# Interpolation in Digital Modems—Part II: Implementation and Performance

Lars Erup, *Member, IEEE,* Floyd M. Gardner, *Fellow, IEEE,* and Robert A. Harris, *Member, IEEE*

***Abstract*— An earlier paper introduced the theoretical background for digital interpolation in a data receiver and described an interpolator control method. Here we report properties of a specific class of interpolators that are based upon polynomials. Several implementations are described, one of which is particularly convenient in practical hardware.**

**Simulations demonstrate that simple interpolators give excellent performance. In many cases, two-point, linear interpolation is adequate. If better performance is needed, classical four-point, third-order polynomials could be used. Better yet, a novel four-point interpolating filter with piecewise-parabolic impulse response can have performance superior to that of the standard cubic interpolator and still be implemented much more simply.**

**The NCO-based control method presented in Part I is shown to be equivalent to a conventional phase locked loop and its operation is verified by simulation.**

## I. Introduction

PART I of this work [1] presented a fundamental equation for digital interpolation of data signals:

$$\begin{aligned} y(kT_i) &= y[(m_k + \mu_k)T_s] \\ &= \sum_{i=I_1}^{I_2} x[(m_k - i)T_s]\ h_I[(i + \mu_k)T_s] \end{aligned} \qquad (1)$$

where $\{x(m)\}$ is a sequence of signal samples taken at intervals $T_s$, and $h_I(t)$ is the finite-duration impulse response of a fictitious, time-continuous, analog interpolating filter. Digital operations in (1) deliver interpolants $y(k)$ at adjustable intervals $T_i$; $T_i$ is in general incommensurate with $T_s$.

Parameters in (1) are: the filter index $i = I_1$ to $I_2$, the basepoint index $m_k$ (which identifies the $I = I_2 - I_1 + 1$ signal samples to be used for the $k$th interpolant), and the fractional interval $\mu_k$ (which identifies the $I$ filter coefficients to be employed in (1) for the $k$th interpolant).

Part I established desirable characteristics of the interpolating filter, within broad limits; it remains to evaluate and choose specific filter responses for application to practical modems. This paper, in Section II, explores properties of simple polynomial-based filters — a category that includes the classical interpolating polynomials.

Several digital structures for performing the computations in (1) have been considered, and are discussed in Section III.

Paper approved by U. Mengali, the Editor for Synchronization Systems and Techniques of the IEEE Communications Society. Manuscript received August 27, 1991; revised February 6, 1992.

L. Erup and R. A. Harris are with European Space Agency, ESTEC/TST, P. O. Box 299, Noordwijk, NL–2200 AG, The Netherlands.

F. M. Gardner is with Gardner Research Company, Palo Alto, CA 94301.

IEEE Log 9208035.

Simulations have been run on several candidate filters to evaluate their effect on modem performance, with results given in Section IV. We have found that extremely simple interpolators can be employed with negligible impact on modem performance.

The design of a complete timing control loop, based on the NCO method devised in [1], is illustrated in Section V. It is shown how familiar techniques used to design phase-locked loops can be employed directly.

## II. Polynomial-Based Filters

There is an infinite variety of functions that interpolation filters could be based upon. One such class of functions is that of polynomial-based filters, in which the impulse response $h_I(t)$ of the continuous underlying filter [1] is a polynomial, or piecewise polynomial, in $t$ (or its surrogate $\mu_k$). This polynomial impulse response is completely distinct from, and should not be confused with, the associated *approximating* or *interpolating polynomial* discussed below.

Polynomial-based filters are not inherently optimum; why should they be given serious attention? There are several reasons:

- An extensive literature exists on polynomial interpolation, which is a small but important subclass of polynomial-based filters.
- They are easy to describe.
- Good filter characteristics can be achieved.
- A special FIR structure (to be described below) permits simple handling of filter coefficients. The structure is applicable only to polynomials.

Associated with every polynomial-based filter is a time-continuous *approximating polynomial* $p_k(t)$, that approximates $x(t)$ in the vicinity of $t = kT_i$. The interpolating filter of (1) computes samples $y(kT_i) = p_k(kT_i)$ of this polynomial. The approximating polynomial is different for each $I$-point basepoint set $\{x(m)\}$. It is not necessary that $p_k(t)$ be a good approximation to $x(t)$; a modem's interpolator is allowed to perform filtering on the signal. For the most part, there is no need to have explicit analytical knowledge of the approximating polynomial [2].

### A. Interpolating Polynomials

As a special case, if $p_k(mT_s) = x(mT_s)$ for all $I$ points of the $k$th basepoint set, and for all $k$, then $p_k(t)$ is said to be an *interpolating polynomial.* This case includes all

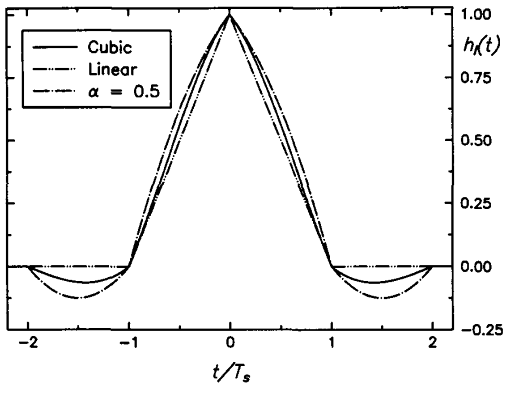

Fig. 1. Impulse responses of selected interpolating filters.

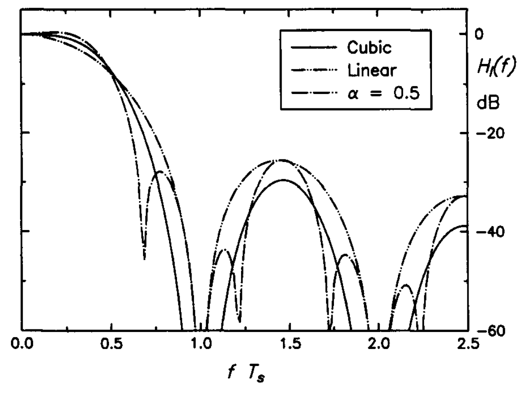

Fig. 2. Frequency responses of selected interpolating filters

of the classical polynomial interpolation formulas found in mathematics textbooks.

Any classical polynomial interpolation can be described in terms of its Lagrange coefficients (see Appendix, any numerical-analysis textbook, or [3]). The formulas for the coefficients are themselves polynomials in $t$, or equivalently $\mu_k$, of degree $I - 1$ with one (unique) piecewise formula per interval between basepoints. The Lagrange coefficient formulas constitute the filter impulse response $h_I(t)$, so that the latter is piecewise polynomial for classical polynomial interpolation.

What size $I$ should be selected for the basepoint set? Schafer and Rabiner [4] have shown that to obtain a unique basepoint set for an interpolant: 1) there must be an even number of samples in the basepoint set and 2) interpolation should be performed only in the central interval of the basepoint set. The latter restriction is also necessary to avoid delay distortion in the interpolation.

If classical interpolating polynomials are used, an even number of basepoints implies a polynomial of odd degree. The simplest odd polynomial has degree of one and provides linear interpolation between two basepoints. A linear interpolator is not ordinarily regarded as providing good interpolation on curved functions unless the samples are densely spaced. Nonetheless, as will be shown, a linear interpolator may in fact be adequate for many modem applications.

