Comparing Offset Curve Approximation Methods

Gershon Elber
Technion, Israel

In-Kwon Lee and Myung-Soo Kim
Postech, Korea

Offset curves have diverse engineering applications, spurring extensive research on various offset techniques. Research in the early 1980s focused on approximation techniques to solve immediate application problems. This trend continued until 1988, when Hoschek applied nonlinear optimization techniques to the offset approximation problem.[1,2] Since then, it has become quite difficult to improve the state of the art of offset approximation, with only incremental advances achieved.

Offset research turned more theoretical in the 1990s. Farouki and Neff[3,4] clarified the fundamental difficulty of exact offset computation. Farouki and Sakkalis[5] suggested the Pythagorean Hodograph curves, which allow simple rational representation of their exact offset curves. Although many useful plane curves such as conics do not belong to this class, the Pythagorean Hodograph curves might have much potential in practice, especially when used for offset approximation.

In a recent paper[6] on offset curve approximation, we suggested a new approach based on approximating the offset circle instead of the offset curve itself. To demonstrate the effectiveness of this approach, we compared it extensively with previous methods. To our surprise, Tiller and Hanson's simple method[7] outperforms other methods for offsetting (piecewise) quadratic curves, even though its performance is not as good for high-degree curves.

The experimental results revealed other interesting facts, too. Had these details been reported several years ago, we believe offset approximation research might have developed somewhat differently. This article is intended to fill an important gap in the literature. We conducted qualitative as well as quantitative comparisons employing various contemporary offset approximation methods for freeform curves in the plane. We measured the efficiency of the offset approximation in terms of the number of control points generated while making the approximations within a prescribed tolerance.

Offset of planar curves

Given a regular parametric curve, $C(t) = (x(t), y(t))$, in the plane its offset curve $C_d(t)$ by a constant radius d is defined by

$$C_d(t) = C(t) + d \cdot N(t) \quad \textbf{(1)}$$

where $N(t)$ is the unit normal vector of $C(t)$:

$$N(t) = \frac{(y'(t), -x'(t))}{\sqrt{x'(t)^2 + y'(t)^2}} \quad \textbf{(2)}$$

The regularity condition of $C(t)$ guarantees that $(x'(t), y'(t)) \neq (0, 0)$ and $N(t)$ is well defined on the curve $C(t)$. Equation 2 has a square root term in the denominator. Therefore, even if the given curve $C(t)$ is polynomial, its offset is not generally a polynomial or rational curve. This fundamental deficiency motivated the development of various polynomial and rational approximation techniques of $C_d(t)$. While the offset to a polynomial or rational parametric curve must be approximated, a close cousin of the offset, the evolute, is somewhat counterintuitively representable as a rational curve (see the sidebar "Evolute").

Most offset approximation techniques iteratively fit an approximation curve, measure its accuracy, and subdivide the problem if the approximation error exceeds the tolerance. This divide-and-conquer approach exploits the subdivision property of the base curve $C(t)$. Henceforth, we assume $C(t)$ is represented by a Bezier or nonuniform rational B-spline (NURBS) curve.

Denote by $C_d^a(t)$ the approximation of $C_d(t)$. Traditionally, the offset approximation error has been measured only at finite sample points along $C(t)$, computing $\varepsilon_i = ||C(t_i) - C_d^a(t_i)|| - d$. Elber and Cohen[8] proposed a symbolic method to compute the global error between squared distances:

$$\varepsilon(t) = \left\| C(t) - C_d^a(t) \right\|^2 - d^2 \quad \textbf{(3)}$$

The error function $\varepsilon(t)$ is obtained by symbolically computing the difference and inner product of Bezier or NURBS curves[9] (see the sidebar "Symbolic Computation"

0272-1716/97/$10.00 © 1997 IEEE

on the next page). Therefore, it can be represented as a Bezier or NURBS scalar function. As a scalar field, the largest coefficient of $\varepsilon(t)$ globally bounds the maximal possible error due to the convex hull property of Bezier or NURBS formulation. In this article, we exploit the error functional $\varepsilon(t)$ of Equation 3 to measure all the offset approximation errors. This provides not only a global bound for each method but an equal basis for comparing different methods.

Qualitative comparisons

We first compare how various offset approximation methods perform on freeform curves. Focusing on why certain techniques yield different levels of under- and overestimation, we suggest ways to alleviate these inaccuracies.

Control polygon-based methods

Let $C(t)$ be a B-spline curve with k control points of order n and defined over a knot sequence $\tau = \{t_i\}$, $0 \le i < n+k$. The ith node parameter value ξ_i of $C(t)$ is defined as

$$\xi_i = \frac{\sum_{j=i+1}^{i+n-1} t_j}{n-1} \qquad (7)$$

for $0 \le i < k$. Hence, a node parameter value is an average of $n-1$ consecutive knots in τ. Each control point P_i of $C(t)$ is associated with one node, ξ_i. $C(\xi_i)$ is typically close to P_i; in general, however, it is not the closest point of $C(t)$ to P_i.

Cobb[11] translated each control point P_i by $d \cdot N(\xi_i)$, whereas Tiller and Hanson[7] translated each edge of the control polygon into the edge normal direction by a distance d. Unfortunately, Cobb's method always underestimates the offset—that is, $\varepsilon(t) \le 0$—for all t. (For the proof and related issues, see the sidebar "Under- and Overestimation" on page 65.)

Tiller and Hanson's technique[7] did not underestimate the offset curve. In addition to computing the exact linear and circular offset curves, their method outperforms the other methods for offsetting (piecewise) quadratic curves. However, for offsetting high degree curves, this simple method performs about as well as Cobb's.[11]

Coquillart[12] solved the underestimation problem. Her method takes into account the distance between P_i and $C(\xi_i)$ and the curvature $\kappa(\xi_i)$ of $C(t)$ at ξ_i. Using numerical approximation, it computes the closest point of $C(t)$ to the control point P_i, using $C(\xi_i)$ as an initial solution. With these enhancements, Coquillart was able to offset the linear and circular segments exactly.

Elber and Cohen[13] took a different approach that exactly computes the offsets of linear and circular elements. Using the values of $\varepsilon(t)$ (in Equation 3) at $t = \xi_0, \ldots, \xi_n$, they estimated the error in the neighborhood of each control point P_i and used it to adjust the translational distance applied to P_i. This perturbation-based approach iteratively converges to the exact circular offset segment. For general curves, using Cobb's result[11] as an initial solution, the perturbation process typically reduces Cobb's offset approximation error by an order of magnitude. In principle, this method can be applied to any offset approximation method that produces piecewise polynomial curves.

