

# Formula Based Performance Design

by *Gerrit Muller* HSN-NISE

e-mail: `gaudisite@gmail.com`

`www.gaudisite.nl`

## Abstract

Performance models are mostly simple mathematical formulas. The challenge is to model the performance at an appropriate level. In this presentation we introduce several levels of modeling, labeled zeroth order, second order, et cetera. AS illiustration we use the performance of MRI reconstruction.

# Theory Block: n Order Formulas

**0<sup>th</sup> order**    main function    *order of magnitude*  
parameters  
*relevant for main function*

**1<sup>st</sup> order**    add overhead    *estimation*  
secondary function(s)

**2<sup>nd</sup> order**    interference effects    *main function, overhead*  
circumstances    *and/or secondary functions*  
*more accurate, understanding*

# CPU Time Formula Zero Order

---

$$t_{\text{cpu total}} = t_{\text{cpu processing}} + t_{\text{UI}}$$

$$t_{\text{cpu processing}} = n_x * n_y * t_{\text{pixel}}$$

# CPU Time Formula First Order

---

$$t_{\text{cpu total}} = t_{\text{cpu processing}} + t_{\text{UI}}$$

$$+ t_{\text{context switch overhead}}$$

# CPU Time Formula Second Order

---

$$t_{\text{cpu total}} = t_{\text{cpu processing}} + t_{\text{UI}} + t_{\text{context switch overhead}} +$$

$$t_{\text{stall time due to cache efficiency}} + t_{\text{stall time due to context switching}}$$

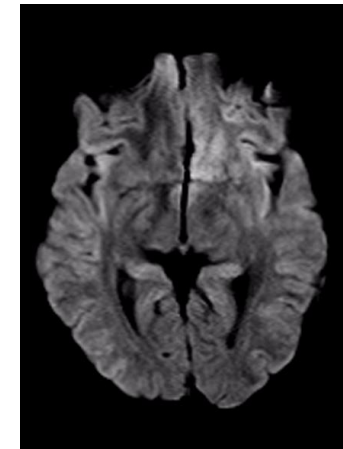
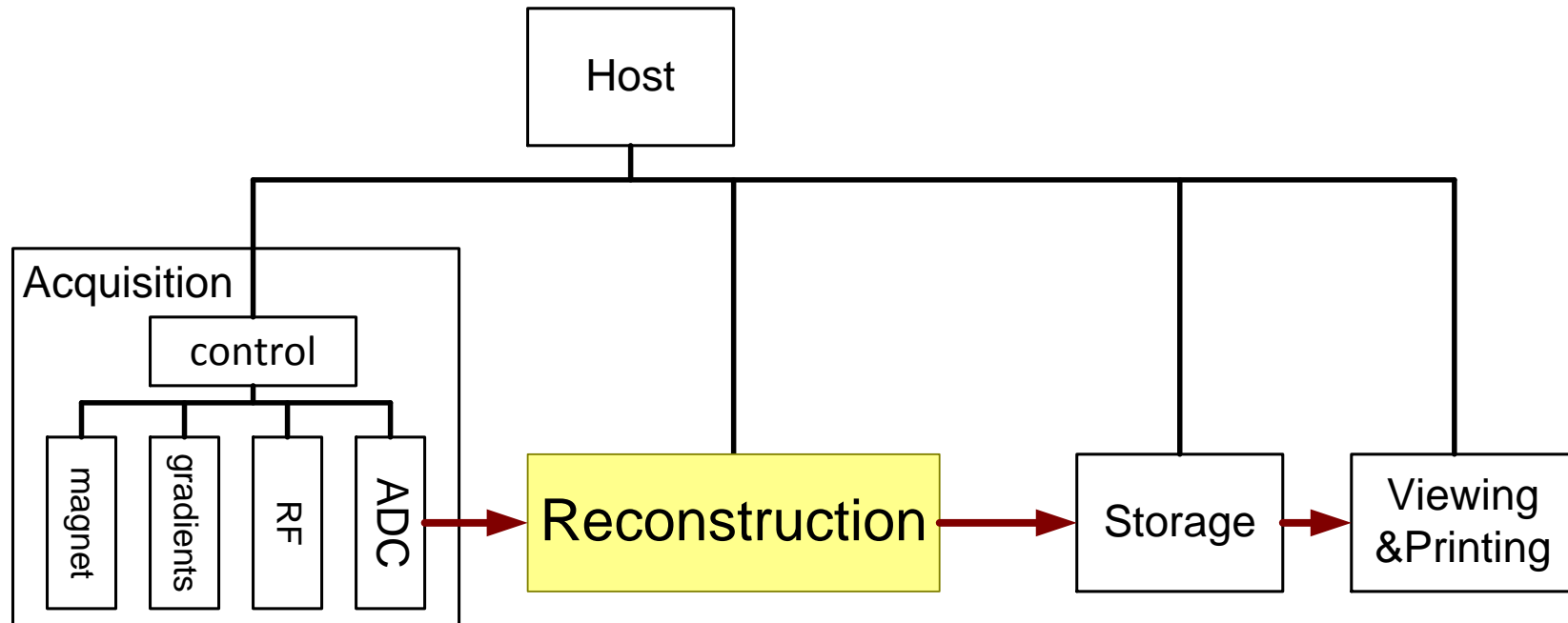
signal processing: high efficiency  
control processing: low/medium efficiency