The time-continuous impulse response associated with a linear interpolator is simply an isosceles triangle, with $h_I(0) = 1$, and basewidth of $2T_s$; see Fig. 1. Its frequency response (Fourier transform)

$$H_{I1}(f) = T_s\left[\frac{\sin \pi f T_s}{\pi f T_s}\right]^2$$

is plotted in Fig. 2.

The next odd-degree interpolating polynomial is third order (cubic) and so has a four-sample basepoint set. Simulations, reported below, have demonstrated that the cubic interpolator can work extremely well in typical modem applications. Despite its apparent simplicity, it appears to be even more complex than necessary for many practical situations.

The impulse response for the cubic interpolator is plotted in Fig. 1. Each segment, of duration $T_s$, is a cubic polynomial; the entire impulse response is piecewise polynomial. The impulse response is symmetric about $t = 0$, which is a requirement for linear-phase filtering. It has nulls at $t = iT_s, i \neq 0$, and $h_I(0) = 1$. This property ensures that the basepoint set is interpolated exactly.

Linear phase is highly desirable in most modems, but interpolating the basepoint set is only *interesting, not necessary*. All odd-degree, classical interpolating polynomials exhibit linear phase (provided that the interpolant is taken in the central interval of the basepoint set), and interpolate the basepoint set exactly.

Fourier transform $H_{I3}(f)$ is the frequency response of the cubic interpolating filter, and is plotted in Fig. 2. Several desirable features can be seen:

- Broad nulls are centered on harmonics of the sampling frequency, exactly coincident with the locations of spectral images of the input sample sequence. Therefore, stopband attenuation is concentrated where it is needed most. The nulls are wider than those produced by the linear interpolator.
- The main lobe is broad, contributing only modest amounts of attenuation over much of its passband. (Oetken [5] has shown that the classical interpolating polynomials have maximally-flat frequency response.) The main lobe of the cubic is relatively flat over a wider frequency range than that of the linear. Even at two samples per symbol, the attenuation at half the symbol frequency (i.e., $0.25/T_s$) is only $\approx 0.6$ dB. Therefore, little or no compensation for interpolator distortion need be provided in other filters of the receiver.
- The first sidelobe is down by 30 dB from the peak of the main lobe — a respectable attenuation.

The importance of these characteristics of the frequency response is illustrated on Fig. 3. This figure shows a signal at 2 samples/symbol with 100% excess bandwidth, together with the frequency response of the cubic interpolator. The shaded areas represent the image components (aliasing) [1] which are passed by the interpolator.

Fig. 3 shows a worst case situation. The amount of aliasing

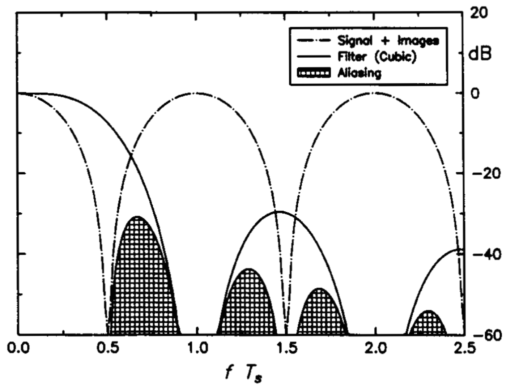

Fig. 3. Aliasing process, cubic interpolator.

can be reduced by reducing the excess signal bandwidth, by increasing the sampling rate or by designing the interpolator such that it has a sharper roll-off between $0.5/T_s$ and $1/T_s$.

## B. Alternative Polynomials

It is not essential that the polynomials describing $h_I(t)$ have degree $I-1$. For example, one can reduce computational load in a four-point interpolator by using quadratic formulas for the segments of $h_I(t)$, as opposed to the cubic expressions provided by the Lagrange formulas.

Taking into account the constraints given by the desire to interpolate the basepoint set, linear phase (even symmetry) and also requiring that the dc gain (sum of coefficients) be independent of $\mu_k$, a single design parameter $\alpha$ remains for characterising a piecewise-quadratic formula. Varying $\alpha$ leads to different interpolator responses. Details of the coefficient formulas are provided in the Appendix.

Fig. 1 shows the impulse responses corresponding to $\alpha = 0$ (linear interpolator) and $\alpha = 0.5$. As will be shown in Section IV, the latter provides excellent performance.[1] The frequency responses of the piecewise-parabolic interpolators are broadly similar to that of the cubic (Fig. 2). For certain values of $\alpha$ however, an extra null and side lobe appear at a frequency below $F_s = 1/T_s$. By varying $\alpha$, a trade-off can be made between the steepness of roll-off of the main lobe and the level of this first side lobe. This behavior can be adjusted to minimize the amount of aliasing from the first signal image.

## C. Other Alternatives

With such encouraging performance from simple polynomials, why might a different filter be preferred?

1) Superior filter functions can be devised; see [6] for examples. They might be higher degree interpolating polynomials, or noninterpolating polynomials, or nonpolynomial functions entirely.
2) In some structures, to be described below, a superior filter function requires no more on-line computation than an inferior function of the same length.

[1]A response very similar to that of the cubic an be obtained by letting $\alpha = 0.25$. Its performance is similar to that of the cubic, but both are inferior to that of the $\alpha = 0.5$ interpolator.

## III. Filter Structures

Two types of implementation can be identified:

1) Precompute and store sample values of filter impulse response $h_I(t)$. Load $I$ stored samples — the filter coefficients — into a transversal filter from memory for each interpolation.
2) Compute interpolants directly on-line without storing impulse-response samples, or possibly even without calculating impulse response explicitly. (Appears to be feasible only for polynomial-based filters.)

Characteristics of these alternatives are pursued below. Additional viewpoints can be found in [6].

## A. Stored Impulse Response

Coefficient memory will be addressed by the desired fractional interval $\mu_k$. To store the impulse response in a finite memory requires that $\mu_k$ be quantized, say into $L$ uniform intervals. In consequence, the recovered clock suffers a timing jitter $T_s/L$, peak-to-peak, with respect to the signal waveform.

There will be $IL$ words of $b_I$ bits per word stored in the memory. For each interpolation, $I$ words of $b_I$ bits must be transferred to the filter structure. The transfer bus can easily become overly wide as either $I$ or $b_I$ grows beyond trivial size. Bus width is likely to constitute a strong impediment to practical implementation of a stored impulse response, at least for high-speed applications where the transfer must be fully parallel.

Implementation depends only upon the structure parameters $L$, $I$, and $b_I$, not $h_I(t)$ itself, which consists merely of the numbers stored in memory. Computing burden is independent of $h_I(t)$; a superior filter characteristic demands no more effort than an inferior one with the same structure parameters. Any filter function whatever may be employed; the stored-response method is not restricted to polynomials.

The stored filter samples are exactly equivalent to the tap coefficients of a polyphase filter [7] with $L$ arms and $I$ taps per arm. Each arm corresponds to a different quantized value of $\mu_k$. A polyphase implementation could be substituted for the stored-filter implementation, with an $L$-fold increase of filter hardware. A polyphase filter is equivalent to a bank of filters, such as described in [8].

