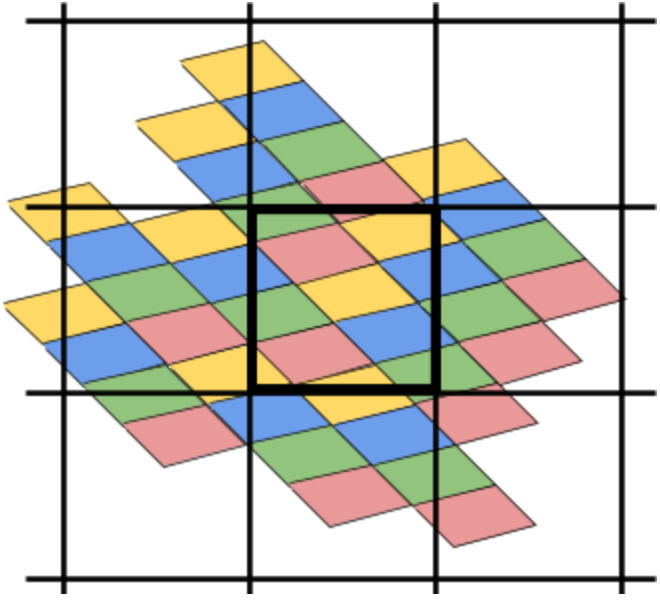
Filtering the pixel footprint: what should be “the right filter” ?

BTW, why do we filter ? What purpose ?



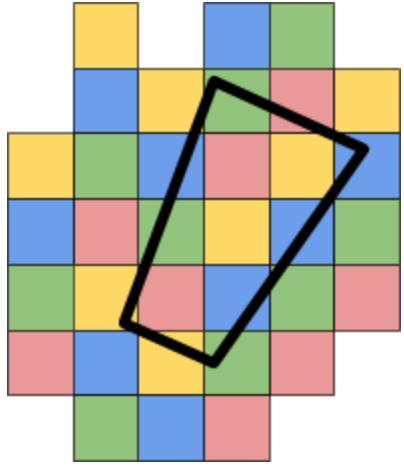
Too much / too small data in one pixel

One value to ~~rule~~ represent them all

Average on the pixel footprint

- at run time (path tracing)
- precomputation

(MIP-map)



Wait a minute. “Average” ? “Footprint” ? What does it mean ?

- average = normalized integral. (“sum everything, divide by pixel size”).
- things weighted by contribution to the pixel (e.g. apparent surface)

→ what should contribute to a pixel ?

What physical are we trying to simulate here ?

Indeed, what are we trying to do ?

- “Uh, we just want less data and less calculation, just arbitrary choices !”
- “Uh, we just want to match the ground truth !”
- “Uh, we want reality, like these windows with small translucent tiles, u’know ?
  Just add equally all and only what’s in the small squares !”
- “we want to simulate what a camera physically see (so do as above) !”
- “We want to see what the eye see (as input), (so as above) !”
- “We want to sample and reconstruct an aliasing-free signal: just apply Signal Theory”
- “We want the best looking image: max contrast but no aliasing or artifacts”

Indeed, what are we trying to do ?

- “Uh, we just want less data and less calculation, just arbitrary choices !”
  Then, you’ll never match the ground truth → color sliding at zoom.
- “Uh, we just want to match the ground truth !”

- “Uh, we want reality, like these windows with small translucent tiles, u’know ?
  Just add equally all and only what’s in the small squares !”

- “we want to simulate what a camera physically see (so do as above) !”

- “We want to see what the eye see (as input), (so as above) !”

- “We want to sample and reconstruct an aliasing-free signal: just apply Signal Theory”

- “We want the best looking image: max contrast but no aliasing or artifacts”

Indeed, what are we trying to do ?

- “Uh, we just want less data and less calculation, just arbitrary choices !”
  Then, you’ll never match the ground truth → color sliding at zoom.
- “Uh, we just want to match the ground truth !”
  Well, same question applies to the ground truth: pixel-averaged how ?
- “Uh, we want reality, like these windows with small translucent tiles, u’know ?
  Just add equally all and only what’s in the small squares !”

