

Chapter 12

Variable Phase Interpolation

Contents

- Slide 1 Reason for Variable Phase Interpolation
- Slide 2 Another Need for Interpolation
- Slide 3 Ideal Impulse Sampling
- Slide 4 The Sampling Theorem
- Slide 5 Sampling Theorem for Over-Sampling
- Slide 6 Interpolation Between Samples
- Slide 7 Interpolation Between Samples (cont. 1)
- Slide 8 Interpolation Between Samples (cont. 2)
- Slide 8 Approximating Sections by Polynomials
- Slide 9 Approximating Sections by Polynomials
- Slide 10 Approximating Sections by Polynomials
- Slide 11 Quantized Variable Phase Interpolation
- Slide 12 Quantized Phase Interpolation (cont. 1)
- Slide 13 Quantized Phase Interpolation (cont. 2)
- Slide 14 Symbol Clock Tracking Loop
- Slide 15 Symbol Clock Tracking Loop (cont. 1)
- Slide 16 Clock Drift and the mod T Box
- Slide 17 Updating d_i for the Quantized
Interpolator
- Slide 18 Changing the Sampling Rate by a

Rational Factor

- Slide 19** Changing the Sampling Rate (cont. 1)
- Slide 20** Changing the Sampling Rate (cont. 2)
- Slide 21** Changing the Sampling Rate (cont. 3)
- Slide 22** Changing the Sampling Rate (cont. 4)

Experiments for Variable Phase Interpolation

- Slide 23** Open Loop Phase Shifting
Experiments
- Slide 24** Open Loop Experiments (cont.)
- Slide 25** Making a Symbol Clock Tracking
Loop
- Slide 26** Making a Tracking Loop (cont.)

Reason for Variable Phase Interpolation

The receiver is:

- at a distance from the transmitter,
- has slightly different components,
- and is at a different temperature,

so the locally generated symbol clock in the receiver will differ in phase and slightly in frequency from the transmitter's clock.

The receiver must synchronize its symbol clock to the clock in the signal received from the transmitter using information derived from the received signal.

The codec for the TMS320C6713 DSK runs with a fixed sampling phase and frequency. In this experiment, you will learn how to implement the phase shifting in the DSP by a variable phase interpolator. Two possible solutions are presented.

Another Need for Interpolation

In some recent systems, analog modem signals are transmitted over a digital network like the Internet by directly connecting the modem binary output samples in μ - or A-law format, without D/A conversion, to a digital link like a T1 channel where the sampling rate is fixed at 8 kHz and is different from the rate needed for the modem algorithms. The received input samples are also taken directly from the digital channel. This bypasses the ordinary telephone network.

The rate required by modem algorithms is usually a multiple of the symbol rate which is usually not 8 kHz. We will see how to solve this problem by using an interpolation filter bank to convert between two sampling rates that are rationally related.

Ideal Impulse Sampling

- Let $x(t)$ be a band limited signal with cutoff frequency ω_c , that is, $X(\omega) = 0$ for $|\omega| \geq \omega_c$.
- Let the sampling rate be $\omega_s \geq 2\omega_c$ and sampling period be $T = 2\pi/\omega_s$.
- The sampling frequency in Hertz is $f_s = \omega_s/(2\pi) = 1/T$.

The ideal impulse sampled signal is

$$x^*(t) = \sum_{n=-\infty}^{\infty} x(nT)\delta(t - nT)$$

and it can be shown the Fourier transform of $x^*(t)$ is given by the *aliasing formula*

$$X^*(\omega) = f_s \sum_{n=-\infty}^{\infty} X(\omega - n\omega_s)$$

The Sampling Theorem

According to the sampling theorem, $x(t)$ can be exactly reconstructed for any t from its samples $\{x(nT)\}$ by applying $x^*(t)$ to an ideal lowpass filter with the frequency response

$$H_0(\omega) = \begin{cases} 1/f_s = T & \text{for } |\omega| < f_s/2 \\ 0 & \text{elsewhere} \end{cases}$$

and impulse response

$$h_0(t) = \frac{\sin \frac{\omega_s}{2} t}{\frac{\omega_s}{2} t}$$

The reconstruction formula or sampling theorem is

$$\begin{aligned} x(t) &= \sum_{k=-\infty}^{\infty} x(kT) h_0(t - kT) \\ &= \sum_{k=-\infty}^{\infty} x(kT) \frac{\sin \frac{\omega_s}{2} (t - kT)}{\frac{\omega_s}{2} (t - kT)} \end{aligned}$$