## B. On-Line Computation

If all computations are to be performed on-line, there is no need for filter coefficient memory nor the quantization that is imposed thereby. All calculations can be performed in the native word size of the machine in use, with only such quantization as is imposed by this word size.

Three examples of direct interpolation are described below. All are restricted to polynomial-based filters. No feasible method for direct, high-speed interpolation using nonpolynomial filters has been uncovered.

One direct approach might be to evaluate filter coefficients (e.g., from Lagrange formulas) as a function of $\mu_k$ for each individual interpolant. Particularly simple formulas, or low speed, or high parallelism would be required in most instances to accomplish this method expeditiously. Linear interpolation, or the four-point piecewise-parabolic impulse response provide simple formulas and are well-suited for rapid direct evaluation of filter coefficients.

The simulations described in Sections IV and V employ direct evaluation of filter coefficients, since speed of operation is not paramount.

Although a filter characteristic underlies any interpolator, one is not constrained to compute the filter coefficients if other satisfactory methods can be devised. It is not necessary that the impulse response be explicitly visible in the interpolator structure. The objective is to produce interpolants, not build filters. For example, linear interpolation can be performed by the simple formula

$$\begin{aligned} y(kT_i) = y(k) &= (1-\mu_k)x(m_k) + \mu_k x(m_k+1) \\ &= x(m_k) + \mu_k[x(m_k+1) - x(m_k)] \end{aligned} \quad (2)$$

As another example, Newton's method is a textbook approach [9], which computes the interpolant directly from a tableau of the signal samples and their differences. In Newton's method, one never computes the filter coefficients (nor the coefficients of the polynomial $p_k(t)$ either). Yet the same interpolant is obtained as if the basepoint samples had been applied to the appropriate filter. Newton's method is convenient for hand computation from tabular data.

Another approach, better suited for signal interpolation by machine, has been devised by Farrow [10]. Let the impulse response be piecewise polynomial in each $T_s$-segment, $i = I_1$ to $I_2$:

$$h_I(t) = h_I[(i+\mu_k)T_s] = \sum_{\ell=0}^{N} b_\ell(i)\mu_k^\ell \quad (3)$$

Substitute (3) into (1) and rearrange terms to show that the interpolants can be computed from

$$\begin{aligned} y(k) &= \sum_{i=I_1}^{I_2} x(m_k - i) \sum_{\ell=0}^{N} b_\ell(i)\mu_k^\ell \\ &= \sum_{\ell=0}^{N} \mu_k^\ell \sum_{i=I_1}^{I_2} b_\ell(i)x(m_k-i) \\ &= \sum_{\ell=0}^{N} \mu_k^\ell v(\ell), \quad v(\ell) = \sum_{i=I_1}^{I_2} b_\ell(i)x(m_k-i) \end{aligned} \quad (4)$$

The coefficients $b_\ell(i)$ are fixed numbers, independent of $\mu_k$, determined solely by the filter's impulse response $h_I(t)$. There are $NI$ such coefficients if all impulse-response segments are described by polynomials of the same degree $N$.

Equation (4) is itself a polynomial in $\mu_k$. Nested evaluation of (4) is the most-efficient approach. For a cubic interpolation:

$$y(k) = [\{v(3)\mu_k + v(2)\}\mu_k + v(1)]\mu_k + v(0) \quad (5)$$

A block diagram for hardware evaluation of (4) is shown in Fig. 4. This *Farrow Structure* consists of $N+1$ columns of FIR transversal filters, each column with fixed tap coefficients. Each FIR column has $I$ taps. Since tap weights are fixed, they might readily be implemented by table look-up, rather than as actual multipliers.

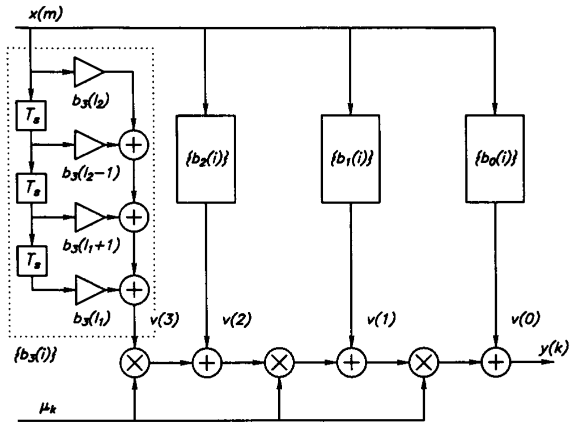

Fig. 4. Farrow structure for cubic interpolator.

TABLE I
FARROW COEFFICIENTS $b_\ell(i)$ FOR CUBIC INTERPOLATOR

| $i$ | $\ell=0$ | $\ell=1$ | $\ell=2$ | $\ell=3$ |
|---|---|---|---|---|
| $-2$ | 0 | $-\frac{1}{6}$ | 0 | $\frac{1}{6}$ |
| $-1$ | 0 | 1 | $\frac{1}{2}$ | $-\frac{1}{2}$ |
| 0 | 1 | $-\frac{1}{2}$ | $-1$ | $\frac{1}{2}$ |
| 1 | 0 | $-\frac{1}{3}$ | $\frac{1}{2}$ | $-\frac{1}{6}$ |

TABLE II
FARROW COEFFICIENTS $b_\ell(i)$ FOR PIECEWISE-PARABOLIC INTERPOLATOR

| $i$ | $\ell=0$ | $\ell=1$ | $\ell=2$ |
|---|---|---|---|
| $-2$ | 0 | $-\alpha$ | $\alpha$ |
| $-1$ | 0 | $\alpha+1$ | $-\alpha$ |
| 0 | 1 | $\alpha-1$ | $-\alpha$ |
| 1 | 0 | $-\alpha$ | $\alpha$ |

In general, $I$ and $N$ are independent; $I$ is set by duration of impulse response and $N$ is determined by the chosen filter piecewise polynomials. In the special case that the approximating polynomial (not the filter polynomials) interpolates the basepoints, then $I = N+1$ and the structure is square.

Nested evaluation is performed by a cascade of $N$ multipliers. One input to each multiplier is the fractional interval $\mu_k$ and the other is the part of (5) computed so far.

A Farrow structure needs to transfer only the one variable $\mu_k$ itself for each interpolation, instead of $I$ filter coefficients as required by a stored-filter implementation. The transfer problem is greatly reduced, but at a cost of increased complexity in the interpolator structure.

A Farrow decomposition can be found for any polynomial-based filter. Farrow coefficients $\{b_\ell(i)\}$ are shown in Table I for the particular case of a cubic interpolating polynomial; these numbers are derived from the Lagrange formulas.

Table II shows the coefficients for a Farrow structure

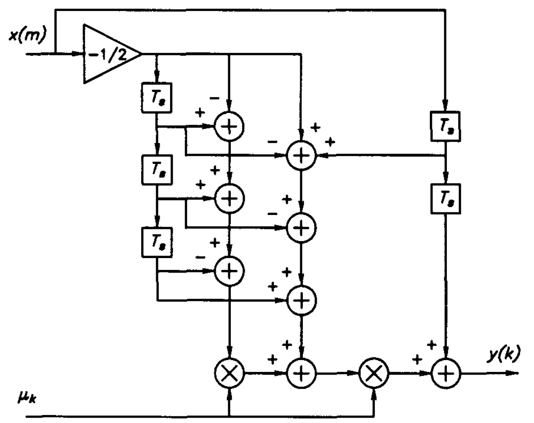

Fig. 5. Farrow structure for piecewise parabolic interpolator ($\alpha = 0.5$).