- “we want to simulate what a camera physically see (so do as above) !”

- “We want to see what the eye see (as input), (so as above) !”

- “We want to sample and reconstruct an aliasing-free signal: just apply Signal Theory”

- “We want the best looking image: max contrast but no aliasing or artifacts”

Indeed, what are we trying to do ?

- "Uh, we just want less data and less calculation, just arbitrary choices !"
  Then, you'll never match the ground truth → color sliding at zoom.
- "Uh, we just want to match the ground truth !"
  Well, same question applies to the ground truth: pixel-averaged how ?
- "Uh, we want reality, like these windows with small translucent tiles, u'know ?
  Just add equally all and only what's in the small squares !"
  This is Box filter. Cause aliasing. Glint jumps one pixel when x: 17.999→ 18.000
- "we want to simulate what a camera physically see (so do as above) !"

- "We want to see what the eye see (as input), (so as above) !"

- "We want to sample and reconstruct an aliasing-free signal: just apply Signal Theory"

- "We want the best looking image: max contrast but no aliasing or artifacts"

Indeed, what are we trying to do ?

- “Uh, we just want less data and less calculation, just arbitrary choices !”
  Then, you’ll never match the ground truth → color sliding at zoom.
- “Uh, we just want to match the ground truth !”
  Well, same question applies to the ground truth: pixel-averaged how ?
- “Uh, we want reality, like these windows with small translucent tiles, u’know ?
  Just add equally all and only what’s in the small squares !”
  This is Box filter. Cause aliasing. Glint jumps one pixel when x: 17.999→ 18.000
- “we want to simulate what a camera physically see (so do as above) !”
  Are you sure you know how CCD captor are made ? (CoC, microlenses)
- “We want to see what the eye see (as input), (so as above) !”
  Are you sure you know how eye captor is made ? (CoC, multi-layer diffusion)
- “We want to sample and reconstruct an aliasing-free signal: just apply Signal Theory”

- “We want the best looking image: max contrast but no aliasing or artifacts”

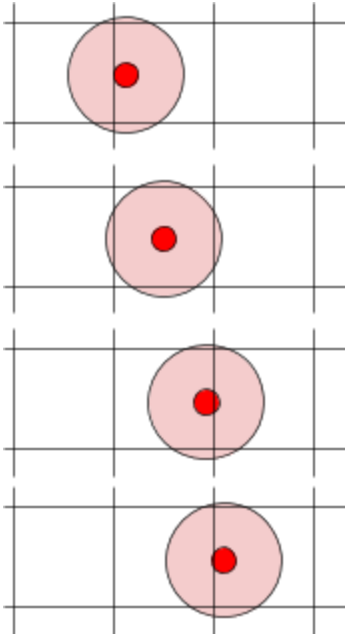
Indeed, what are we trying to do ?

- "Uh, we just want less data and less calculation, just arbitrary choices !"
  Then, you'll never match the ground truth → color sliding at zoom.
- "Uh, we just want to match the ground truth !"
  Well, same question applies to the ground truth: pixel-averaged how ?
- "Uh, we want reality, like these windows with small translucent tiles, u'know ?
  Just add equally all and only what's in the small squares !"
  This is Box filter. Cause aliasing. Glint jumps one pixel when x: 17.999→ 18.000
- "we want to simulate what a camera physically see (so do as above) !"
  Are you sure you know how CCD captor are made ? (CoC, microlenses)
- "We want to see what the eye see (as input), (so as above) !"
  Are you sure you know how eye captor is made ? (CoC, multi-layer diffusion)
- "We want to sample and reconstruct an aliasing-free signal: just apply Signal Theory"
  This is Sinc filter. The thing with negative + overshot lobes, right ?
- "We want the best looking image: max contrast but no aliasing or artifacts"

Indeed, what are we trying to do ?

