

Interpolation diskreter Signale

Martin Pitt

13. September 2002

1 Motivation

Werden kontinuierliche Signale in einem Rechner verarbeitet, werden sie durch die Abtastung diskretisiert, d. h. die Werte liegen nur an einzelnen Stellen vor. Die Abtastung geschieht mit einer konstanten *Abtastrate* (bzw. *-frequenz* F , d. h. in Intervallen von $T = 1/F$ Sekunden. Bei vielen Prozessen, z. B. beim sog. *Resampling* von Audio-Signalen oder der Skalierung von Pixelgrafiken, benötigt man aber im allgemeinen Werte, die zwischen den gegebenen Abtastwerten liegen. Die Berechnung dieser Werte wird *Interpolation* genannt.

2 Ausgangssituation

Das durch die Abtastung diskretisierte (und quantisierte) Eingangssignal $x(t)$ liegen die Werte an ganzzahligen Vielfachen der Abtastperiode T vor. x_n sei dabei der n -te Abtastwert, d. h. $x_n = x(nT)$ mit $n \in \mathbb{N}$.

Gesucht ist nun eine Funktion, die den ursprünglichen Wert des Eingangssignals an einer bestimmten Stelle t_0 aus einer Teilmenge der vorhandenen Abtastwerte errechnet, bzw. annähert. Je nach Anwendung kann dabei zwischen schnellen oder genauen Algorithmen gewählt werden.

Im folgenden zerlegen wir t_0 in einen ganzzahligen und einen gebrochenen Anteil:

$$t_0 = (n + \mu)T \quad \text{mit} \quad n \in \mathbb{N}, \mu \in [0, 1)$$

D. h.:

$$n = \left\lfloor \frac{t_0}{T} \right\rfloor \quad \mu = \frac{t_0 - \lfloor t_0 \rfloor}{T} = \frac{t_0 - nT}{T}$$

Der zu interpolierende Wert $x(t_0)$ (veranschaulicht: $x_{n+\mu}$) liegt also zwischen den existierenden Abtastwerten x_n und x_{n+1} .

3 Lineare Interpolation

Einfachste aller Interpolationsarten. Zwischen jeweils zwei Abtastwerten wird das Eingangssignal als linear angenommen. Die Aufstellung der Geradengleichung zwischen den zwei uns interessierenden Abtastpunkten x_n und x_{n+1} liefert:

$$x(t) \approx x_n + (x_{n+1} - x_n) \left(\frac{t - nT}{T} \right) \quad ; \quad nT \leq t \leq (n+1)T$$

Einsetzen von t_0

$$x(t_0) \approx x_n + (x_{n+1} - x_n) \underbrace{\left(\frac{t_0 - nT}{T} \right)}_{=\mu}$$

liefert schließlich die gesuchte Formel:

$$x(t_0) \approx (1 - \mu)x_n + \mu x_{n+1}$$

Dies ist nichts weiter als das durch μ gewichtete arithmetische Mittel der beiden benachbarten Abtastwerte.

4 Polynomiale Interpolation

4.1 Herleitung

Das Eingangssignal wird durch eine Folge von Polynomen approximiert. Ich habe folgende Strategie gewählt: Zwischen zwei benachbarte Abtastwerte x_n und x_{n+1} wird ein Polynom $p_n(\mu)$ gelegt. Das gesamte aus den Polynomen gebildete Abtastsignal sollte stetig und differenzierbar sein und außerdem genau durch die vorhandenen Abtastwerte verlaufen. An den Anschlußstellen x_n müssen die Anstiege der Polynome damit gleich sein: $p'_n(1) = p'_{n+1}(0)$

Nun muß noch die Größe des Anstieges an den Anschlußstellen festgelegt werden. Der letzte und der folgende Abtastwert x_{n-1} und x_{n+1} geben den groben Verlauf, d. h. die Tendenz des Signals an der Stelle x_n an; somit legen sie den Anstieg an dieser Stelle fest. Abbildung 1 veranschaulicht diesen Sachverhalt. D. h. der Anstieg an der Stelle x_n ist gleich der Steigung der Geraden durch x_{n-1} und x_{n+1} .