TABLE III
COMPUTATIONAL COMPLEXITY OF DIFFERENT INTERPOLATORS

| Operation | Linear (Eq. 2) | Parab. $\alpha = 0.5$ (Farrow) | Cubic (Farrow) | Cubic (Direct) |
|---|---|---|---|---|
| Delay | 1 | 5 | 8 | 3 |
| Scale by constant | 0 | 1 | 3 | 0 |
| Add/Subtract | 2 | 9 | 11 | 15 |
| Multiply/Divide | 1 | 2 | 3 | 16 |

Note: Multiple-input operations are counted as the equivalent number of 2-input operations required to obtain the same result.

implementing the four-point piecewise-parabolic interpolation filter, expressed in terms of the design parameter $\alpha$. It can be seen for $\alpha = 0.5$ — which provides good filter characteristics — that the multiplications become simple and many coefficients become identical. Fig. 5 shows a Farrow structure implementing the piecewise-parabolic interpolator with $\alpha = 0.5$.

### C. Computational Complexity

Table III shows the computational complexities of some of the interpolator structures discussed, expressed in terms of the numbers of different operations required to compute one interpolant. In this table, the "linear interpolator" implements the second line of (2) and the piecewise-parabolic is that of Fig. 5. A practical Farrow structure for the cubic can be derived from Fig. 4 and Table I in a number of ways; the way we have chosen exploits the appearance of the same coefficient value $b_\ell(i)$ in several different positions and tends to minimize the number of scalings and multiplications, at the expense of some extra delay elements. The direct-form cubic computes the Lagrange formulae at each interpolation point. We have separated genuine multiplications from scalings by a fixed constant in Table III because the latter is often simpler to implement in hardware.

The linear interpolator is clearly attractive in terms of hardware complexity. Fortunately, as will be shown in Section IV, it appears to provide adequate performance in many situations.

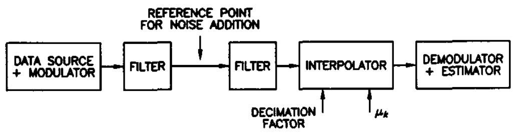

Fig. 6. Simulation model for performance estimation.

## IV. PERFORMANCE SIMULATION

Performance degradation contributed by the interpolator to reception of a BPSK signal was investigated with the simulation program BOSS, using a model as shown on Fig. 6. Data filters were modeled as transversal structures, truncated to 10-symbols length by a rectangular window. All interpolators used direct, on-line, floating-point computation of filter coefficients.

In order to isolate the distortions contributed by the interpolator from other imperfections, the signal generation and filtering in all cases was performed at a high sampling rate of 10–16 samples/symbol. The signal was subsequently decimated in order to model an interpolator at a lower rate.

Data signals were generated with an exact integer number of samples per symbol. When subsequent decimation preserves an integer number of samples per symbol, the decimated samples are already synchronised to the symbol rate of the simulated data signal and a *fixed* value of $\mu_k$ is used for recovery of all data symbols. In order to avoid timing offsets, different values of $\mu_k$ are required for different choices of the samples that survive decimation ("decimation phases"). In all the results reported below we have used combinaions of values of the decimation phase and $\mu$ which result in samples being interpolated at the "optimum" position within each symbol (minimum error probability).

The overall performance results reported below were obtained by averaging results over a range of decimation phases corresponding to a variation in $\mu$ between 0 and 1. Some examples of the $\mu$-dependence of the performance are provided.

The interpolator filter is a part of the total shaping of the signal performed in the receiver. When designing other filters, the response of the interpolator must therefore be taken into account in order to achieve the desired overall response. Starting from a conventional receiver filter design containing all the desired shaping, accounting for the interpolator can be considered a process of compensation.

In a practical system this compensation must be fixed, independent of $\mu$. A natural first guess of the required compensation would therefore be the inverse response of the continuous underlying filter (CUF) of the interpolator in the passband of the signal. Fig. 7 shows, as an example, the frequency responses of a $\sqrt{100\%\ \text{CRO}}$ filter[2] with and without compensation for the CUF of a linear interpolator at 2 samples/symbol. We have used this type of compensation throughout the work reported here. Although superior compensations may exist, excellent results are obtained using this type.

[2] $\sqrt{100\%\ \text{CRO}}$ = root cosine roll-off filter with 100% excess bandwidth.

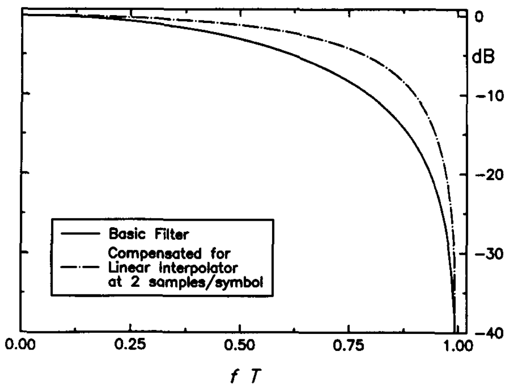

Fig. 7. Effect of compensation for linear interpolator on $\sqrt{100\%\ \text{CRO}}$ filter.

TABLE IV
Degradation Caused By Cubic Interpolator

| #Samp. /symb. | R/O % | $P_e$ | $\Delta E_b/N_0$ (dB) | |
|---|---|---|---|---|
| | | | Unc. | Comp. |
| 2 | 50 | $10^{-2}$ | 0.03 | 0.01 |
| | | $10^{-6}$ | 0.10 | 0.03 |
| | 100 | $10^{-2}$ | 0.07 | 0.04 |
| | | $10^{-6}$ | 0.14 | 0.06 |

Most results were obtained using semi-analytic error probability estimation, using BPSK modulation and averaging over 1023 symbols (the data source was a PN sequence generated by a 10-stage shift register).

The semianalytic technique requires the equivalent noise transmission of the entire receiver chain to be calibrated explicitly at the beginning of each simulation. Calibration consists of summing the squared impulse response of the entire receiver to determine the variance of noise added to the data strobes. Noise calibration in an interpolating receiver is not a constant; it is a function of $\mu_k$.

In a time-invariant receiver, the noise calibration can be interpreted as equivalent to a noise bandwidth, but that interpretation is not strictly valid in a time-varying receiver. For the special case of an exact integer number of samples per symbol, $\mu_k$ is fixed for a given sample sequence and so one can postulate a fictitious $\mu_k$-dependent noise bandwidth.[3]

## A. Cubic Interpolator

Table IV shows the performance loss introduced by a cubic interpolator at 2 samples/symbol. Even without compensation for the frequency response, the degradation is small. With compensation, it becomes negligible.

As expected, the performance is worse with a higher excess bandwidth. The loss occurs because more signal energy at image frequencies is aliased back into the signal's passband (cf. Fig. 3). As with any ISI-like disturbance, the effect of the interpolator in terms of $E_b/N_0$ degradation is worse at lower error probabilities.

The very small degradations observed indicate that it should be feasible to use interpolator structures simpler than the cubic.

