Teoria dei Segnali – Elaborazione multifrequenza

Valentino Liberali

Dipartimento di Fisica
Università degli Studi di Milano
valentino.liberali@unimi.it

UNIVERSITÀ DEGLI STUDI
DI MILANO

Teoria dei Segnali – Elaborazione multifrequenza – 22 novembre 2010

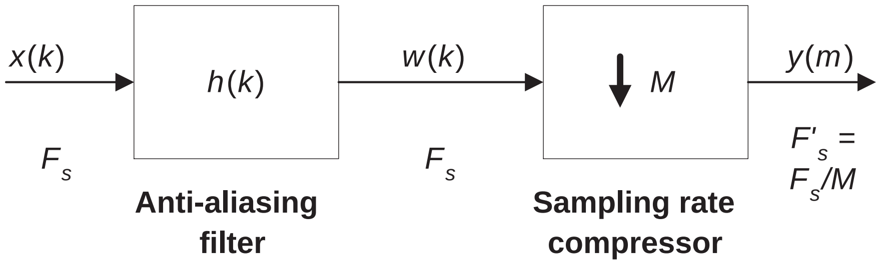
Contenuto

Decimation by an Integer Factor **M**

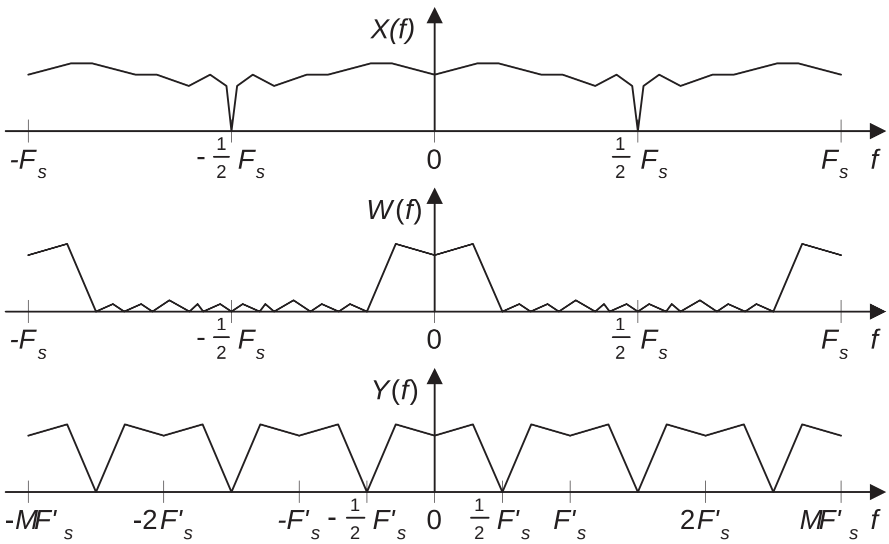
DECIMATION := reduction of sampling rate

The new sampling period is $T' = MT$
The new sampling rate is $F'_s = \frac{F_s}{M}$

→ Sampling theorem requires an ANTI-ALIASING FILTER **before** re-sampling



Sampling Rate Reduction



Typical spectra for decimation by M

Decimation in Practice

Sampling rate compression is done by taking only one sample out of M.
The remaining $M-1$ samples are lost.

A delay z^{-r} in the input sequence $x[k]$ (or in the sequence $w[k]$) modifies the output sequence $y[m]$, unless r is an integer multiple of M.

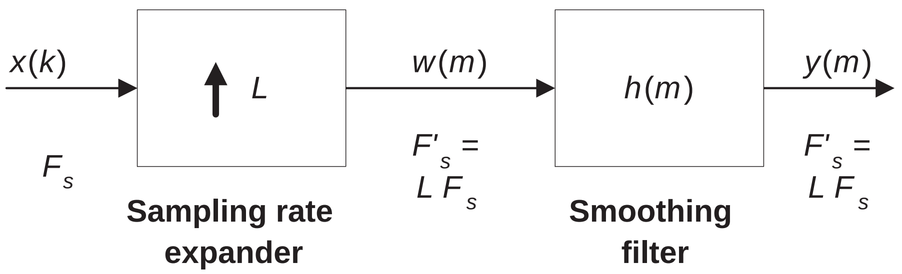
Therefore, decimation is NOT a time-invariant process.