Most traditional techniques subdivide $C(t)$ at the middle of the parametric domain; however, a better solution uses the parameter of the location with the maximum error. Since $\varepsilon(t)$ represents the exact squared error function, we can find the parameter location of the maximal error and subdivide $C(t)$ there. Alternatively, instead of subdividing $C(t)$, we can insert new knots into $C(t)$ at the parameter locations with errors larger than the allowed tolerance. Elber and Cohen[8] took this refinement approach.

Evolute

The evolute of $C(t)$ is defined as

$$E(t) = C(t) + \frac{N(t)}{\kappa(t)}$$

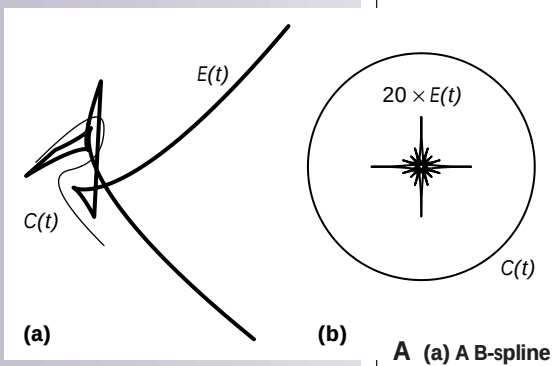
where $\kappa(t)$ is the curvature of $C(t) = (x(t), y(t))$:

$$\kappa(t) = \frac{x'(t)y''(t) - x''(t)y'(t)}{\left(x'(t)^2 + y'(t)^2\right)^{3/2}}$$

That is, $E(t)$ is a variable radius offset with offset radius $d(t) = 1/\kappa(t)$. Figure A shows two examples of evolute curves. Quite surprisingly, $E(t)$ is a rational curve, provided $C(t)$ is a rational or polynomial curve:

$$\begin{aligned} E(t) &= C(t) + \frac{N(t)}{\kappa(t)} \\ &= C(t) + \frac{\dfrac{\left(y'(t),\ -x'(t)\right)}{\left(x'(t)^2 + y'(t)^2\right)^{1/2}}}{\dfrac{x'(t)y''(t) - x''(t)y'(t)}{\left(x'(t)^2 + y'(t)^2\right)^{3/2}}} \\ &= C(t) + \frac{x'(t)^2 + y'(t)^2}{x'(t)y''(t) - x''(t)y'(t)} \cdot \left(y'(t),\ -x'(t)\right) \end{aligned}$$

In contrast to the offset computation in Equation 1, there is no square root term in the representation of $E(t)$. In Figure A, the curves and their evolutes are both represented as B-spline curves.



A (a) A B-spline curve $C(t)$, in light curve, is shown along with its evolute $E(t)$, in bold curve. (b) The evolute $E(t)$ for a cubic polynomial approximation of a circle $C(t)$, is shown. $E(t)$ is scaled up by a factor of 20.

Symbolic Computation

In this article, we employed $\varepsilon(t)$ (in Equation 3) and $\varepsilon_m(t)$ (in Equation 11) to estimate the offset approximation error. We also need to compute the composition of $U(s(t))$ (in Equation 10). The symbolic computation of these equations involves the difference, product, and sum of (piecewise) scalar polynomial or rational curves.

Let $C_1(t) = \sum_{i=0}^{m} P_i B_{i,1}(t)$ and $C_2(t) = \sum_{j=0}^{n} P_j B_{j,n}(t)$ be two (piecewise) polynomial regular parametric curves, in the Bezier or NURBS representations. The computation of $C_1(t) \pm C_2(t)$ can be accomplished by elevating both $C_1(t)$ and $C_2(t)$ to a common function space. The order of the common function space is equal to the maximal order of $C_1(t)$ and $C_2(t)$. If either $C_1(t)$ or $C_2(t)$ is a B-spline curve, the common function space is defined by considering both knot vectors τ and η and preserving the lowest degree of continuity at each knot. Once we determine the common function space, we elevate both $C_1(t)$ and $C_2(t)$ to this space via degree raising and refinement. (See Elber[9] and Farouki and Rajan[10] for more details and the extension to rationals.)

The computation of $C_1(t)C_2(t)$ is somewhat more involved. Here, we consider only the case of Bezier polynomial curves. (See Elber[9] for the more general cases of piecewise polynomials and rationals.) The ith Bernstein Bezier basis function of degree k is defined by

$$B_i^k(t) = \binom{k}{i} t^i (1-t)^{k-i} \tag{4}$$

The product of two Bernstein Bezier basis functions is

$$\begin{aligned} B_i^k(t)B_j^l(t) &= \binom{k}{i} t^i (1-t)^{k-i} \binom{l}{j} t^j (1-t)^{l-j} \\ &= \binom{k}{i}\binom{l}{j} t^{i+j} (1-t)^{k+l-i-j} \\ &= \frac{\binom{k}{i}\binom{l}{j}}{\binom{k+l}{i+j}} B_{i+j}^{k+l}(t) \end{aligned} \tag{5}$$

Therefore, we have

$$\begin{aligned} C_1(t)C_2(t) &= \sum_{i=0}^{m} P_i B_i^m(t) \sum_{j=0}^{n} P_j B_j^n(t) \\ &= \sum_{i=0}^{m}\sum_{j=0}^{n} P_i P_j B_i^m(t) B_j^n(t) \\ &= \sum_{i=0}^{m}\sum_{j=0}^{n} P_i P_j \frac{\binom{m}{i}\binom{n}{j}}{\binom{m+n}{i+j}} B_{i+j}^{m+n}(t) \\ &= \sum_{k=0}^{m+n} Q_k B_k^{m+n}(t) \end{aligned} \tag{6}$$

where Q_k accumulates all the combinatorial terms

$$P_i P_j \frac{\binom{m}{i}\binom{n}{j}}{\binom{m+n}{i+j}}$$

for $k = i + j$. Hence, $C_1(t)C_2(t)$ is represented as a Bezier polynomial curve of degree $m + n$.