## B. Linear Interpolator

Performance of the linear interpolator is summarized in Table V. The results for 100% excess bandwidth are also shown on Fig. 8. This figure shows that, when CUF compensation is performed, it is possible to use the linear interpolator at quite low sampling rates. For example, if an allowance of 0.1 dB is made for interpolator imperfection, then a linear interpolator will suffice down to 2.5 samples/symbol, even at an error probability of $10^{-6}$.

The degradations quoted above are averages of several simulations with different values of $\mu$, as would arise in a system with nonsynchronous sampling. If however the sampling is nearly synchronous, then $\mu$ could remain almost constant for extended periods. It is therefore of interest to see

[3]For the particular case of a $\sqrt{100\%\ \text{CRO}}$ filter in cascade with a linear interpolator at 2 samples/symbol, the range of this variation is $\approx 30\%$ when $\mu_k$ varies between 0 and 0.5.

TABLE V
Degradation Caused By Linear Interpolator

| #Samp. /symb. | R/O % | $P_e$ | $\Delta E_b/N_0$ (dB) | |
|---|---|---|---|---|
| | | | Unc. | Comp. |
| 4 | 50 | $10^{-2}$ | 0.02 | 0.01 |
| | | $10^{-6}$ | 0.05 | 0.02 |
| | 100 | $10^{-2}$ | 0.02 | 0.01 |
| | | $10^{-6}$ | 0.05 | 0.01 |
| 3 | 50 | $10^{-2}$ | 0.04 | 0.02 |
| | | $10^{-6}$ | 0.14 | 0.04 |
| | 100 | $10^{-2}$ | 0.05 | 0.02 |
| | | $10^{-6}$ | 0.15 | 0.04 |
| 2.5 | 100 | $10^{-2}$ | 0.11 | 0.04 |
| | | $10^{-6}$ | 0.33 | 0.10 |
| 2 | 50 | $10^{-2}$ | 0.20 | 0.06 |
| | | $10^{-6}$ | 0.74 | 0.20 |
| | 100 | $10^{-2}$ | 0.27 | 0.10 |
| | | $10^{-6}$ | 0.86 | 0.24 |

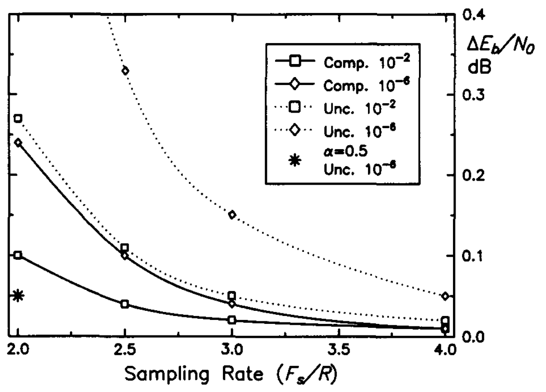

Fig. 8. Performance dependence on sampling rate. (100% CRO filtering, linear interpolator).

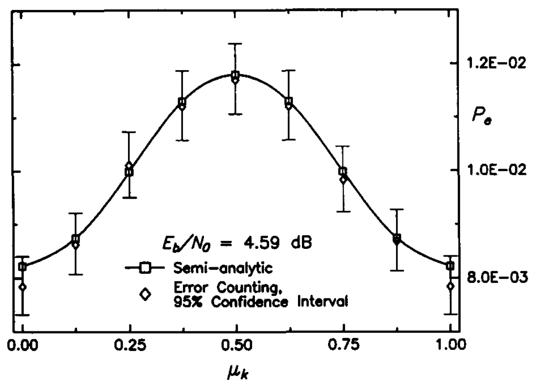

Fig. 9. Performance dependence on $\mu$. (100% CRO filtering, linear interpolator, uncompensated).

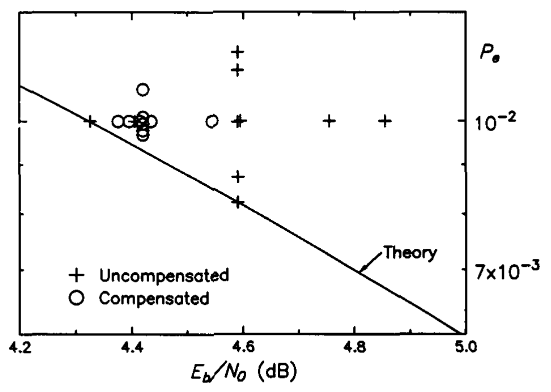

Fig. 11. Performance dependence on $\mu$. (100% CRO filtering, linear interpolator).

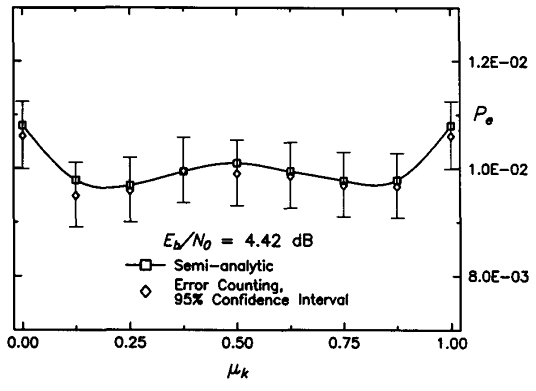

Fig. 10. Performance dependence on $\mu$. (100% CRO filtering, linear interpolator, compensated).

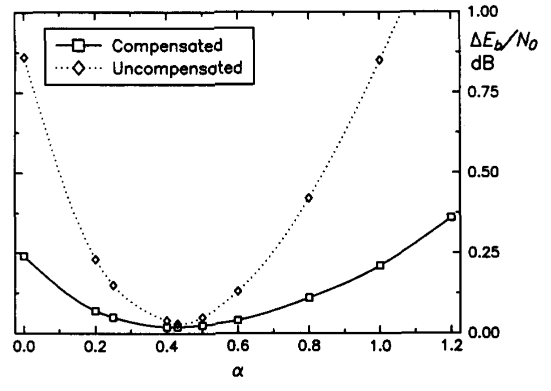

Fig. 12. Performance dependence on $\alpha$. (100% CRO filtering, piecewise-parabolic interpolator).

the variation of the degradation with $\mu$. This is shown on Figs. 9 and 10 for the uncompensated and compensated receivers, respectively[4].

Fig. 9 shows a minimum degradation at $\mu = 0$ and $\mu = 1$, corresponding to strobes coincident with basepoint samples (no interpolation needed). In between, passband distortion and $\mu$-dependent aliasing combine to increase the degradation. The compensated system on the other hand, does not have minimum degradation at $\mu = 0$, because the receiver filter in this case is not exactly matched to the incoming signal and the overall response is not Nyquist. Apart from the overall gain in performance achieved by the compensation, the variation with $\mu$ is greatly reduced in the compensated system, so there is less need to worry about quasi-stationary operation near the degradation peak.

Another display of the effect of the distortion is presented on Fig. 11. This figure shows on the one hand the $P_e$ values obtained for different values of $\mu$, with an $E_b/N_0$ set to achieve a target average $P_e$ (vertical clusters), and on the other hand, the $E_b/N_0$ values required to achieve the target error probability for particular values of $\mu$ (horizontal clusters). The reduction from compensation, both in the average degradation and in the variation thereof, is clearly seen.