## *MRI reconstruction*

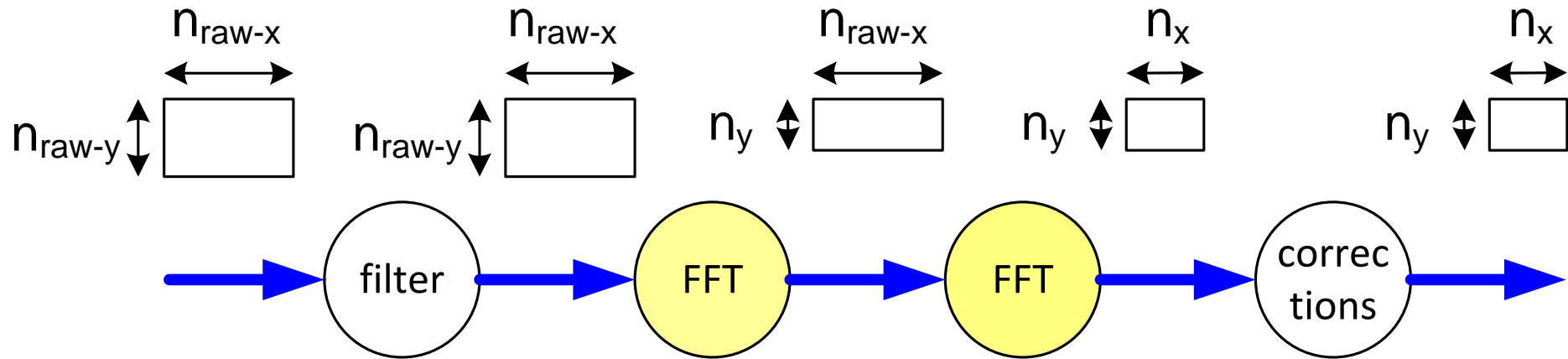
"Test" of performance model on another case

Scope of performance and significance of impact

# MR Reconstruction Context



# MR Reconstruction Performance Zero Order



$$t_{\text{recon}} = n_{\text{raw-x}} * t_{\text{fft}}(n_{\text{raw-y}}) + n_{\text{y}} * t_{\text{fft}}(n_{\text{raw-x}})$$

$$t_{\text{fft}}(n) = c_{\text{fft}} * n * \log(n)$$

# Zero Order Quantitative Example

---

Typical FFT, 1k points ~ 5 msec  
( scales with  $2 * n * \log(n)$  )

using:

$$n_{\text{raw-x}} = 512$$

$$n_{\text{raw-y}} = 256$$

$$n_x = 256$$

$$n_y = 256$$

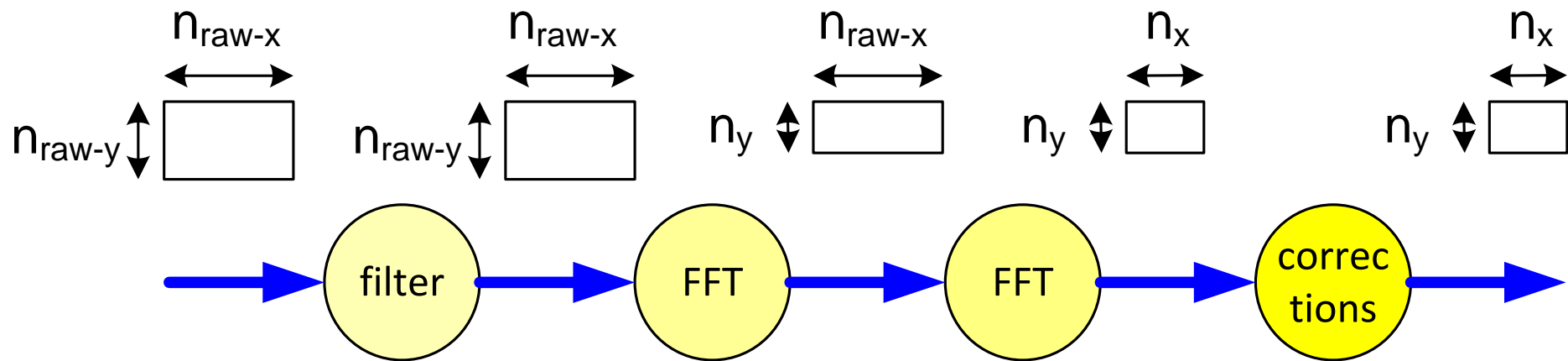
$$t_{\text{recon}} = n_{\text{raw-x}} * t_{\text{fft}}(n_{\text{raw-y}}) +$$

$$n_y * t_{\text{fft}}(n_{\text{raw-x}}) +$$

$$512 * 1.2 + 256 * 2.4$$

$$\approx 1.2 \text{ s}$$

# MR Reconstruction Performance First Order



$$t_{\text{recon}} = t_{\text{filter}}(n_{\text{raw-x}}, n_{\text{raw-y}}) + n_{\text{raw-x}} * t_{\text{fft}}(n_{\text{raw-y}}) + n_y * t_{\text{fft}}(n_{\text{raw-x}}) + t_{\text{corrections}}(n_x, n_y)$$

$$t_{\text{fft}}(n) = c_{\text{fft}} * n * \log(n)$$

# First Order Quantitative Example

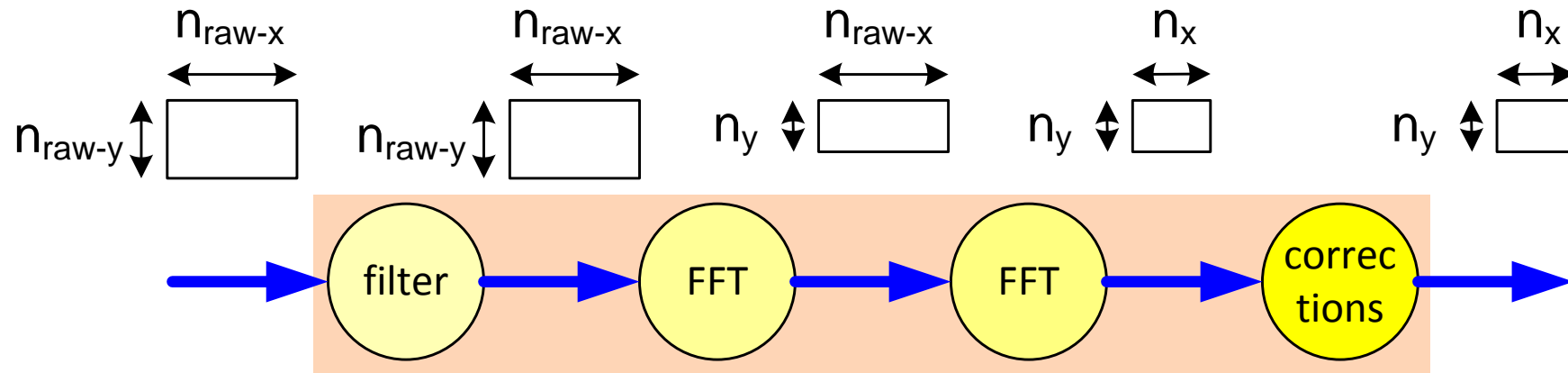
---

Typical FFT, 1k points ~ 5 msec  
( scales with  $2 * n * \log(n)$  )

Filter 1k points ~ 2 msec  
( scales linearly with  $n$  )

Correction ~ 2 msec  
( scales linearly with  $n$  )