Interpolation methods

Klass[14] used a cubic Hermite curve to approximate the offset curve. He determined the cubic Hermite curve by interpolating the position and velocity of the exact offset curve at both endpoints. Klass' numerical approximation procedure is quite unstable when the offset curve becomes almost flat. Therefore, instead of using the original algorithm, we compute the first derivative of the offset curve based on the following simple closed form equation (see also Farouki and Neff[3]):

$$C_d'(t) = (1 + d \cdot \kappa(t)) C'(t) \tag{8}$$

where $k(t)$ is the curvature of $C(t)$.

Hoschek[1] suggested a least-squares solution for determining $C_d'(t)$ at the curve endpoints. That is, at each endpoint of $C_d(t)$, the direction of $C_d'(t)$ is maintained parallel to $C'(t)$; however, instead of using Equation 8, their lengths are determined so that the cubic Hermite curve best fits $C_d(t)$ in the least-squares sense. For computational efficiency, the optimization uses only finite samples of $C_d(t)$.

Hoschek and Wissel[2] used a general nonlinear optimization technique to approximate a high-degree spline curve with low-degree spline curves. They applied the same technique to approximate an exact offset curve with low-degree spline curves.

We would expect the least-squares-based methods[1,2] to perform better than other methods. However, a question remains about whether the least-squares solution is optimal when searching for the smallest number of (say, cubic) curve segments to approximate an exact offset curve. In general, it is not. In the special case of offsetting quadratic curves, Tiller and Hanson's simple method[7] performs much better than the least-squares methods.[1,2]

It is important to ask how we got this unexpected result. The answer could prove useful in improving the accuracy of offset approximation. The least-squares solution optimizes the integrated summation of the least-squares errors in the approximation. Therefore, even if a small portion of the approximation curve has a large error, as long as the rest of the curve tightly approximates the exact curve, the overall least-squares error can be very small. That is, the least-squares solution provides an optimal solution with respect to an L_2 norm. Further evaluating this L_2 optimal solution with respect to the L_∞ norm (of Equation 3), we find the optimality is no longer guaranteed. This important observation suggests possible improvements over the nearly optimal solutions.[1,2]

Pham[15] suggested a simple B-spline interpolation method to approximate the offset curve. Finite sample points are generated on the exact offset curve and interpolated by a piecewise cubic B-spline curve. This simple method also performs pretty well. In many examples, its

Under- and Overestimation

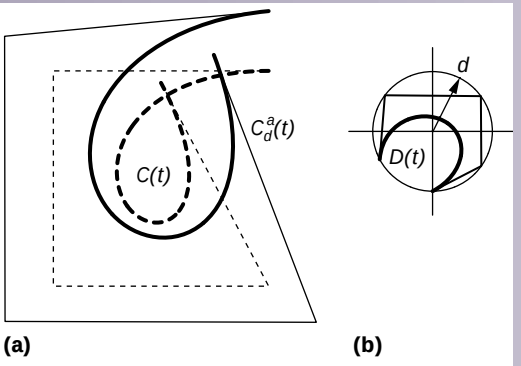
Cobb's offset approximation[11] is formally defined as follows:

$$C_d^a(t) = \sum_{i=0}^{n} \left(P_i + dN(\xi_i)\right) B_i(t) = \sum_{i=0}^{n} P_i B_i(t) + d\sum_{i=0}^{n} N(\xi_i)\, B_i(t) = C(t) + dV(t)$$

where $\|N(\xi)\| = 1$ for $i = 0, \ldots, n$. The vector field curve $V(t) = \sum_{i=0}^{n} N(\xi_i) B_i(t)$ has all its control points $N(\xi_i)$ on the unit circle S^1. By the convex hull property, we have $\|V(t)\| \le 1$ and

$$\left\|C(t) - C_d^a(t)\right\| = \left\|dV(t)\right\| = d\left\|V(t)\right\| \le d$$

If $N(\xi_i) \neq N(\xi_j)$, for some $0 \le i, j \le n$, we have $\min \|V(t)\| < 1$, and this results in an error in the offset approximation. Hence, this method always underestimates the exact offset. Figure B shows a quartic Bezier curve $C(t)$ and its offset approximation $C_d^a(t)$. The difference vector field $D(t) = C(t) - C_d^a(t)$ is completely contained in a disk of radius d. All the control points of $D(t)$ lie on the circumference of the disk.



B (a) An offset approximation $C_d^a(t)$ computed by translating the control points of the original curve $C(t)$ (dashed lines) by an amount equal to the offset distance will always underestimate the real offset. (b) $D(t) = C(t) - C_d^a(t)$ is found to be fully contained in a circle of the offset radius size, d.

Underestimating offsets may lead to undesirable results. For example, in numerically controlled (NC) machining, the underestimation leads to gouging. Assume the underestimation of the offset is bounded from below by

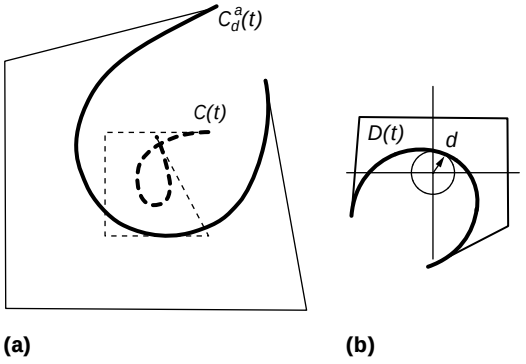
$$d_{\min} = \min(\|dV(t)\|).$$

When we translate control point P_i in the direction of $N(\xi_i)$ by a distance $d^2/d_{\min}$, the resulting curve completely overestimates the exact offset (see Figure C):

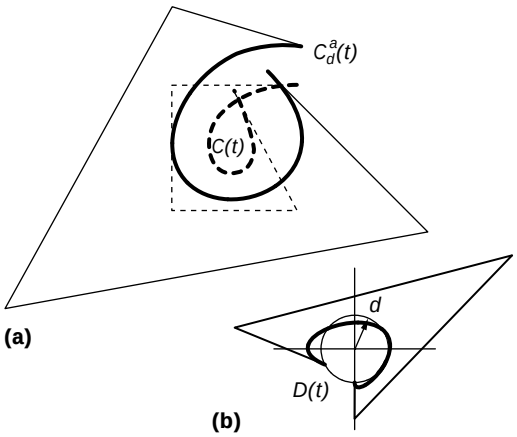
$$\left\|C(t) - C_d^a(t)\right\| = \left\|d\,\frac{d}{d_{\min}}\,V(t)\right\| \ge \left\|d\,\frac{d}{\|dV(t)\|}\,V(t)\right\| = d$$

We can reduce the relative error in the offset approximation by alternating the under- and overestimations. We do this by adjusting the offset distance at each control point appropriately.