[4]Figs. 9 and 10 also show results obtained by Monte-Carlo simulation, to verify the validity of the semi-analytic model. As can be seen, agreement with semi-analytic estimation is excellent.

In summary, the linear interpolator appears to provide good performance, even with high excess bandwidths, down to sampling rates of about 2.5 samples per symbol. At lower sampling rates, or if extremely low loss is required, it may be necessary to consider more complex interpolators.

## C. Piecewise-Parabolic Interpolator

As indicated above, the piecewise-parabolic interpolator has a complexity in between those of the linear and cubic interpolators. Which value of the design parameter $\alpha$ should be used for best performance? The answer may depend on many factors; for a particular case (100% CRO, 2 samples/symbol, $P_e = 10^{-6}$), the variation of performance with $\alpha$ is shown on Fig. 12. The optimum seems to be close to $\alpha = 0.43$. However, as discussed in Section III, a value of 0.5 allows a particularly simple hardware implementation. The loss compared to $\alpha = 0.43$ seems minimal; we have therefore investigated the $\alpha = 0.5$ interpolator further.

Table VI shows the results. The degradations are extremely small, even smaller than those obtained for the cubic interpolator (cf. Table IV). This reflects the reduction in the level

TABLE VI
DEGRADATION CAUSED BY PIECEWISE-PARABOLIC INTERPOLATOR ($\alpha = 0.5$)

| #Samp. /symb. | R/O % | $P_e$ | $\Delta E_b/N_0$ (dB) | |
|---|---|---|---|---|
| | | | Unc. | Comp. |
| 2 | 50 | $10^{-2}$ | 0.02 | 0.01 |
| | | $10^{-6}$ | 0.04 | 0.02 |
| | 100 | $10^{-2}$ | 0.03 | 0.02 |
| | | $10^{-6}$ | 0.05 | 0.03 |

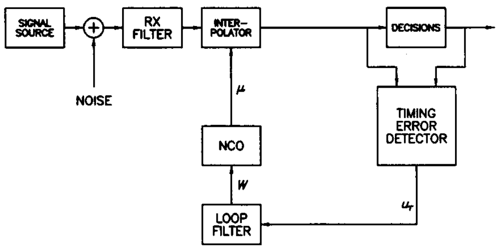

Fig. 13. Closed-loop simulation model.

of aliasing compared to the cubic interpolator (cf. Figs. 2 and 3). Results not reported here reveal that the variation of the degradation with $\mu$ is considerably smaller for this interpolator than for the linear. For comparison, the worst-case result from Table VI (100% CRO, uncompensated, $P_e = 10^{-6}$) is marked on Fig. 8.

## V. Closed-loop Simulation

The NCO-based control method proposed in [1] exhibits many similarities to a conventional, analogue phase-locked loop. In the following, a linearised model of the tracking behavior is derived. The operation of this model is demonstrated by simulations.

Fig. 13 shows a simplified block diagram of the system, with emphasis on the receiver timing loop, consisting of the timing error detector, loop filter, NCO and interpolator. The equivalent PLL operates at the symbol frequency, independent of the number of samples per symbol provided to the timing error detector.

The precise type of interpolator is not important for the principle of the loop operation; results shown later in this section were obtained using the piecewise-parabolic structure with $\alpha = 0.5$.

Similarly, the principle of operation does not depend on the details of the timing error estimation algorithm used; we employed the Data-Transition Tracking Loop (DTTL) algorithm [11,12] to compute the timing error signal $u_\tau$. For BPSK, the expression is

$$u_\tau(k+1) = x_r\left(k + \frac{1}{2}\right) \cdot (x_d(k) - x_d(k+1)) \qquad (6)$$

where $x_r$ is the real part of the (time- and phase-corrected) sampled baseband signal, $x_d$ is a ($\pm 1$) hard-decision based on $x_r$ and $k$ is an (arbitrary) symbol index. This algorithm requires two samples per symbol, one at the decision instant and the other at the nominal zero-crossing time.

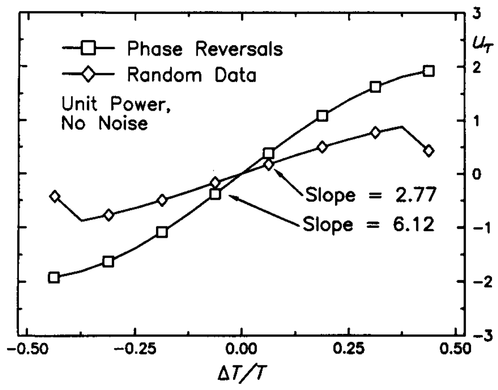

Fig. 14. $s$-curves for DTTL estimator.

### A. Loop Design

Two parameters important for the loop design are the error detector sensitivity $K_d$ and the NCO frequency sensitivity $K_0$. For tracking analysis, $K_d$ is the slope of the detector around zero error. The output of a DTTL detector is proportional to the signal amplitude and also depends on the pulse shape. In addition, the algorithm has self-noise (except in some special situations), so the average output will also depend on the message statistics. Fig. 14 shows (additive noise-free) $s$-curves of the DTTL algorithm for BPSK signals with unit power at the input to the receiver filter. The curve labeled "random" represents the average output when the input is a pseudorandom sequence, with 50% CRO filtering. The "reversals" curve is the response to a 101010 . . . data sequence, equivalent to a sine wave at 1/2 the symbol rate. The algorithm has no self-noise with the latter signal, making results easier to interpret.

The frequency sensitivity of the NCO and interpolator can be derived in the following way. Consider the loop in tracking mode, with the NCO receiving a constant control value $W$ ($W \approx -1$) and overflowing at regular intervals[5]. Now add a small value $\delta$ ($\delta \ll 1$) to $W$ at a single sampling instant. This will cause the overflow times to shift by exactly $\delta$ sampling intervals (the slope of the NCO is $-1$), but the frequency will not be affected.

Adding $\delta$ to $W$ at all sampling times instead of at a single instant corresponds to a step change in $W$ of amount $\delta$. This would cause a signal phase change of $\delta T_s/T$ symbol periods in each sampling interval, or a change of $\delta$ symbol periods per symbol interval. This is equivalent to a change in frequency by $\delta$ symbol rates. The NCO-plus-interpolator frequency sensitivity $K_0$ is thus 1 symbol rate per unit input.

Knowing $K_0$ and $K_d$, the loop can be designed and analysed using classical PLL theory [13]. Detailed loop design considerations are outside the scope of this paper; we shall look at just a single example.

[5]The NCO in [1] had modulus 1 whereas the one considered here has modulus $T_i/T_s$.

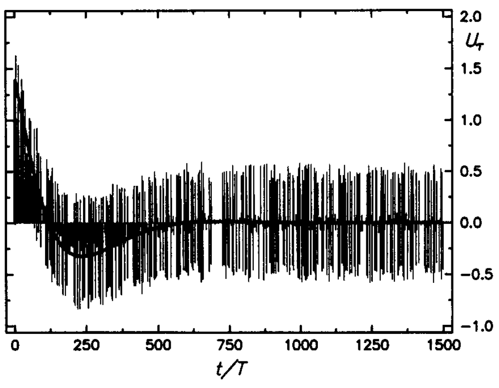

Fig. 15. Error signal transient response, phase step.