# MR Reconstruction Performance Second Order



$$\begin{aligned}
 t_{\text{recon}} = & t_{\text{filter}}(n_{\text{raw-x}}, n_{\text{raw-y}}) + \\
 & n_{\text{raw-x}} * ( t_{\text{fft}}(n_{\text{raw-y}}) + t_{\text{col-overhead}} ) + \\
 & n_y * ( t_{\text{fft}}(n_{\text{raw-x}}) + t_{\text{row-overhead}} ) + \\
 & t_{\text{corrections}}(n_x, n_y) + \\
 & t_{\text{control-overhead}}
 \end{aligned}$$

$$t_{\text{fft}}(n) = c_{\text{fft}} * n * \log(n)$$

# Second Order Quantitative Example

---

Typical FFT, 1k points ~ 5 msec

( scales with  $2 * n * \log(n)$  )

Filter 1k points ~ 2 msec

( scales linearly with  $n$  )

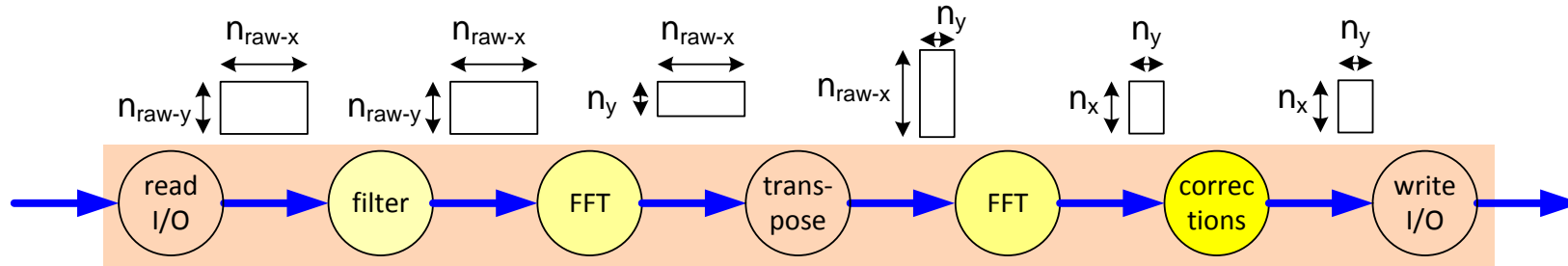
Correction ~ 2 msec

( scales linearly with  $n$  )

Control overhead =  $n_y * t_{\text{row overhead}}$

 10 .. 100  $\mu\text{s}$

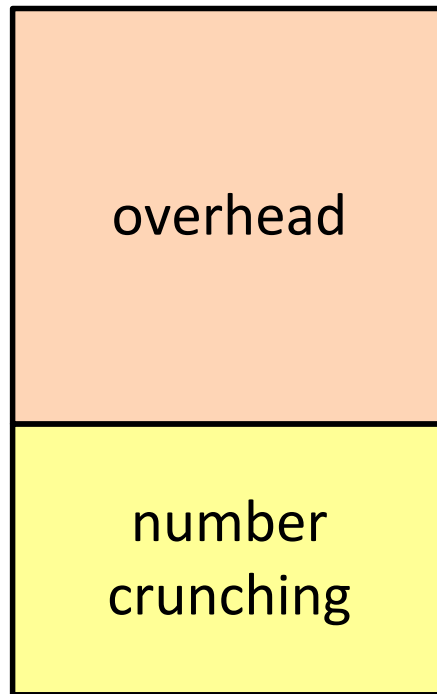
# MR Reconstruction Performance Third Order



$$t_{recon} = t_{filter}(n_{raw-x}, n_{raw-y}) + n_{raw-x} * (t_{fft}(n_{raw-y}) + t_{col-overhead}) + t_{fft}(n) = c_{fft} * n * \log(n)$$

$$n_y * (t_{fft}(n_{raw-x}) + t_{row-overhead}) + t_{corrections}(n_x, n_y) + t_{read\ I/O} + t_{transpose} + t_{write\ I/O} + t_{control-overhead}$$

bookkeeping
transpose
malloc, free
write I/O
read I/O
overhead
correction computations
row overhead
FFT computations
column overhead
FFT computations
overhead
filter computations



focus on overhead reduction  
 is more important  
 than faster algorithms  
 this is not an excuse  
 for sloppy algorithms

## *MRI reconstruction*

System performance may be determined by other than standard facts

E.g. more by overhead I/O rather than optimized core processing

==> Identify & measure what is performance-critical in application

The ASP™ course is partially derived from the EXARCH course developed at *Philips CTT* by *Ton Kostelijk* and *Gerrit Muller*.

Extensions and additional slides have been developed at *ESI* by *Teun Hendriks*, *Roland Mathijssen* and *Gerrit Muller*.