- "Uh, we just want less data and less calculation, just arbitrary choices !"
  Then, you'll never match the ground truth → color sliding at zoom.
- "Uh, we just want to match the ground truth !"
  Well, same question applies to the ground truth: pixel-averaged how ?
- "Uh, we want reality, like these windows with small translucent tiles, u'know ?
  Just add equally all and only what's in the small squares !"
  This is Box filter. Cause aliasing. Glint jumps one pixel when x: 17.999→ 18.000
- "we want to simulate what a camera physically see (so do as above) !"
  Are you sure you know how CCD captor are made ? (CoC, microlenses)
- "We want to see what the eye see (as input), (so as above) !"
  Are you sure you know how eye captor is made ? (CoC, multi-layer diffusion)
- "We want to sample and reconstruct an aliasing-free signal: just apply Signal Theory"
  This is Sinc filter. The thing with negative + overshot lobes, right ?
- "We want the best looking image: max contrast but no aliasing or artifacts"
  Uh, can you state that a bit more mathematically ? (perceptual ?)

Indeed, what are we trying to do ?



Small ‘impulse’ content impacts as a blob pixel-size (at least)

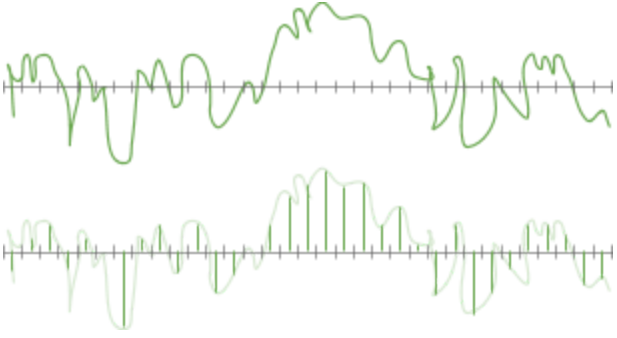
As it approach next pixel, continuous transition

- no aliasing
- perceptual continuity (artifacts = false features)

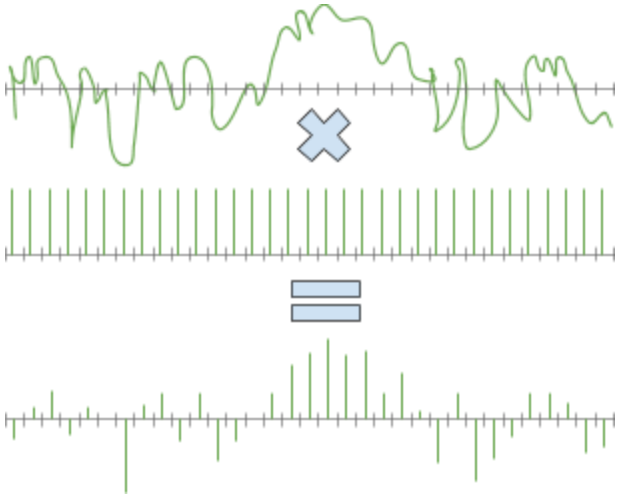
Basically reproduces real optics (lense + pre-captor)

→ So we need a Kernel (filter). Which ?

Smooth sampling/reconstruction: Signal theory

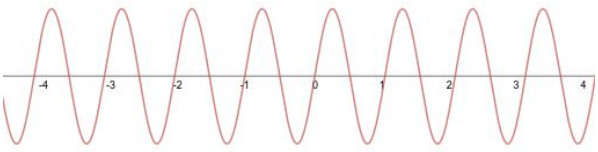


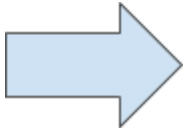
Smooth sampling/reconstruction: Signal theory

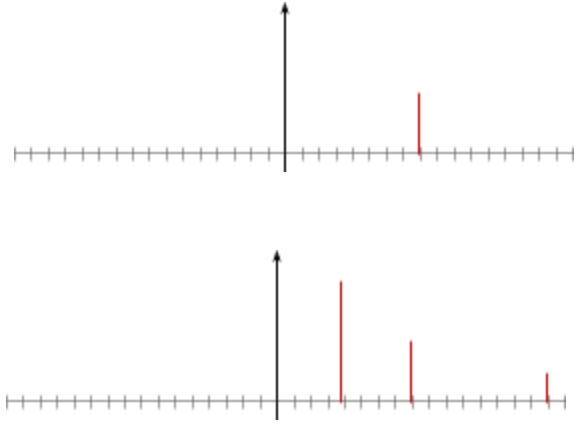


Smooth sampling/reconstruction: Signal theory

The Fourier Transform :