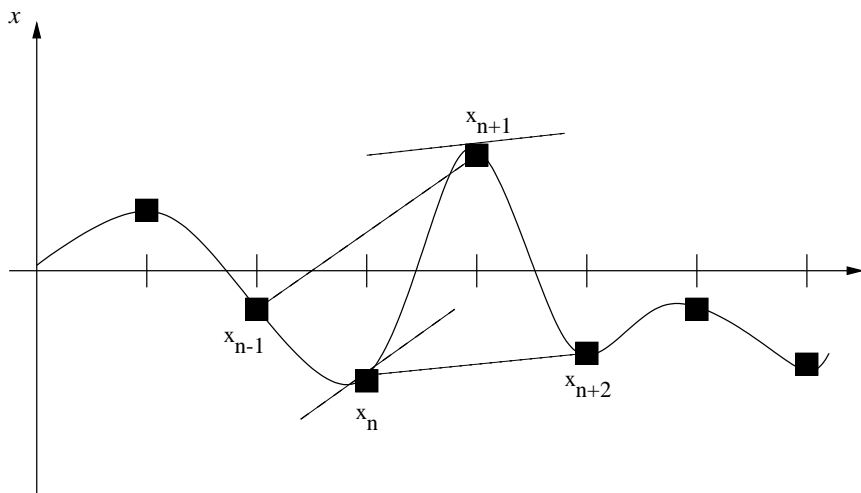


Abbildung 1: Veranschaulichung der Polynom-Anstiege an den Rändern

Die Randbedingungen des Polynoms lauten also:

$$p_n(0) = x_n \tag{1}$$

$$p_n(1) = x_{n+1} \tag{2}$$

$$p'_n(0) = \frac{1}{2}(x_{n+1} - x_{n-1}) \tag{3}$$

$$p'_n(1) = \frac{1}{2}(x_{n+2} - x_n) \tag{4}$$

Da vier Bestimmungsgleichungen vorhanden sind, wird ein Polynom 3. Ordnung benötigt.

$$p_n(\mu) := a_n\mu^3 + b_n\mu^2 + c_n\mu + d_n \quad \Rightarrow \quad p'_n(\mu) = 3a_n\mu^2 + 2b_n\mu + c_n$$

Die Aufstellung des Gleichungssystems liefert die Lösung der Parameter a_n , b_n , c_n und d_n :

$$\begin{aligned} a_n &= \frac{1}{2}(3(x_n - x_{n+1}) + x_{n+2} - x_{n-1}) \\ b_n &= x_{n-1} + 2x_{n+1} - \frac{1}{2}(x_{n+2} + 5x_n) \\ c_n &= \frac{1}{2}(x_{n+1} - x_{n-1}) \\ d_n &= x_n \end{aligned}$$

Zur Berechnung von $x(t_0)=x((n+\mu)T)$ müssen also die Koeffizienten a_n , b_n und c_n berechnet und das Polynom ausgewertet werden:

$$x((n+\mu)T) \approx a_n\mu^3 + b_n\mu^2 + c_n\mu + x_n$$

4.2 Optimierung für Ganzzahlarithmetik

Liegen die Abtastwerte x_n als ganze Zahlen vor (üblich z. B. bei Audiotbearbeitung auf Rechnern) kann die (meist sehr viel schnellere) Ganzzahlarithmetik eines Rechners benutzt werden, indem die Polynomkoeffizienten mit 2 multipliziert werden (womit alle Koeffizienten ganzzahlig werden) und das Ergebnis erst zum Schluss halbiert wird:

$$\begin{aligned} a'_n &= 3(x_n - x_{n+1}) + x_{n+2} - x_{n-1} \\ b'_n &= 2x_{n-1} + 4x_{n+1} - (x_{n+2} + 5x_n) \\ c'_n &= x_{n+1} - x_{n-1} \\ \Rightarrow x((n+\mu)T) &\approx \frac{1}{2}(a'_n\mu^3 + b'_n\mu^2 + c'_n\mu) + x_n \end{aligned}$$