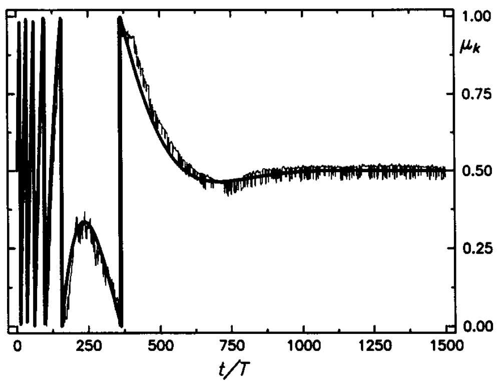

Fig. 17. $\mu$ transient response, phase step (16 samples/symbol).

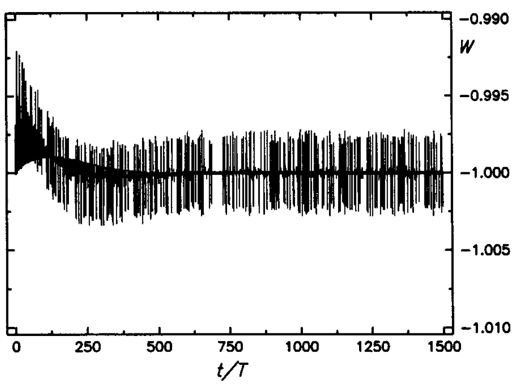

Fig. 16. NCO control word transient response, phase step.

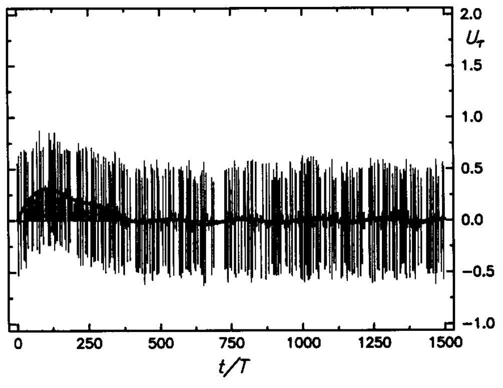

Fig. 18. Error signal transient response, frequency step.

### B. Loop Operation

Fig. 15 shows the transient response of a critically damped loop having a double-sideband noise bandwidth of 1% of the symbol rate. The loop filter is a proportional-plus-integral structure, so the overall loop is Type II. Input to the loop is a phase step of one-quarter symbol period.

Two separate curves are overlaid upon each other in this and succeeding figures. A smooth, solid curve is obtained from a data signal with alternating 1's and 0's, while the ragged curve results from a signal with random data. Self-noise causes the fluctuations in the latter, since no external noise was added for the simulation. Loop gain is adjusted so that the loop has the same bandwidth and damping for both patterns. The transient behavior is consistent with that predicted by theory (e.g. [13]).

Fig. 16 shows the corresponding transient in the NCO control word $W$. Because the input was a phase step, rather than a frequency change, the steady-state value of $W$ returns to $-1$.

Fig. 17 shows the variation of $\mu$ in response to the same input. This particular simulation was performed using 16 samples/symbol, so the input phase change of 1/4 symbol period corresponds to a shift in optimum timing of 4 samples. Later, $\mu$ settles to a constant steady-state value, in this case 0.5. If the sampling rate is reduced to 4 per symbol, we would expect only a single cycle of $\mu$ in response to the same input. This is confirmed by results not presented here.

Response to a step in symbol frequency of 0.1% is shown on Figs. 18, 19, and 20. The error signal $u_r$ settles to 0, confirming that this change is within the tracking capabilities of the Type-II loop (The residual steady-state oscillation will be explained presently). The NCO control word settles to a value different from $-1$, reflecting the change in data rate. Recalling that the $K_0$ of the NCO-plus-interpolator is 1 symbol rate/unit input, the steady-state change should be 0.001 for a 0.1%-step, as confirmed by Fig. 19.

Sampling rate was not altered when the step in data rate occurred. The signal structure therefore slides with respect to the sampling grid, so there should be a linear variation of the optimum $\mu$ with time. At 4 samples per symbol, 0.1% of the symbol rate corresponds to a shift of 1 sample in 250 symbols. This is confirmed by Fig. 20.

The error signal output of the detector is directly proportional to the sample value at the nominal zero-crossing time, half-way between the decision points (6). When the signal is sliding with respect to the sampling structure, this point is interpolated with steadily-varying values of $\mu$. Because the interpolator is nonideal, a $\mu$-dependent interpolation error in this sample value is introduced. The result is the residual oscillation, with a period equal to the period of $\mu$, which can be seen on Fig. 18. In turn, this variation will cause perturbations

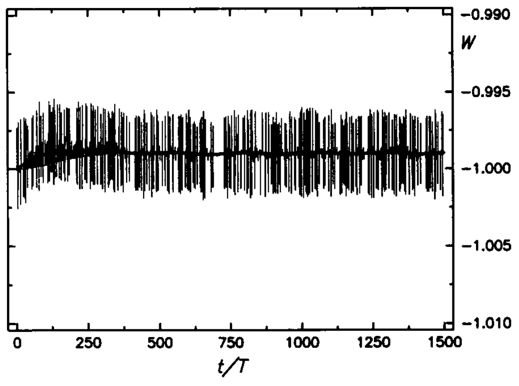

Fig. 19. NCO control word transient response, frequency step.

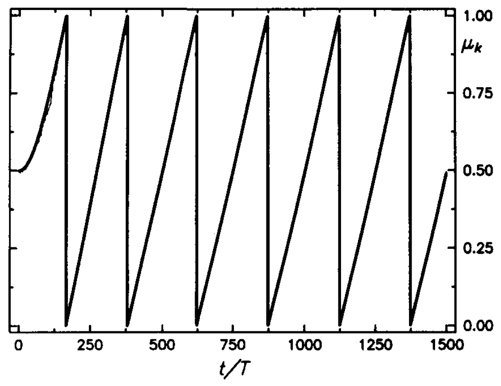

Fig. 20. $\mu$ transient response, frequency step.

on $W$ and $\mu$ itself. As long as the loop is stable however, the error will tend to be compensated. With a good interpolator, the effect is in any case small.

The results presented above were all obtained without any thermal noise. Noise affects the behavior of the loop in several ways; a detailed analysis is beyond the scope of this paper. We have performed a limited amount of simulation with thermal noise and obtained results which are in close agreement with those reported in [14].

## VI. Summary And Discussion

Part I of this work [1] presented the fundamentals of timing adjustment in digital modems using interpolation. It discussed the broad characteristics required of the interpolating filter and introduced an NCO-based control scheme.

In Part II we have studied the use of polynomial based filters, specifically those based on interpolating polynomials. Several different implementation possibilities were presented. The Farrow structure [10] appears to be particularly interesting for high-speed operation as it only requires the transfer of a single control value and uses a small number of multiplications.

We have not looked at nonpolynomial filters or noninterpolating polynomials. These might in principle give improved performance, but we found that excellent performance is available with very simple interpolating polynomial filters.

Computer simulations have shown that simple interpolators, compensated for distortion introduced by the interpolating filter, afford excellent results. With compensation, two-point linear interpolation results in very small interpolation error for many applications. For more critical applications, four-point interpolation using piecewise-parabolic interpolating filters ($\alpha = 0.5$) gives excellent performance with only moderate complexity increase. Performance is better than for the four-point cubic interpolator and only two multipliers are needed to implement the Farrow Structure.