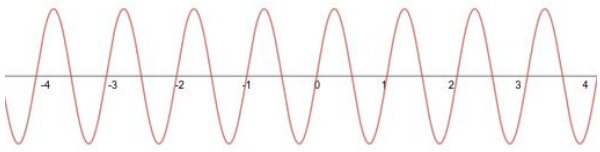


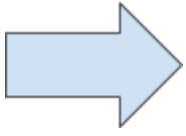


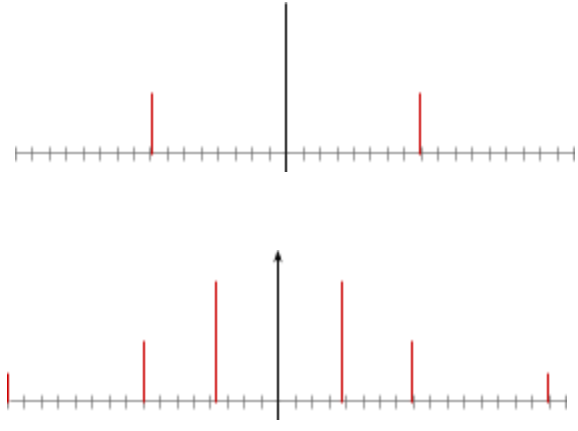
+ phase: $A_f.(\cos(\phi_f) + i\sin(\phi_f)) = A_f e^{i\phi_f}$

Smooth sampling/reconstruction: Signal theory

The Fourier Transform :