Außerdem kann die Polynomrechnung nach dem HORNER-Schema durchgeführt werden:

$$x((n+\mu)T) \approx \frac{1}{2}[(a'_n\mu + b'_n)\mu + c'_n]\mu + x_n$$

Gleitkommaoperationen: 4 x Multiplikation, 3 x Addition

Ganzzahloperationen: 4 x Multiplikation, 3 x Addition, 4 x Subtraktion

4.3 Optimierung für Fließkommaarithmetik

Obige Optimierungsstrategie hilft wenig, wenn die Abtastwerte als reelle Zahlen (im Fließkommaformat auf Rechnern) vorliegen, da von vornherein keine Ganzzahlarithmetik durchgeführt werden kann.

In dem Fall läßt sich die Anzahl der Rechenschritte um zwei Multiplikationen reduzieren, indem man die Koeffizientengleichungen in die Polynomgleichung einsetzt, ausmultipliziert und nach den Abtastwerten auflöst:

$$x((n+\mu)T) \approx \frac{1}{2} \left[x_{n-1}(2\mu^2 - \mu^3 - \mu) + x_n(3\mu^3 - 5\mu^2 + 2) + x_{n+1}(4\mu^2 - 3\mu^3 + \mu) + x_{n+2}(\mu^3 - \mu^2) \right]$$

μ^2 , μ^3 und $3\mu^3$ können vorher berechnet und gespeichert werden.

Gleitkommaoperationen: 6 x Multiplikation, 4 x Addition, 6 x Subtraktion

5 Bandbreitenbeschränkte Interpolation

5.1 Theorie

SHANNONS Sampling-Theorem besagt, dass ein Signal *exakt* durch seine Abtastwerte rekonstruiert werden kann, wenn es keine Frequenzanteile oberhalb der halben Abtastfrequenz enthält. Daher rührt der Name *bandbreitenbeschränkte* Interpolation, die auf diesem Theorem basiert:

$$x(t) \equiv \sum_{n=-\infty}^{\infty} x_n H(t - nT)$$

Dabei ist $H(x)$ die Stoßantwort des idealen Tiefpaßfilters:

$$H(x) = \operatorname{sinc}\left(\frac{x}{T}\right) \quad \text{mit} \quad \operatorname{sinc}(x) = \frac{\sin(\pi x)}{\pi x}$$

Soll diese Gleichung dazu benutzt werden, die Abtastrate zu verringern (*downsampling*), muß die Grenzfrequenz des Tiefpasses angepaßt werden, um Aliasing-Effekte zu vermeiden:

$$H(x) = \frac{1}{T'} \operatorname{sinc}\left(\frac{t}{T'}\right)$$

T' sei dabei die neue Abtastfrequenz.

5.2 Implementation

6 Vergleich

Die lineare Interpolation ist sehr schnell und leicht zu implementieren und bringt, wenn es nicht um hohe Qualität geht, bereits akzeptable Ergebnisse. Sie ist deshalb vor allem für Echtzeitanwendungen auf langsameren Rechnern vorteilhaft einsetzbar.

Polynomiale Interpolation ist nicht sehr viel aufwändiger und bringt sehr gute Ergebnisse. Soll die Abtastrate reduziert werden (*downsampling*), ist sie auf jeden Fall zu empfehlen. Bei Erhöhung der Abtastrate ist im Allgemeinen ebenfalls die Polynominterpolation ausreichend. Nur wenn es um höchste Qualität geht und der Berechnungsaufwand eine nur untergeordnete Rolle spielt, sollte die bandbreitenbeschränkte Interpolation gewählt werden.