Sampling Theorem for Over-Sampling

If the signal is over-sampled so that ω_s is strictly greater than $2\omega_c$, the following narrower band ideal lowpass filter can be used for signal reconstruction to eliminate out-of-band noise:

$$H(\omega) = \begin{cases} 1/f_s & \text{for } -\omega_c < \omega < \omega_c \\ 0 & \text{elsewhere} \end{cases}$$

The impulse response of this filter is

$$h(t) = 2 \frac{\omega_c}{\omega_s} \frac{\sin \omega_c t}{\omega_c t}$$

Then, $x(t)$ can be reconstructed from its samples by the formula

$$\begin{aligned} x(t) &= \sum_{k=-\infty}^{\infty} x(kT) h(t - kT) \\ &= \sum_{k=-\infty}^{\infty} x(kT) 2 \frac{\omega_c}{\omega_s} \frac{\sin \omega_c (t - kT)}{\omega_c (t - kT)} \end{aligned}$$

Interpolation Between Samples

Letting $t = nT + dT$ gives the following formula for interpolating between samples of $x(t)$:

$$\begin{aligned}x(nT + dT) &= \sum_{k=-\infty}^{\infty} x(kT)h(nT - kT + dT) \\&= \sum_{k=-\infty}^{\infty} h(kT + dT)x(nT - kT)\end{aligned}$$

The variable, d , is the time advance normalized by the sampling period.

For actual computation, the sum must be truncated. This can be done by truncating $h(t)$ with a Hanning window. Suppose the impulse response is to be truncated to the time interval $-(L + 0.5) < t/T < L + 0.5$ where L is an integer, and that $-0.5 \leq d < 0.5$.

The required Hanning window is

$$w(t) = \begin{cases} 0.5 + 0.5 \cos \frac{\pi t}{(L + 0.5)T} & \text{for } |t/T| < L + 0.5 \\ 0 & \text{elsewhere} \end{cases}$$

Interpolation Between Samples (cont. 1)

Let the windowed impulse response be

$$\begin{aligned} g(t) &= h(t)w(t) \\ &= \begin{cases} 2 \frac{\omega_c}{\omega_s} \frac{\sin \omega_c t}{\omega_c t} \left(\frac{1}{2} + \frac{1}{2} \cos \frac{\pi t}{(L + \frac{1}{2})T} \right) \\ \quad \text{for } -(L + \frac{1}{2}) < \frac{t}{T} < L + \frac{1}{2} \\ 0 & \text{elsewhere} \end{cases} \end{aligned}$$

Then, the truncated sum becomes

$$\hat{x}(nT + dT) = \sum_{k=-L}^L g(kT + dT)x(nT - kT)$$

A delay of L input samples must be added to make the filter physically realizable, so the actual formula that would be computed is

$$\hat{x}(nT - LT + dT) = \sum_{k=-L}^L g(kT + dT)x(nT - LT - kT)$$

With fixed d , this interpolation formula represents a $2L + 1$ tap FIR filter with tap coefficients $g(nT + dT)$ and input sequence $x(nT)$.

Interpolation Between Samples (cont. 2)

Inside the window the desired interpolation filter impulse response with normalized advance d is

$$\begin{aligned} g(nT + dT) &= h(nT + dT)w(nT + dT) \\ &= 2 \frac{\omega_c}{\omega_s} \frac{\sin \omega_c(n + d)T}{\omega_c(n + d)T} \left(0.5 + 0.5 \cos \frac{\pi(n + d)}{L + 0.5} \right) \\ &\quad \text{for } n = -L, -L + 1, \dots, L \\ &\quad \text{and } -0.5 \leq d < 0.5 \end{aligned}$$

This equation represents $g(t)$ over the interval $[-(L + 0.5)T, (L + 0.5)T]$ by $2L + 1$ sections over the sub-intervals

$$[(n - 0.5)T, (n + 0.5)T] \quad \text{for } n = -L, \dots, L$$

as d varies between -0.5 and 0.5 for each section.

