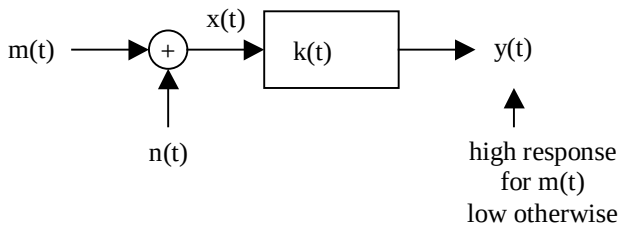
Lecture notes: Matched filter, Wiener filter

A previous lecture introduced template matching and matched spatial filtering. A matched filter process can also be modeled as follows:



where m(t) is the desired signal (the thing to be found, or matched), n(t) is a noise signal (or background or clutter), x(t) is the observed signal (or sensor output), k(t) is the matched filter, and y(t) is the desired output.

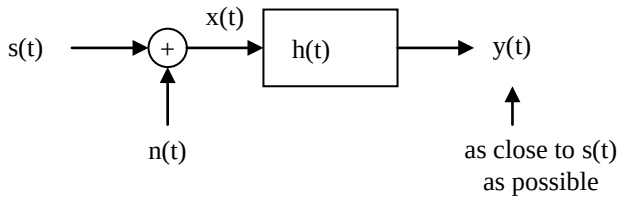
As we saw last time, the "best" filter k(t) is a copy (reflection) of the desired signal m(t). However, various **problems** affect the performance of matched filtering, including (a) variation in the signal m(t) to be detected, (b) occlusion of the signal of interest, and (c) non-Gaussian noise. This lecture covers methods that can be used to help overcome these problems.

First, the template does not always have to come from an **example** of the input. Instead it can be generated using a mathematical **model**. An advantage is that the model can help express the variability in the expected appearance of the signal. At the web site is a paper on using matched spatial filters for finding blood vessels in a retinal image. In this case, the filter was created by developing a model of a blood vessel from an inverted Gaussian.

Second, a **filter bank** can help overcome variance in the desired signal, by giving templates for several desired variations. The paper on blood vessel detection shows an example of using 16 filters for finding blood vessels. Each filter describes a small segment of blood vessel in another orientation, where the 16 orientations span 360 degrees.

Another method is to use **deformable templates**. Deformable templates are created by warping a given template using a geometric function. The function is selected in an attempt to mimic the warpings expected in the desired signal. For example, in handwriting one can expect variations in the slanting of the written characters. This is demonstrated in another paper posted at the web site. Note in this case, the model is of the expected deformations rather than the underlying signal appearance.

Closely related to the matched filter is the **Wiener filter**, which can be modeled as:



The Wiener filter is applied when the original signal s(t) is corrupted by noise that is unknown and does not satisfy any of the "standard" models (Gaussian, salt and pepper, etc.). We wish to construct a filter custom designed to mitigate the noise in an image/signal, restoring the signal to as close an approximation to the original as is possible. How do we design h(t) for this purpose?

The Wiener filter can be designed when we do not know the noise signal n(t), but we can measure its power. The Wiener filter is implemented via convolution after constructing it in the following steps:

1. Obtain a sample of the input signal s(t).
2. Autocorrelate the input signal:
$$R_s(\tau) = \int_{-\infty}^{+\infty} s(\tau)s(t+\tau)dt$$
3. Calculate the Fourier transform of the autocorrelated signal. The symbol f indicates the frequency space.
$$P_s(f) = \mathcal{F}(R_s(\tau))$$
4. Obtain a sample of the input signal without noise (or build a sample from a model, or build it from an average of multiple samples). Call this sample x(t) (see above figure).
5. Cross-correlate the noise-less signal with the actual signal.
$$R_{xs}(\tau) = \int_{-\infty}^{+\infty} s(\tau)x(t+\tau)d\tau$$
6. Calculate the Fourier transform of the result.
$$P_{xs}(f) = \mathcal{F}(R_{xs}(\tau))$$
7. Calculate the transfer function:
$$H(s) = \frac{P_{xs}(f)}{P_s(f)}$$
8. Calculate the filter:
$$h(t) = \mathcal{F}^{-1}(H(s))$$

The filter produced by the last step can be used in convolution with new instances of the signal s(t) to reduce the noise in the data. At the website is an excerpt from Castleman's textbook showing an example.

In practice, the two most common problems to which Wiener filtering is applied are motion blur and focus blur. The following image (from Wikipedia) shows an example of motion blur:

Original (left), with motion blur (middle), after Wiener filter (right)

The following shows another example (courtesy San Diego State Univ.)

Original image | Blurred image | Image restored by Wiener filter

In practice, the Wiener transform for smoothing is most useful when the **noise spectrum is constant**. Mathematically it is interesting because convolution, filtering, and the Fourier transform have relationships that guide its construction, as shown above. But in practice, a reasonably-sized Gaussian kernel will perform nearly the same for many problems, with the exception of motion and focus blur.