The results presented here were all obtained using conventional computer floating-point arithmetic. In practice, for high-speed operation, relatively coarse quantisation and fixed-point arithmetic might be more appropriate. The choice of suitable parameters must be made in the context of the overall digital receiver design. Tradeoffs between word length and sampling rates are reported in [14].

Closed-loop simulations have shown that the NCO-based control system has many similarities to a conventional analogue phase-locked loop. Results have been presented showing responses to clock phase- and frequency-steps.

## Appendix: Coefficient Formulas For Interpolating filters

Classical polynomial interpolation of an $N$-point basepoint set $\{t_i, x(i)\}$ can be performed by the Lagrange formulas

$$y(t) = \sum_{i=I_1}^{I_2} C_i x(I_1 + I_2 - i) \tag{A.1}$$

where

$$C_i = \prod_{j=I_1, j\neq i}^{I_2} \frac{t - t_j}{t_i - t_j} \tag{A.2}$$

For even values of $N$, the limits are set to $I_1 = -N/2$ and $I_2 = N/2 - 1$, respectively. With this normalization, $t = (i + \mu)T_s$ and

$$h_I[(i+\mu)T_s] = C_i(\mu) \tag{A.3}$$

$$\begin{array}{rclcrrrr}
C_{-2} & = & \frac{(\mu+1)\mu(\mu-1)}{6} & = & \frac{1}{6}\mu^3 & & -\frac{1}{6}\mu & \\
C_{-1} & = & \frac{(\mu+1)\mu(\mu-2)}{-2} & = & -\frac{1}{2}\mu^3 & +\frac{1}{2}\mu^2 & +\mu & \\
C_0 & = & \frac{(\mu+1)(\mu-1)(\mu-2)}{2} & = & \frac{1}{2}\mu^3 & -\mu^2 & -\frac{1}{2}\mu & +1 \\
C_1 & = & \frac{\mu(\mu-1)(\mu-2)}{-6} & = & -\frac{1}{6}\mu^3 & +\frac{1}{2}\mu^2 & -\frac{1}{3}\mu &
\end{array} \tag{A.4}$$

for regularly spaced samples. For $N = 4$, we get the coefficient formulas for cubic interpolation shown above in (A.4).

With the constraints discussed in the main text, the coefficient formulas for the piecewise-parabolic interpolator are

$$\begin{array}{rcrrr}
C_{-2} & = & \alpha\mu^2 & -\alpha\mu & \\
C_{-1} & = & -\alpha\mu^2 & +(\alpha+1)\mu & \\
C_0 & = & -\alpha\mu^2 & +(\alpha-1)\mu & +1 \\
C_1 & = & \alpha\mu^2 & -\alpha\mu &
\end{array} \tag{A.5}$$

where $\alpha$ is the design parameter. $C_1$ and $C_{-2}$ are identical. As can be seen from (4) in the main text and Tables I and II, the coefficients of the polynomials are identical to the tap weights in the Farrow structure.

With $\alpha = 0$, (A.5) reduces to a linear interpolator:

$$\begin{aligned} C_{-1} &= \mu \\ C_0 &= 1 - \mu \end{aligned} \tag{A.6}$$

with $C_1$ and $C_{-2}$ both being 0.

## REFERENCES

[1] F.M. Gardner, "Interpolation in digital modems— Part I: Fundamentals," *IEEE Trans. Commun.*, vol. 41, pp. 502–508, Mar. 1993.

[2] W.H. Press, B.P. Flannery, S.A. Teukolsky, W.T. Vetterling, *Numerical Recipes*. Cambridge, England: Cambridge University Press, 1986, Section 3.5.

[3] M. Abramowitz and I.A. Stegun, Eds., *Handbook of Mathematical Functions*. Nat. Bur. Stds., *Appl. Math. Series,* vol. 55, June 1964, p. 878.

[4] R. W. Schafer and L.R. Rabiner, "A digital signal processing approach to interpolation," Proc. IEEE, v. 61, pp. 692–702, June 1973.

[5] G. Oetken, "A new approach for the design of digital interpolating filters," *IEEE Trans. Acoust., Speech, Signal Prccessing*, vol. ASSP-27, pp. 637–643, Dec. 1979.

[6] T.A. Ramstad, "Digital methods for conversion between arbitrary sampling frequencies," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-32, pp. 577–591, June 1984.

[7] R.E. Crochiere and L.R. Rabiner, *Multirate Digital Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1983.

[8] O.E. Agazzi *et. al.*, "A digital signal processor for an ANSI standard ISDN transceiver," *IEEE J. Solid-State Circuits*, vol. 24, pp. 1605–1613, Dec. 1989.

[9] F.B. Hildebrand, *Introduction to Numerical Analysis*. New York: McGraw-Hill, 1956, Section 2.5.

[10] C.W. Farrow, "A continuously variable digital delay element," in *Proc. IEEE Int. Symp. Circuits & Syst.*, Espoo, Finland, June 6–9, 1988, pp. 2641–2645.

[11] F.M. Gardner, "A BPSK/QPSK timing-error detector for sampled receivers," *IEEE Trans. Commun.*, vol. COM-34, pp. 423–429, May 1986.

[12] W.C. Lindsey and M.K. Simon, *Telecommunications Systems Engineering*. Englewood Cliffs, NJ: Prentice-Hall, 1973, ch. 9.

[13] F.M. Gardner, *Phaselock Techniques*. New York: Wiley, 1979, 2nd ed.

[14] T. Jesupret, M. Moeneclaey, and G. Ascheid, "Digital demodulator synchronisation," Final Rep. to ESTEC Contract 8437/89/NL/RE, June 1991.

**Lars Erup** (M'81) was born in Angmagssalik, Greenland, in 1955. He received the Akademiingenior degree from the Technical University of Denmark in 1978.

He joined the European Space Agency in 1980 and has worked mainly on system planning and transmission design for communications satellite systems. He has contributed to the ECS, Olympus, and DRS Satellite projects, specializing in application of computer techniques to system design problems. Current interests include digital techniques for modern synchronization.

Mr. Erup is a member of Dansk Ingeniorforening (Danish Society of Chemical, Civil, Electrical and Mechanical Engineers).

**Floyd M. Gardner** (S'49–A'54–SM'58–F'80) for a photograph and biography, please see the March 1993 issue of this TRANSACTIONS, p. 507.

**Robert A. Harris** (S'69–M'70) received the Ph.D. from the University of Southampton, England, in 1971. In 1970 he joined the European Space Agency (then ESRO), based at the European Space Research and Technology Centre (ESTEC), Noordwijk, The Netherlands, working on telemetry and telecommand systems.

Since 1978 he has been Head of the Transmission Techniques Section in the Directorate of Telecommunications. He was involved in the transmission design of the OTS, ECS, Olympus, and European Data Relay Systems. He has worked on a wide range of analog and digital tranmission topics, with emphasis on modulation, coding, and synchronization techniques and on computer simulation. He is currently involved with the COST 226 project for the wideband interconnection of LAN's by satellite.

Dr. Harris is a member of the Institution of Electrical Engineers.