Approximating Sections by Polynomials

Direct computation of the tap coefficients for each new value of d requires evaluation of trigonometric functions and division which takes significantly more time than addition or multiplication in DSP's.

Approximating Sections by Polynomials

A solution to this problem is to approximate the $2L + 1$ sections of $g(t)$ by low degree polynomials. We will approximate the sections by least-squares cubic polynomial fits of the form

$$g_k(d) = c_{0,k} + c_{1,k}d + c_{2,k}d^2 + c_{3,k}d^3 \quad \text{for } k = -L, \dots, L$$

with $-0.5 \leq d < 0.5$

The resulting approximate interpolator for a given d is

$$\tilde{x}(n; d) = \sum_{k=-L}^L g_k(d)x(nT - kT)$$

Substituting the polynomial form for $g_k(d)$ gives

$$\tilde{x}(n; d) = \sum_{i=0}^3 \left[\sum_{k=-L}^L c_{i,k}x(nT - kT) \right] d^i$$

For each i , the sum inside the square brackets is a $2L + 1$ tap FIR filter with tap coefficients $c_{i,k}$ and input sequence $x(nT)$.

Computing the Least-Squares Fits

The program `interp.exe` in the directory `C:\digfil\interp` computes the least-squares fit polynomial coefficients for the interpolator sections.

- The impulse response is truncated with a Hanning window.
- Some samples from adjoining sections are used in computing the section polynomial approximation to reduce discontinuities at section boundaries. To approximate $g(t)$ over the interval $[(n - 0.5)T, (n + 0.5)T]$ a least-squares fit is performed by using samples of $g(t)$ taken uniformly over the interval $[(n - 0.5 - \alpha)T, (n + 0.5 + \alpha)T]$ with $0 \leq \alpha \leq 1$. Experimentally, it was found that $\alpha = 0.04$ gives good results.
- The user is given the option of choosing the cutoff frequency f_c , sampling rate f_s , number of sections, and overlap factor $G = \alpha$.

The program `response.exe` in the `interp01` directory computes the amplitude response and envelope delay relative to the center tap for the filters designed by `interp.exe`. The program asks you to “ENTER ALPHA” which is the desired normalized advance, d .

Quantized Variable Phase Interpolation

Another approach to variable phase interpolation is to divide the symbol period into relatively finely spaced points and design a fixed interpolation filter to achieve the phase shift corresponding to each separate point.

The symbol clock recovery and tracking system then selects the filter with the phase shift closest to the desired value.

Let the sampling period is divided by a factor of M so $d = m/M$ for $m = 0, \dots, M - 1$.

Quantized Phase Interpolation (cont. 1)

Then, for $t = nT + m(T/M)$

$$\begin{aligned}\hat{x}\left(nT + m\frac{T}{M}\right) &= \sum_{k=-\infty}^{\infty} x(kT)g\left(nT + m\frac{T}{M} - kT\right) \\ &= \sum_{k=-L}^L x(kT)g_m(n - k) \\ &\quad \text{for } m = 0, \dots, M - 1\end{aligned}$$

where subfilter m has the impulse response

$$g_m(n) = g\left(nT + m\frac{T}{M}\right) \quad \text{for } m = 0, \dots, M - 1$$

