It can also be adapted to provide a method for direct rasterization.

This method also works for surfaces and higher dimensional objects, although care must be taken to ensure that an approximation based on this sample is continuous [von Herzen–Barr (1987)].

Acknowledgements

The research reported here was done in the VisGraf computer graphics laboratory at IMPA and during a visit to the Computer Science Department at Unicamp (State University of Campinas). The VisGraf project is partially supported by CNPq, FAPERJ, and IBM Brasil. My visit to Unicamp was supported by FAPESP. I am grateful to Marcelo Gattass for our discussions on this subject.

References

D. H. Ballard, Strip trees: a hierarchical representation for curves, *Communications of the ACM* **24** (1981) 310–321.

R. E. Chandler, A recursive technique for rendering parametric curves, *Computers & Graphics* **14** (1990) 477–479.

G. Farin, *Curves and Surfaces for Computer Aided Geometric Design*, second edition, Academic Press, 1990.

G. E. Forsythe, M. A. Malcolm, C. B. Moler, *Computer Methods for Mathematical Computations*, Prentice-Hall, 1977.

J. D. Foley, A. van Dam, S. K. Feiner, J. F. Hughes, *Computer Graphics: principles and practice*, Addison-Wesley, 1990.

B. Guenter, R. Parent, Computing the arc length of parametric curves, *IEEE Computer Graphics & Applications* **10** (1990) 72–78.

B. von Herzen, A. H. Barr, Accurate triangulations of deformed intersecting surfaces, *Computer Graphics* **21** (1987) 103–110 (Proceedings of SIGGRAPH '87).

T. Lindgren, J. Sanchez, J. Hall, Curve tesselation criteria through sampling, in: D. Kirk (ed), *Graphics Gems III*, Academic Press, 1992, 262–265.

M. Visvalingam, J. D. Whyatt, The Douglas–Peucker algorithm for line simplification: re-evaluation through visualization, *Computer Graphics Forum* **9** (1990) 213–228.

C. A. Wüthrich, An analysis of digitalizations algorithms for non-linear curves, in: B. Falcidieno, I. Herman, C. Pienovi (eds.), *Computer Graphics and Mathematics*, Springer, 1992, 285–297.