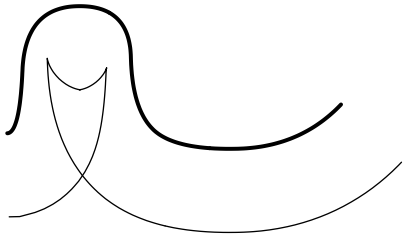
Figure D shows an example of this approach. We use the same quartic Bezier curve as in Figures B and C. The quartic Bezier offset approximation curve interpolates the exact offset at five discrete locations, corresponding to the node values $i/4$, $0 \le i \le 4$.



C (a) An offset approximation $C_d^a(t)$ of a quartic Bezier curve $C(t)$ (dashed lines) is computed by forcing $C_d^a(t)$ to overestimate the error. (b) $D(t) = C(t) - C_d^a(t)$. Compare with Figure B.

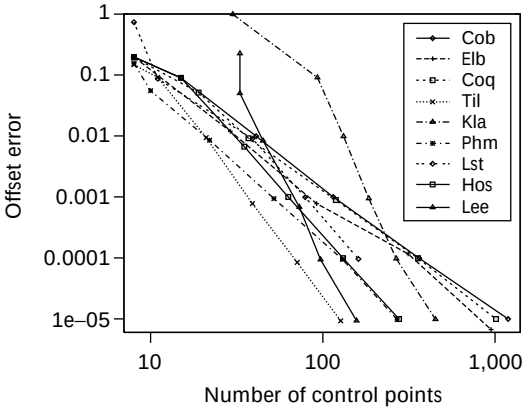


D (a) An offset approximation $C_d^a(t)$ of a quartic Bezier curve $C(t)$ (dashed lines) is computed by forcing $C_d^a(t)$ to interpolate at five locations on the exact offset curve computed at the node values on $C_d^a(t)$. (b) $D(t) = C(t) - C_d^a(t)$. Compare with Figures B and C.



ε	Cob	Elb	Coq	Til	Kla	Phm	Lst	Hos	Lee
10^{-0}	8	8	8	8	30	8	8	8	33
10^{-1}	15	8	19	11	93	10	11	15	33
10^{-2}	41	29	37	21	132	22	39	35	45
10^{-3}	115	92	119	39	185	52	79	63	73
10^{-4}	363	313	357	71	267	130	161	131	97
10^{-5}	1191	948	1013	127	451	270	332	277	157

1 An offset approximation (light curve) for a quadratic polynomial B-spline curve with eight control points.



performance is only slightly worse than and sometimes even better than the local least-squares methods.[1,2]

Circle approximation methods

Assume the base curve $C(t)$, $t_0 \le t \le t_1$, is a polynomial curve with no inflection point, and a unit circular arc $U(s)$, $s_0 \le s \le s_1$, is parameterized so that

$$C'(t_0) \parallel U'(s_0) \text{ and } C'(t_1) \parallel U'(s_1)$$

If we compute a reparameterization $s(t)$ so that

$$C'(t) \parallel U'(s(t))$$

we can then compute the offset curve as

$$C_d(t) = C(t) + d \cdot U(s(t)) \qquad \textbf{(9)}$$

The offset curve is not a polynomial or rational curve; therefore, we have to approximate $U(s)$ and/or $s(t)$ by a polynomial or rational.

Lee et al.[6] approximated the unit circle $U(s)$ with piecewise quadratic polynomial curve segments $Q_j(s)$, $j = 0, \ldots, n$. The Hodograph curve $Q_j'(s)$ is piecewise linear; therefore, the parallel constraint

$$C'(t) \parallel Q'(s(t))$$

provides the reparameterization of $s(t)$ as a rational polynomial of degree $d-1$, where d is the degree of $C(t)$. For a polynomial curve $C(t)$ of degree d, the resulting offset approximation (computed as in Equation 9) is a rational curve of degree $3d-2$. (For a rational curve $C(t)$ of degree d, the offset approximation curve is of degree $5d-4$.)

For a quadratic polynomial curve $C(t)$, this technique also provides a simple method to represent the exact offset curve $C_d(t)$ as a rational curve of degree six. Assume that the exact circle $Q(s)$, $0 \le s \le 1$, is represented by a rational quadratic curve. Then the parallel constraint

$$C'(t(s)) \parallel Q'(s)$$

provides the reparameterization of $t(s)$ as a rational polynomial of degree two. Therefore, the exact offset curve $C_d(t)$ is a rational curve of degree six. Even with the high degree of six, the exact offset capability suggests this as the method of choice for offsetting (piecewise) quadratic polynomial curves, especially for high-precision offset approximation. However, this exact offset capability does not extend to rational quadratic curves. (Some rational quadratic curves have no exact rational parametrization of their offset curves.)

We can attempt to globally approximate $s(t)$ by maximizing the constraint energy

$$\max_{s(t)} \int_{t_0}^{t_1} \frac{\langle C'(t), U'(s(t)) \rangle}{\|C'(t)\| \, \|U'(s(t))\|} dt \qquad \textbf{(10)}$$

Lee et al.[16] took this approach, in which the composition of $U(s(t)) = (U \circ s)(t)$ is carried out symbolically[9] (see also the sidebar "Symbolic Computation").

The offset approximation of Lee et al.[6] depends on the method used for the piecewise quadratic approximation to the circle. The error in the offset approximation stems only from the quadratic polynomial approximation of the circular arc, scaled by the offset radius d. Lee et al. used five different circle approximation methods.[6] Two of the five methods generate G^1-continuous circle approximations with quadratic Bezier curve segments. In the first method, the unit circle $U(s)$ is totally contained in the closed convex region bounded by the quadratic curve segments. The corresponding offset curve approximation completely overestimates the exact offset curve. In the second method, the quadratic curve segments pass through both the interior and exterior of the unit circle $U(s)$. Therefore, the offset approximation curve both under- and overestimates the exact offset curve, while the approximation error is reduced by half from the overestimation of the first method. We use this second method, referred to as Lee in the next section, for comparison with other methods.