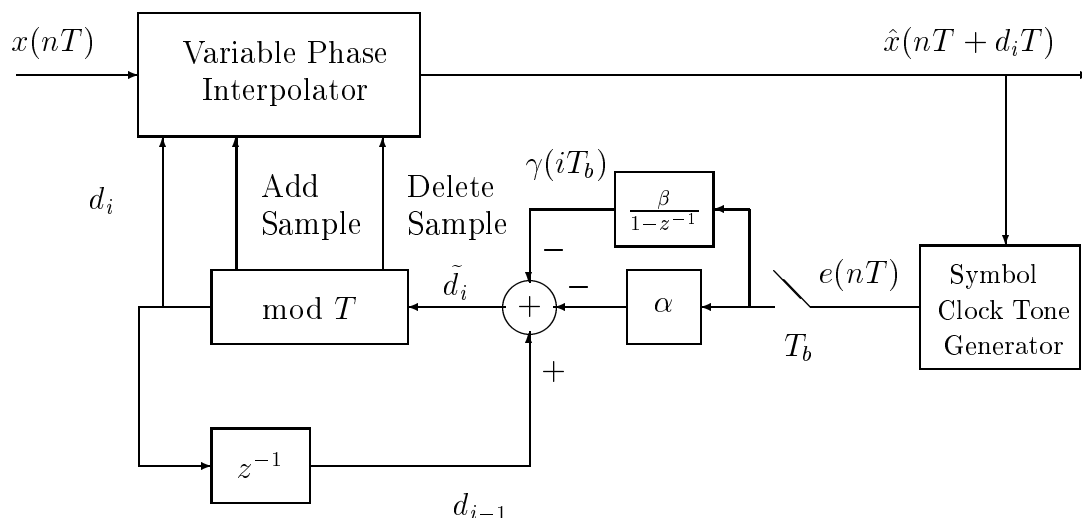
- The range for m was chosen to be 0 to $M - 1$ above while the range for d was selected to be -0.5 to 0.5 . These choices were somewhat arbitrary. For example, if M is even, the range for m could also have been chosen to be $-0.5M$ to $0.5M - 1$.
- For typical modem applications, a reasonable value for the number of phase increments, M , might be between 32 and 64. To get finer

Quantized Phase Interpolation (cont. 2)

resolution, you can linearly interpolate between the outputs of adjacent subfilters based on the required value of d .

- The continuous phase shift method using polynomials is efficient as far as data storage memory is concerned. However, when d is changing frequently, it effectively requires recomputation of the filter coefficients for each new d and is not computationally efficient.
- The quantized step phase shifter is computationally efficient because all the filter coefficients are pre-computed and stored in data memory. However, it is not as efficient in terms of data memory usage. The choice between the two methods is a choice the designer must make based on system constraints.

Symbol Clock Tracking Loop



A variable phase interpolator and the symbol clock tone generator presented in Chapter 11 can be combined into a phase-locked loop for tracking the symbol clock of a PAM signal,

- T is the sampling period
- $T_b = KT$ is the symbol period.
- K is typically 3 or 4 in telephone line modems

It is assumed that the normalized advance, d_i , changes very slowly. The output of the Tone Generator $e(nT)$ is sampled at the symbol rate to generate the phase error sequence $e(iKT) = e(iT_b)$.

Symbol Clock Tracking Loop (cont. 1)

The goal of the loop is to adjust d_i to force $e(iT_b)$ to zero so the loop locks to the positive zero crossings of the generated clock tone. The adjustment formula is:

$$\tilde{d}_i = d_{i-1} - \alpha e(iT_b) - \gamma(iT_b)$$

where

$$\gamma(iT_b) = \beta e(iT_b) + \gamma((i-1)T_b)$$

The accumulator generating $\gamma(iT_b)$ adjusts for a constant frequency offset. The positive constant β should be 50 to 100 times smaller than α for reasonable transient response.

Clock Drift and the mod T Box

When there is a frequency offset between the transmitter and receiver symbol clocks, $\tilde{d}_i$ will slowly drift in a positive or negative direction. In the continuously variable phase interpolator, its value was restricted to the range $[-0.5, 0.5)$.

Clock Drift and the mod T Box

When $\tilde{d}_i$ falls outside $[-0.5, 0.5)$ some corrective action must be taken. We will assume it can only fall a small distance outside this range.

- If $\tilde{d}_i > 0.5$, the normal sample of $x(\cdot)$ should be shifted into the FIR filter delay line and, additionally, the next new sample in time should be shifted in. Then 1 should be subtracted from $\tilde{d}_i$.
- If $\tilde{d}_i < -0.5$, no new sample should be shifted into the delay line and 1 should be added to $\tilde{d}_i$.

The mod T box performs this corrective action. It generates the final phase advance value d_i and informs the interpolator if a sample should be added or deleted.

Updating d_i for the Quantized Variable Phase Interpolator

A similar strategy can be used for the quantized variable phase interpolator.