Interpolation by an Integer Factor **L**

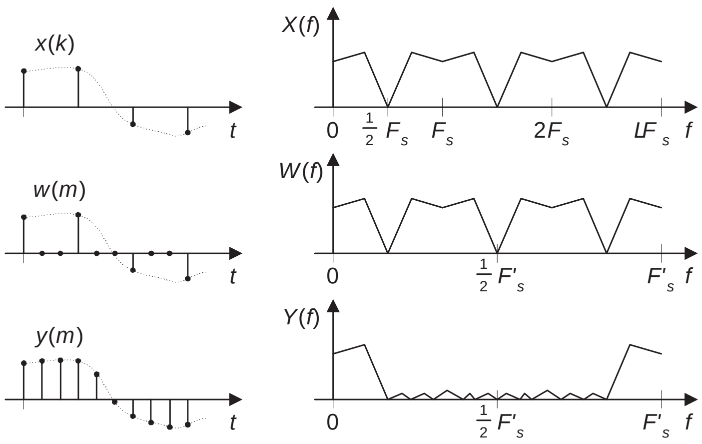
INTERPOLATION := increase of sampling rate

The new sampling period is $T' = \frac{T}{L}$
The new sampling rate is $F'_s = LF_s$
In case of "ideal" sampling, $x\left[\frac{kT}{L}\right] = 0$ when k is not an integer multiple of L (*zero padding*)

$\rightarrow$ a SMOOTHING FILTER is required **after** interpolation



Sampling Rate Increase



Time and frequency representation of interpolation by L

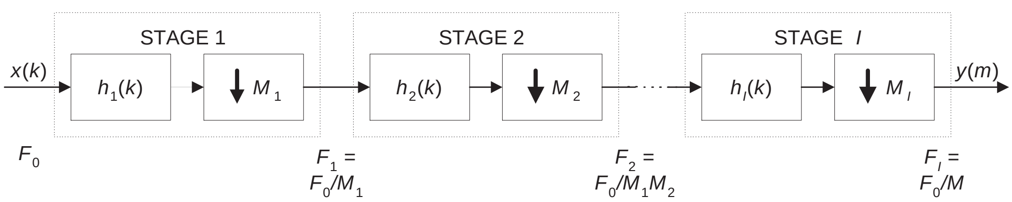
A delay in the input sequence produces the same (delayed) output sequence → interpolation is time-invariant

Multistage Decimators

If the decimation ratio can be factored into the product of integer numbers:

$$M = \prod_{i=1}^{I} M_i$$

then the decimator can be realized with **I independent stages**.



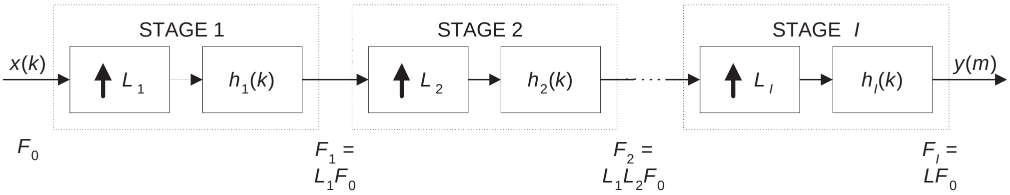
Multistage decimator

Multistage Interpolators

In a similar way, if the interpolation ratio can be factored:

$$L = \prod_{i=1}^{I} L_i$$

then the interpolator can be realized with **I independent stages**.



Multistage interpolator

PROs and CONs of multistage

PRO		CON
☺	**Simple filter stages with reduced computation**	
☺	**Reduced storage**	
☺	**Reduced finite word-length effects**	
	Increased control structures	☹
	Choice of optimum number of stages I	☹

Esempio (1/8)

L'esempio presentato contiene un'immagine tratta dal cartone animato "Duck Dodgers in the $24\frac{1}{2}^{\text{th}}$ century"
(in italiano: "Daffy Rogers nel 24° secolo e un pezzo e mezzo"), prodotto dalla Warner Bros nel 1953 ed interpretato da una grande star del cinema d'animazione: Daffy Duck.

Esempio (2/8)

immagine a colori di 480 × 640 pixel

Esempio (3/8)

Un'immagine è una matrice di pixel.
Ogni pixel è costituito da una terna di numeri interi che indicano il contenuto di ciascuno dei tre colori primari rosso, verde e blu (in inglese: **RGB** = Red, Green, Blue).
In pratica, un'immagine può essere pensata come una matrice tridimensionale a tre "strati": ogni strato contiene le informazioni su un colore.
Solitamente, ciascun colore è codificato con 8 bit senza segno, e assume i valori da 0 (totale assenza del colore) a 255 (massima intensità del colore).