In contrast, Lee et al.[16] approximated the reparametrization $s(t)$, while representing the circle $U(s)$ exactly by a rational quadratic curve. In this method, the error stems only from the inaccurate reparameterization function $s(t)$, which results in a mismatch in the parallel constraint of $C'(t) \parallel U'(s(t))$. To our knowledge, this is the only offset approximation method for which the use of

$\varepsilon(t)$ is completely ineffective in the global error bound. The term $\varepsilon(t)$ always equals zero. Lee et al.[16] measured the angular deviation of $U(s(t))$ from the exact offset direction $N(t)$ by using the following error function:

$$\varepsilon_m(t) = \frac{\left\langle C(t) - C_d^a(t), C'(t) \right\rangle^2}{d^2 \left\| C'(t) \right\|^2} \quad \textbf{(11)}$$

The error equals zero if orthogonality is preserved. Otherwise, it equals $\cos^2\theta$, where θ is the angle between $U(s(t))$ and $C'(t)$.

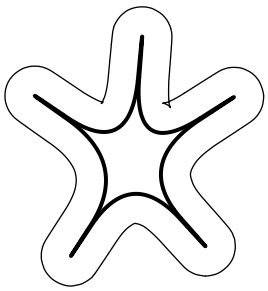
Quantitative comparisons

We consider here how efficiently each method approximates the offset curve given a prescribed tolerance. We give several examples of Bezier and B-spline curves, both in polynomial and rational forms. All methods compared in this article are implemented using the Irit[17] solid modeling system developed at Technion, with some of the offset approximation methods implemented at Postech.

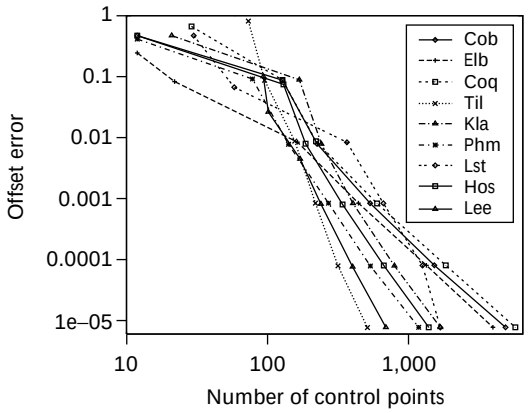
Methods under comparison

We quantitatively compare the following methods:

- *Cob*: Cobb's simple method[11] in which the control points are translated by the offset distance. This method always creates underestimated offsets. (See the sidebar "Under- and Overestimation.")
- *Elb*: An adaptive offset refinement approach suggested in Elber and Cohen.[8] Instead of subdividing the base curve, whenever the error is too large, the offset curve is refined to yield a better approximation (by using more control points). The error analysis of $\varepsilon(t)$ is exploited to find better candidate locations for refinement. This method also underestimates the offset curves.
- *Coq*: The enhancement suggested by Coquillart[12] that allows the exact offset representation of linear as well as circular segments.
- *Til*: Tiller and Hanson's method,[7] which translates the edges of the control polygon rather than the control points.
- *Klass*: Klass's method[14] that fits a cubic Bezier curve to each offset curve segment to interpolate the boundary points and velocities of the exact offset curve.
- *Pham*: Pham's method[15] that interpolates a sequence of finite sample points on the exact offset curve by a nonuniform piecewise cubic B-spline curve. (Pham's original method uses a uniform B-spline curve; we modified it, however.) When the offset approximation error exceeds the prescribed tolerance, more sample offset points give a better fit.
- *Lst*: The global least-squares approximation that fits a uniform piecewise cubic B-spline curve to the offset curve. Again, when the offset approximation error exceeds the prescribed tolerance, more sample offset points give a better fit.
- *Hos*: The least-squares method of Hoschek[1,2] that fits a cubic Bezier curve to each offset curve segment. When the error exceeds the tolerance, the base curve is subdivided into two subsegments and the offset approximation is repeated recursively.
- *Lee*: The approach suggested by Lee et al.[6] that approximates the curve of the convolution between $C(t)$ and the offset circle $d \cdot U(s)$ of radius d.



ε	Cob	Elb	Coq	Til	Kla	Phm	Lst	Hos	Lee
10^{-0}	12	12	29	73	21	12	30	12	93
10^{-1}	127	22	127	95	167	78	58	129	101
10^{-2}	223	162	219	155	239	140	362	185	169
10^{-3}	527	436	591	217	399	268	654	337	237
10^{-4}	1497	1316	1803	313	783	530	1234	659	397
10^{-5}	4763	3878	5587	505	1651	1162	1650	1367	681

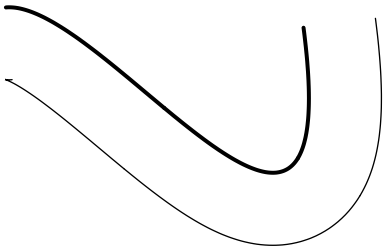


2 An offset approximation (light curve) for a quadratic polynomial with sharp corners.

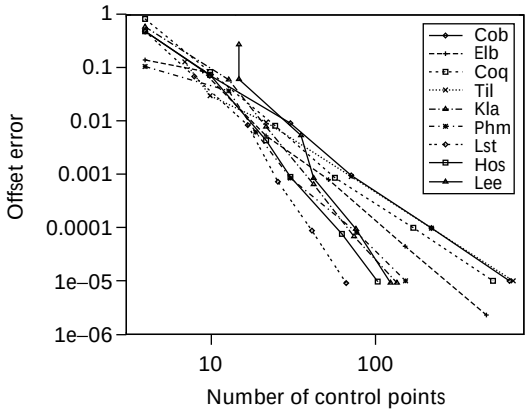
Traditionally, the offset approximation error has been measured only at finite sample points of $C(t)$ and $C_d^a(t)$. As mentioned earlier, we adopt the symbolic approach of error estimation.[8] Therefore, we can provide an L_∞ global upper bound on the offset approximation error for each method under comparison. The global error bound is derived by symbolically computing the error function $\varepsilon(t)$ (in Equation 3). Because of the convex hull property of the Bezier or NURBS representation of the scalar function $\varepsilon(t)$, we can easily determine its upper bound as the maximum coefficient of the Bezier or NURBS basis functions.