- It is convenient to use a number of subfilters that is a power of two, say, $M = 2^J$.
- The subfilter index can be stored in a 32-bit integer in a TMS320C6713 DSP with the top J bits, exclusive of the sign bit, actually used as the index.
- The phase increments can be added into the lower bits of the the word which performs the accumulation and averaging.
- When the upper J bits fall outside the allowed interval, corrective action similar to that described in the previous paragraph must be taken.

Changing the Sampling Rate by a Rational Factor

In some applications, samples of the analog modem transmitted signal must be generated at an 8 kHz sampling rate for transmission over a PCM digital channel like a T1 line. Also, PCM samples received at an 8 kHz rate from the T1 channel must be converted into samples at a rate that is a multiple of the modem's symbol rate.

More generally, let the initial sampling rate be f_1 and the final rate be f_2 . Suppose the ratio of f_1 and f_2 when reduced to lowest terms is

$$\frac{f_1}{f_2} = \frac{n_1}{n_2} \quad (1)$$

with n_1 and n_2 relatively prime. Then the intermediate sampling rate should be

$$f_3 = n_2 f_1 = n_1 f_2 \quad (2)$$

Down-sampling the f_3 rate sequence by a factor of n_1 gives the desired rate of f_2 .

Changing the Sampling Rate (cont. 1)

The first step in changing the sampling rate from f_1 to f_3 is to use the sampling theorem to express the continuous-time signal $x(t)$ in terms of its rate f_1 samples. The reconstruction formula is

$$x(t) = \sum_{k=-\infty}^{\infty} x(kT_1)h(t - kT_1)$$

where

$$h(t) = \frac{\sin \pi f_1 t}{\pi f_1 t}$$

Letting $t = nT_1 + m\frac{T_1}{n_2}$ gives

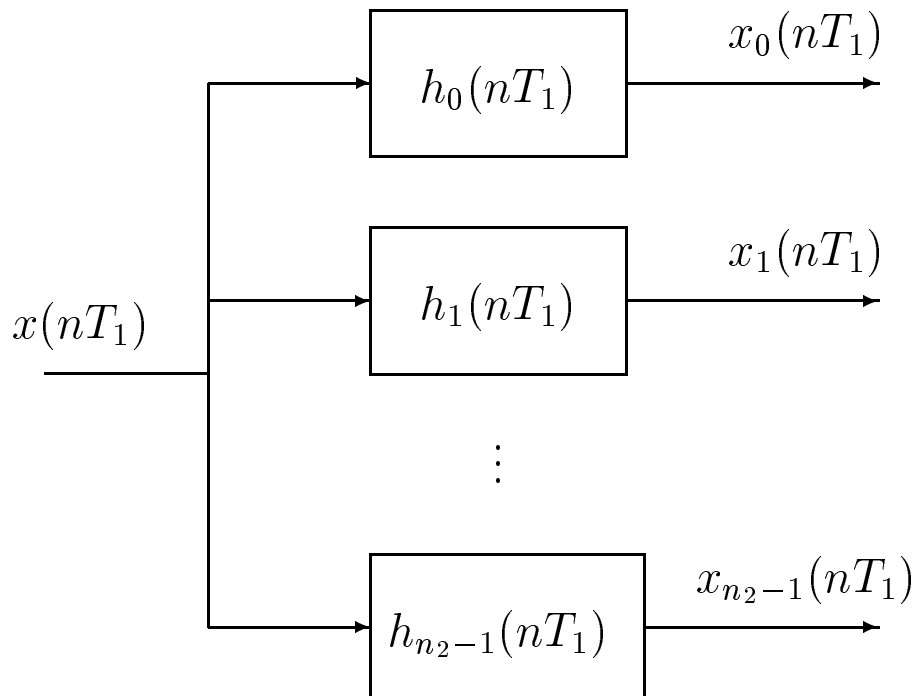
$$\begin{aligned} x_m(nT_1) &= x\left(nT_1 + m\frac{T_1}{n_2}\right) \\ &= \sum_{k=-\infty}^{\infty} x(kT_1)h\left(nT_1 + m\frac{T_1}{n_2} - kT_1\right) \\ &= \sum_{k=-\infty}^{\infty} x(kT_1)h_m(n - k) \\ &\quad \text{for } m = 0, \dots, n_2 - 1 \end{aligned}$$

Changing the Sampling Rate (cont. 2)

Subfilter m is defined to be

$$h_m(n) = h(nT_1 + m\frac{T_1}{n_2}) \quad \text{for } m = 0, \dots, n_2 - 1$$

The interpolation formula shows how to interpolate n_2 points between each rate f_1 sample starting at time nT_1 to generate the rate $f_3 = n_2 f_1$ sequence. For each m , $x_m(nT_1)$ is generated by a discrete-time filter operating with sampling rate f_1 . The interpolation formula is illustrated in the following figure as a bank of n_2 filters.