(0, 0, 0) = NERO	(255, 0, 0) = ROSSO
(0, 255, 0) = VERDE	(0, 0, 255) = BLU
(255, 255, 0) = GIALLO	(255, 255, 255) = BIANCO

Esempio (4/8)


R

G

B

Y

$$Y = (R + G + B + 1) / 3$$

Esempio (5/8)

immagini ridotte a 240 × 320 pixel

senza filtraggio

con filtro a media mobile

L'immagine a sinistra è ottenuta semplicemente eliminando tutti i pixel delle righe e delle colonne pari.
Nell'immagine a destra, ogni pixel risulta dalla media aritmetica di 4 pixel dell'immagine originale.

Esempio (6/8)

immagine a colori di 720 × 960 pixel, con *zero-padding*

Esempio (7/8)

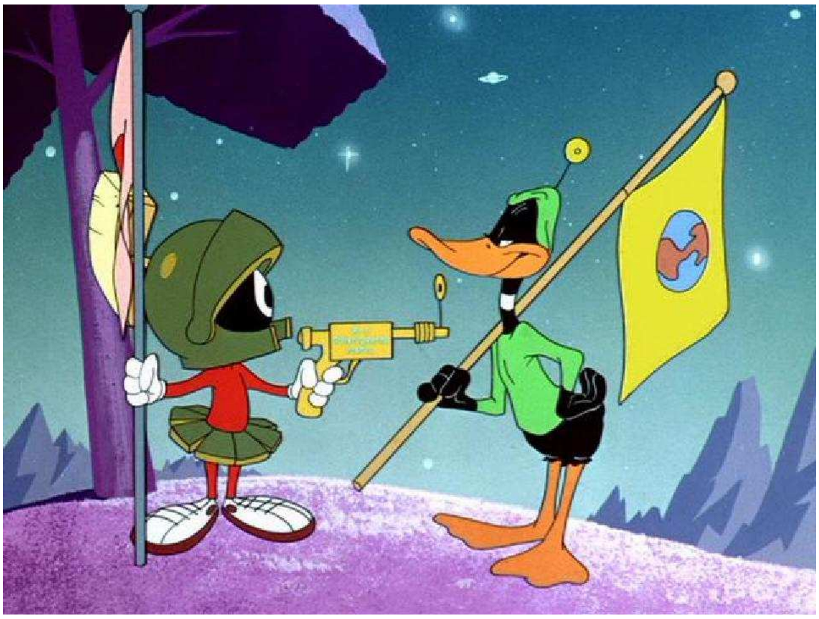


immagine a colori di 720 × 960 pixel, con *holding*

Esempio (8/8)

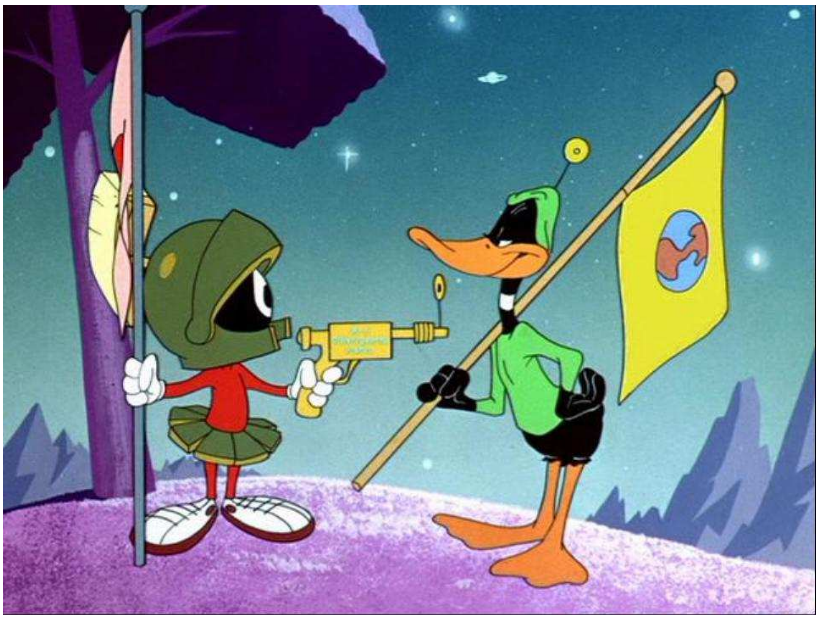


immagine a colori di 720 × 960 pixel, con filtro lineare