Comparison results and remarks

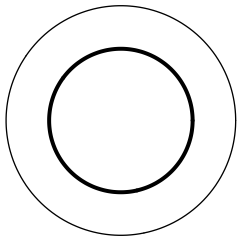
Figure 1 and Figure 2 show the results of offsetting (piecewise) quadratic curves. We compare the number of control points with respect to the accuracy of offset approximation. In these examples, Tiller and Hanson's method[7] outperforms the other methods even if the base curve has sharp corners with high curvature (Figure 2). This surprising result has never been reported in the lit-



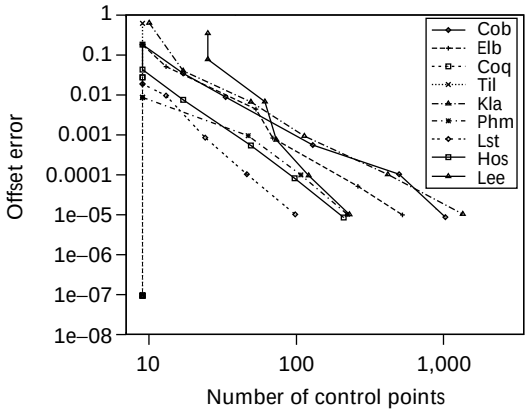
ε	Cob	Elb	Coq	Til	Kla	Phm	Lst	Hos	Lee
10^{-0}	4	4	4	7	4	4	4	4	15
10^{-1}	10	9	10	10	13	13	8	10	15
10^{-2}	31	22	25	22	22	19	17	22	36
10^{-3}	73	53	58	73	43	31	26	31	43
10^{-4}	226	156	175	226	76	79	42	64	78
10^{-5}	679	490	538	715	139	157	68	106	127



3 An offset approximation (light curve) for a polynomial cubic Bezier curve.



ε	Cob	Elb	Coq	Til	Kla	Phm	Lst	Hos	Lee
10^{-0}	9	9	9	9	10	9	9	9	25
10^{-1}	17	13	9	9	17	9	9	9	25
10^{-2}	33	53	9	9	49	9	13	17	61
10^{-3}	129	69	9	9	113	47	24	49	73
10^{-4}	497	261	9	9	417	107	46	97	121
10^{-5}	1025	524	9	9	1345	221	98	209	229



4 An offset approximation (light curve) for a rational quadratic B-spline circle with nine control points.

erature. In fact, we assumed that the least-squares methods provide near-optimal solutions to the offset approximation problem. However, the superior performance of Tiller and Hanson[7] tells us that this is not true in general.

At this moment, we have no clear explanation of the underlying geometric properties of this unusual phenomenon. Nevertheless, we can point to at least two possible sources of the nonoptimality in the current least-squares methods:

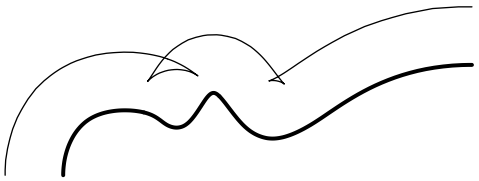
- As discussed above, the least-squares methods provide the optimal solutions in an L_2 norm, which may be quite different from the optimal solutions in an L_∞ norm.
- The least-squares optimization procedure solves an overconstrained problem, the solution of which depends on the distribution of finite sample points on the offset curve. In some degenerate cases, the least-squares solution may have large variation depending on the distribution of data points.

Further investigations should help eliminate these limitations, likely advancing the state of the art of offset curve approximation.

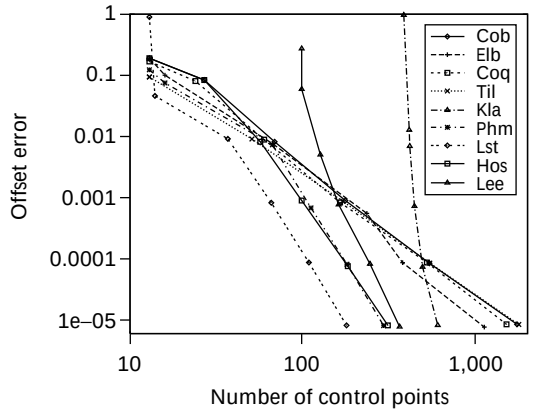
Figures 3 through 6 show other examples of offsetting (piecewise) quadratic and cubic B-spline curves. Throughout our tests we observed the following consistent results:

- The underestimating offset approximation method, *Cob*, performs quite poorly.
- The adaptive offset refinement approach, *Elb*, works better than *Cob*, especially when high precision is desired.
- For offsetting (piecewise) quadratic curve segments, Tiller and Hanson's method outperforms the others, especially when high precision is required.
- For offsetting (piecewise) cubic curve segments, the least-squares methods, *Lst* and *Hos*, perform much better than the other methods, especially when high precision is required.
- In many examples, the local cubic B-spline interpolation method, *Pham*, performs as well as—and sometimes even better than—*Hos*. However, its performance deteriorates when the base curve has a radius of curvature similar to the offset radius.
- The only geometrical method that approaches the efficiency of the least-squares methods is *Lee*, followed not so closely by *Elb*.

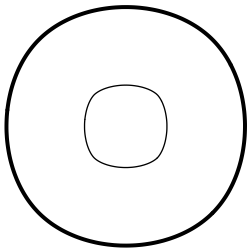
For the case of offsetting (piecewise) cubic curves,



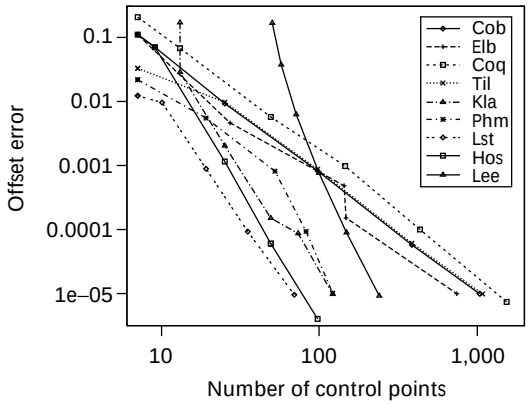
ε	Cob	Elb	Coq	Til	Kla	Phm	Lst	Hos	Lee
10^{-0}	13	13	13	13	386	13	13	13	99
10^{-1}	27	16	24	13	415	16	14	27	99
10^{-2}	69	67	60	51	418	67	37	57	127
10^{-3}	177	236	165	171	445	112	66	99	162
10^{-4}	543	380	525	546	496	181	109	183	246
10^{-5}	1746	1129	1518	1788	607	295	180	312	365