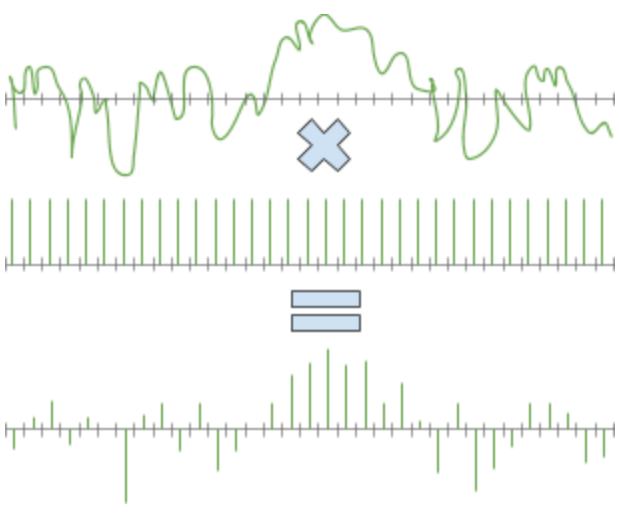


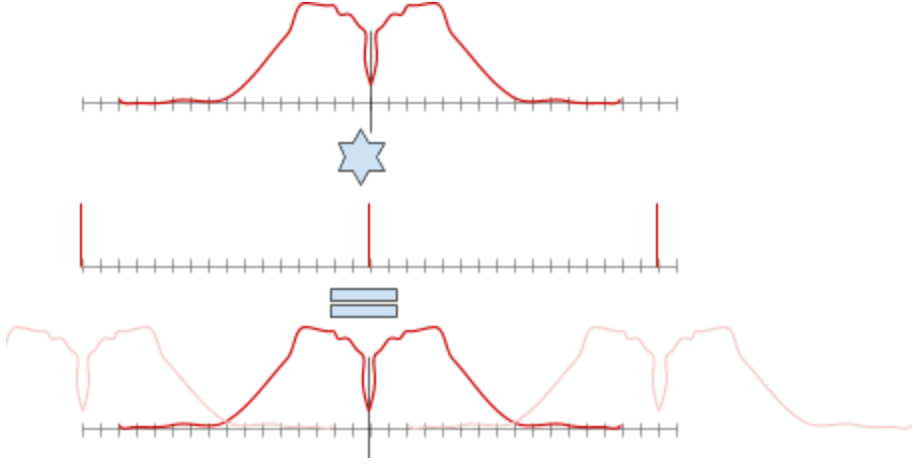




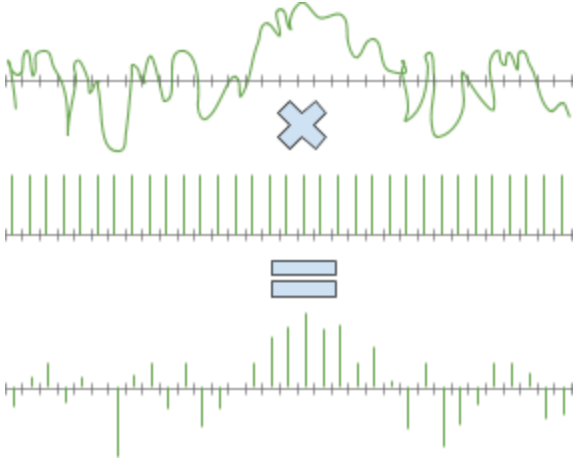
Smooth sampling/reconstruction: Signal theory

Sampling :