Changing the Sampling Rate (cont. 3)

Down Sampling

The next step is to down sample the rate f_3 sequence by a factor of n_1 to get the desired rate f_2 sequence.

- Suppose the output of subfilter m is chosen from the n_2 subfilter outputs generated at time nT_1 .
- The next sample that should be selected is the output of subfilter $m + n_1$ if $m + n_1 < n_2$.
- If $m + n_1 \geq n_2$, the output of subfilter $\text{mod}(m + n_1, n_2)$ generated for the interval starting at time $(n + 1)T_1$ should be selected.

Truncating the Impulse Response

In practice, the impulse response $h(t)$ must be truncated to a finite time duration by, for example, a Hamming window.

Changing the Sampling Rate (cont. 4)

Truncating the Impulse Response (cont.)

- Each truncated subfilter is an FIR filter.
- The data samples used to calculate each subfilter output are the same, so only one “delay line” is required to store them.
- Each subfilter uses a different set of taps to convolve with the contents of this delay line.
- Also, for computational efficiency, the outputs of all n_2 subfilters should not be computed each T_1 period. Only the output of the subfilter for the required down-sampled output sample should be calculated.

The interpolation filter bank can be designed using the program `rascos.exe` in `C:\digfil`. To approximate the ideal lowpass filter, set the excess bandwidth factor, α , to 0 and set the number of points per symbol to n_2 .

Experiments for Variable Phase Interpolation

To continue along with the PAM experiments of Chapter 11, use a sampling rate of 16 kHz for the experiments of this chapter.

Open Loop Phase Shifting Experiments

First, you will design and test a quantized variable phase interpolator. Perform the following tasks:

1. Using the sampling rate of $f_s = 16000$ Hz, generate the samples of a 2000 Hz cosine wave and send the resulting sequence to the left channel of the DAC. Connect the left channel line output to the left channel line input. The 2000 Hz cosine wave represents the dotting signal caused by alternating plus and minus input symbols to a two-level PAM transmitter when the symbol rate is 4000 baud.

Open Loop Phase Shifting Experiments (cont.)

2. Design a quantized step phase shifting filter bank as discussed in this chapter. Use a cutoff frequency of $f_c = f_s/2 = 8000$ Hz and let the number of phase steps between samples be $M = 8$.
3. First take the left channel ADC input samples and pass them thorough subfilter $m = 0$. Send the output sequence to the right channel of the DAC. Observe the right and left line outputs simultaneously on the oscilloscope. You will observe a phase shift caused by the delay introduced to make the subfilters physically realizable and by the system filters in the signal paths.
4. Once the $m = 0$ filter is working, test each of the subfilters for $m = 1$ through 7 to make sure they add the expected advance.

Making a Symbol Clock Tracking Loop

Now you will make the symbol clock tracking loop shown on Slide 12-14. Do the following:

1. Write a program to implement the Symbol Clock Tone Generator. You should be able to use the program you created for Chapter 11. Use the sampled 2000 Hz cosine wave as the input to your variable phase interpolator as in Experiment 12.1. Pipe the output of your Variable Phase Interpolator directly to the input of the Symbol Clock Tone Generator inside the DSP program. Do not close the loop yet. Send the output, $e(nT)$, of the tone generator to the DAC and observe the result on the oscilloscope. You should see a clean 4000 Hz sine wave.
2. Once the clock tone generator is working, implement the rest of the loop. Check that it locks to the 4000 Hz symbol clock.

Making a Tracking Loop (cont.)

3. Change the frequency of the 2000 Hz tone slightly and check that your loop tracks the frequency offset.
4. (Optional) Generate a PAM signal with a 4000 baud symbol rate either internally in the local DSP or on another station. Connect the PAM signal to your clock tracking loop and check that it works.