5 An offset approximation (light curve) for a cubic polynomial B-spline curve with 13 control points.



ε	Cob	Elb	Coq	Til	Kla	Phm	Lst	Hos	Lee
10^{-0}	7	7	7	7	13	7	7	7	50
10^{-1}	9	7	13	7	13	7	7	9	57
10^{-2}	25	27	49	25	25	19	10	25	71
10^{-3}	97	143	145	97	49	52	19	49	99
10^{-4}	385	147	453	385	73	82	35	49	148
10^{-5}	1033	742	1537	1081	121	121	69	97	239



6 An offset approximation (light curve) for a cubic polynomial periodic B-spline curve with four control points.

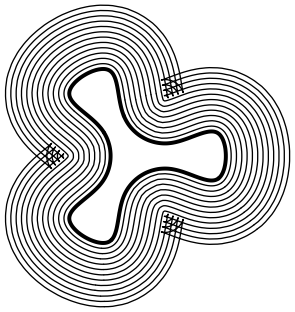
the global least-squares method, *Lst*, outperforms all the other methods, closely followed by the local least-squares method, *Hos*, and by the local cubic B-spline interpolation method, *Pham*. Many practical situations require the production of local optimal solutions based only on the available local data. For example, for data storage, we can store only the subdivision locations of the curve, not all control points generated. We then use the local methods to generate the control points (on the fly) by considering only local data. In this case, *Hos* and *Pham* are the methods of choice.

As discussed above, *Pham*'s performance depends on the radius of curvature of the base curve. When the radius of curvature is similar to the offset radius, the sample offset points cluster together. The B-spline interpolation of these clustered points generates undulation, which is the main source of large approximation error. In this case, it is better to use fewer data points for the interpolation. Figure 7 and Figure 8 exemplify this phenomenon by comparing the relative performances of different offset approximation methods. Given a fixed base curve, by increasing the offset radius gradually, we can observe that Pham's method has the worst relative performance near the offset distance, which starts to develop cusps in the offset curve.

We should consider another source of undulation in Pham's method. The mismatch in speeds between the two curves (the base curve and the offset curve) also causes deterioration in the quality of the offset approximation. To implement *Pham*, we used a nonuniform cubic B-spline curve in which the data points of the offset inherit the knot values of the base curve points. When the offset data points cluster, their knot values are much sparser compared with the offset curve length. This unnatural assignment of knot values generates undulation. Therefore, for a better offset approximation, we must also rearrange the knot values of the offset data points.

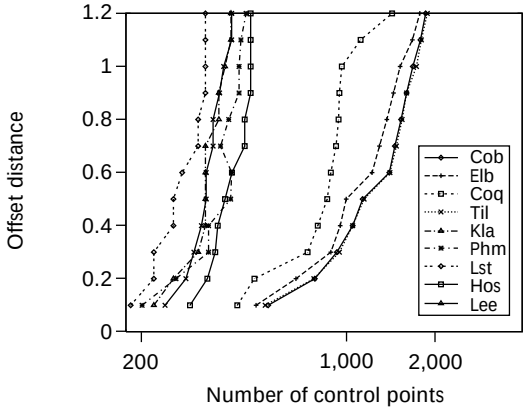
Til's superior performance (in the quadratic case) suggests the possibility of improving the current least-squares methods by resolving their limitations as discussed above. The limitation resulting from the L_2 norm seems more serious. To resolve this problem, we need to develop an efficient algorithm to compute and optimize the L_∞ norm of the offset approximation error; that is, the maximum of the error function $\varepsilon(t)$ (in Equation 3):

$$\max_t \left\{ \left\| C(t) - C_d^q(t) \right\|^2 - d^2 \right\} \qquad \textbf{(12)}$$



d	Cob	Elb	Coq	Til	Kla	Phm	Lst	Hos	Lee
0.1	531	483	417	519	217	198	181	237	288
0.2	765	660	477	759	253	258	217	279	330
0.3	909	867	723	927	307	333	217	297	351
0.4	1029	933	783	1029	325	333	253	315	358
0.5	1113	978	843	1125	325	396	253	327	379
0.6	1371	1197	867	1377	325	399	271	327	400
0.7	1431	1269	903	1449	325	366	307	345	442
0.8	1503	1344	921	1515	361	390	307	345	442
0.9	1569	1416	927	1569	361	423	325	363	463
1.0	1647	1491	945	1695	379	423	325	375	463
1.1	1749	1641	1095	1761	397	429	325	399	463
1.2	1821	1740	1401	1845	397	447	325	399	463

7 Pham's method has the worst relative performance for the offset distances between 0.4 and 0.6. The base curve is a cubic B-spline curve. A tolerance of 0.0001 is used for offset approximation error.

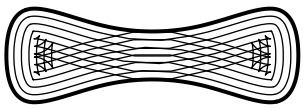


or a more precise geometric distance measure based on the following Hausdorff metric:

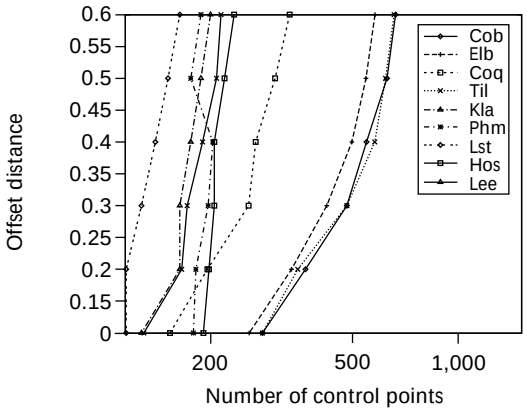
$$\max\left(\max_t \min_s\left\{\left\|C(s)-C_d^a(t)\right\|^2-d^2\right\},\ \max_s \min_t\left\{\left\|C(s)-C_d^a(t)\right\|^2-d^2\right\}\right) \quad \textbf{(13)}$$

where s is assumed to be a local perturbation of the parameter t.

Note that Cobb's method[11] essentially models the L_∞ norm of Equation 12 in terms of the maximum and minimum magnitudes of the distance curve, $D(t) = C(t) - C_d^a(t)$, in the sidebar Figures B, C, and D. Let's consider a variant of *Cobb* that uses the least-squares technique to optimize the offset distance at each control point so that the distance curve $D(t)$ is a best fit to the offset circle of radius d. This method measures the offset error in the L_∞ sense of Equation 12. (Note that the approximation of $D(t)$ to an offset circle still has the limitation of L_2 norm.) Although we have not provided all the details in this article, Lee et al.'s method[6] actually measures the offset approximation error under the L_∞ norm of Equation 13, which is more precise than the L_∞ norm of Equation 12. We expect that future offset approximation techniques (while incorporating these L_∞ norms into their optimization procedures) may provide more accurate results than the current least-squares methods.



d	Cob	Elb	Coq	Til	Kla	Phm	Lst	Hos	Lee
0.1	279	256	153	279	127	178	115	129	190
0.2	369	337	195	351	163	181	115	165	197
0.3	483	424	255	483	163	196	127	171	204
0.4	549	499	267	579	175	202	139	189	204
0.5	627	547	303	621	187	175	151	207	218
0.6	663	580	333	657	199	187	163	213	232



8 Pham's method has the worst performance for the offset distances around 0.4. The base curve is a cubic B-spline curve. A tolerance of 0.001 is used for offset approximation error.

Conclusion

Comparing several contemporary offset approximation techniques for freeform curves in the plane shows that, in general, the least-squares methods perform very well. However, for the case of offsetting quadratic curves, the simple method of Tiller and Hanson[7] is the method of choice. Therefore, the least-squares methods need further improvement to produce near-optimal solutions in all cases. Some of the current methods[6,8,11] have geometric representations of the offset approximation error (in certain L_∞ norms), whereas none of the current least-squares methods have such geometric interpretation of their respective error bounds. We also pointed out two limitations of the current least-squares methods: the L_2 norm they employ and their dependency on the finite sample points used in the optimization.

The B-spline interpolation method also needs further investigation to eliminate the curve undulation resulting from the curve speed mismatch between the base curve and the offset curve. In this respect, there are still many ways to improve the current state of the art of offset curve approximation. We hope that the experimental results reported here and the related remarks will serve as useful guidelines for future research. ■

Acknowledgment

This work was supported in part by the Fund for Promotion of Research at Technion, Haifa, Israel.

References

1. J. Hoschek, "Spline Approximation of Offset Curves," *Computer Aided Geometric Design*, Vol. 5, No. 1, June 1988, pp. 33-40.
2. J. Hoschek and N. Wissel, "Optimal Approximate Conversion of Spline Curves and Spline Approximation of Offset Curves," *Computer-Aided Design*, Vol. 20, No. 8, Oct. 1988, pp. 475-483.
3. R. Farouki and C. Neff, "Analytic Properties of Plane Offset Curves," *Computer Aided Geometric Design*, Vol. 7, 1990, pp. 83-99.
4. R. Farouki and C. Neff, "Algebraic Properties of Plane Offset Curves," *Computer Aided Geometric Design*, Vol. 7, 1990, pp. 101-127.
5. R. Farouki and T. Sakkalis, "Pythagorean Hodograph," *IBM J. Research and Development*, Vol. 34, 1990, pp. 736-752.
6. I.-K. Lee, M.-S. Kim, and G. Elber, "Planar Curve Offset Based on Circle Approximation," *Computer-Aided Design*, Vol. 28, No. 8, Aug. 1996, pp. 617-630.
7. W. Tiller and E. Hanson, "Offsets of Two Dimensional Profiles," *IEEE Computer Graphics and Applications*, Vol. 4, No. 9, Sept. 1984, pp. 36-46.
8. G. Elber and E. Cohen, "Error Bounded Variable Distance Offset Operator for Free Form Curves and Surfaces," *Int'l J. Computational Geometry and Applications*, Vol. 1, No. 1, Mar. 1991, pp. 67-78.
9. G. Elber, *Free Form Surface Analysis Using a Hybrid of Symbolic and Numeric Computation*, PhD dissertation, Computer Science Department, University of Utah, Salt Lake City, Utah, 1992.
10. R.T. Farouki and V.T. Rajan, "Algorithms For Polynomials in Bernstein Form," *Computer Aided Geometric Design*, Vol. 5, No. 1, June 1988, pp. 1-26.
11. B. Cobb, *Design of Sculptured Surfaces Using the B-Spline Representation*, PhD dissertation, Computer Science Dept., University of Utah, Salt Lake City, Utah, 1984.
12. S. Coquillart, "Computing Offset of B-Spline Curves," *Computer-Aided Design*, Vol. 19, No. 6, July/Aug. 1987, pp. 305-309.
13. G. Elber and E. Cohen, "Offset Approximation Improvement by Control Points Perturbation," *Mathematical Methods in Computer Aided Geometric Design II*, T. Lyche and L.L. Schumaker, eds., Academic Press, New York, 1992, pp. 229-237.
14. R. Klass, "An Offset Spline Approximation for Plane Cubic Splines," *Computer-Aided Design*, Vol. 15, No. 5, Sept. 1983, pp. 297-299.
15. B. Pham, "Offset Approximation of Uniform B-Splines," *Computer-Aided Design*, Vol. 20, No. 8, Oct. 1988, pp. 471-474.
16. I.-K. Lee, M.-S. Kim, and G. Elber, "New Approximation Methods for Planar Curve Offset and Convolution Curves," to appear in *Theory and Practice of Geometric Modeling*, W. Strasser, ed., Springer-Verlag, Heidelberg, 1997.
17. *IRIT 6.0 User's Manual*, Technion, Haifa, Israel, Feb. 1996, http://www.cs.technion.ac.il/~irit.

Gershon Elber *is a senior lecturer in the Computer Science Department at Technion in Haifa, Israel. He received a BS in computer engineering and an MS in computer science from Technion in 1986 and 1987, respectively, and a PhD in computer science from the University of Utah in 1992. His interests include computer-aided geometric design and computer graphics. He is a member of ACM and IEEE.*

In-Kwon Lee *is a postdoctoral researcher in computer science at Postech, Korea. He received a BS from Yonsei University, Korea in 1989 and an MS and PhD from Postech in 1992 and 1997, respectively, all in computer science. His research interests include computer graphics and computer-aided geometric design.*

Myung-Soo Kim *is an associate professor in computer science at Postech. He received a BS and MS in mathematics from Seoul National University in 1980 and 1982, respectively. He also received MS degrees in applied mathematics (1985) and computer science (1987) from Purdue University, where he completed his PhD in computer science in 1988. His research interests include computer graphics, computer animation, and geometric modeling.*

Contact Elber at the Department of Computer Science, Technion, Israel Institute of Technology, Haifa 32000, Israel, e-mail gershon@cs.technion.ac.il