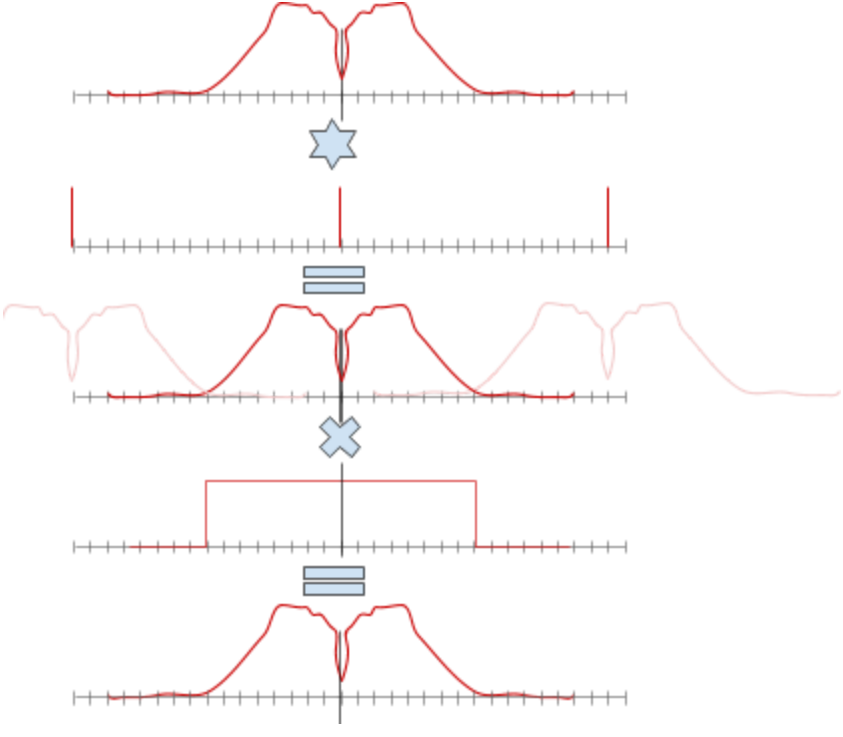




Smooth sampling/reconstruction: Signal theory

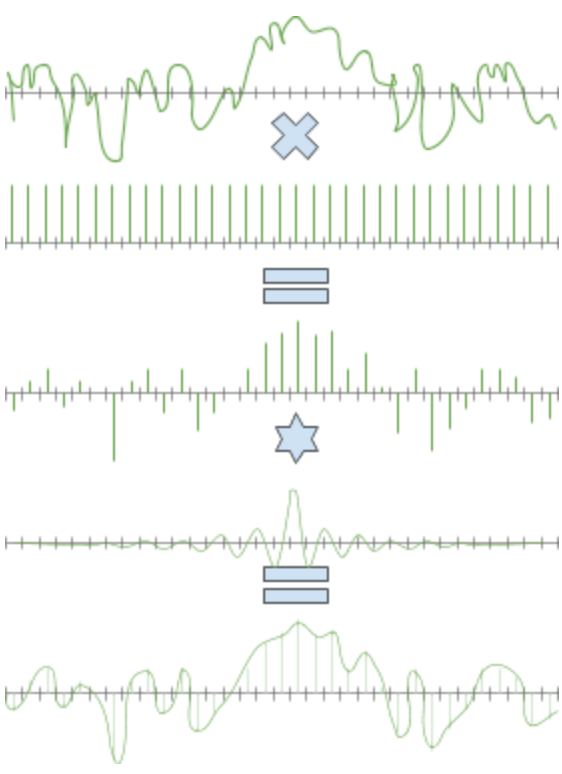


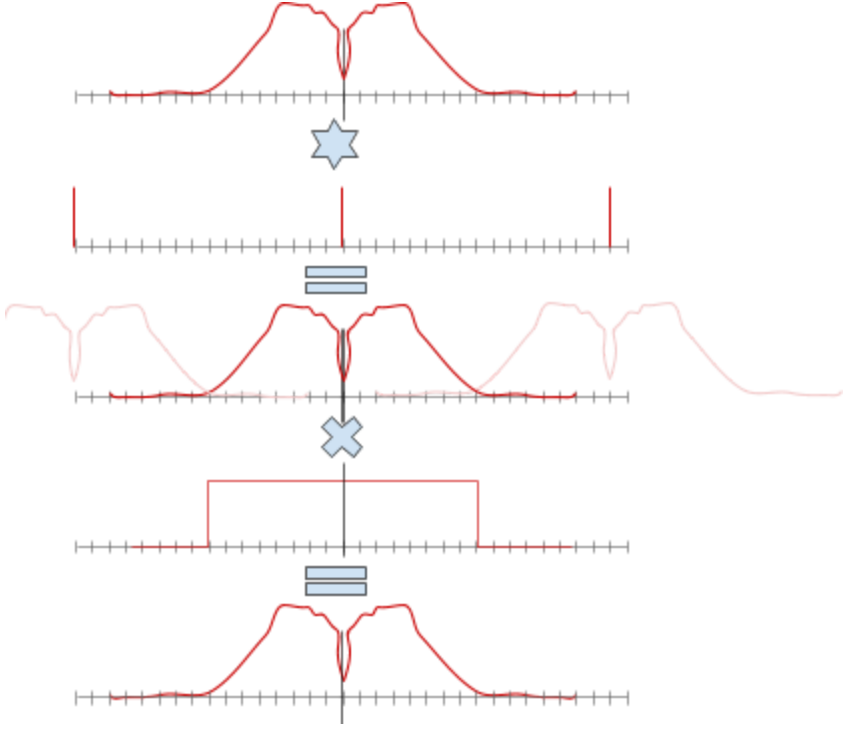
Reconstruction :



Smooth sampling/reconstruction: Signal theory

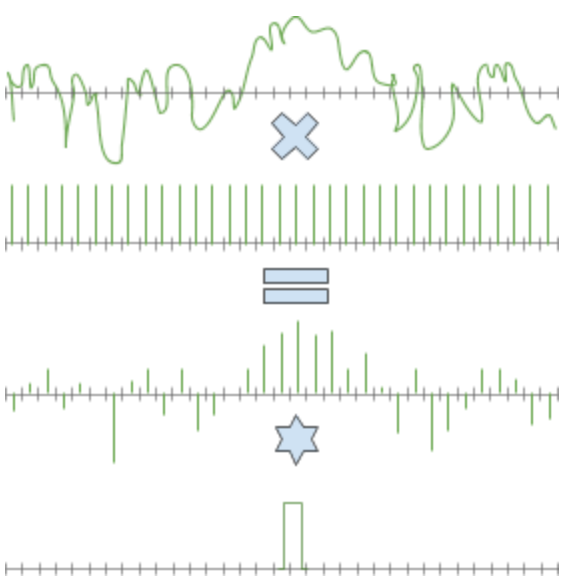
Reconstruction :

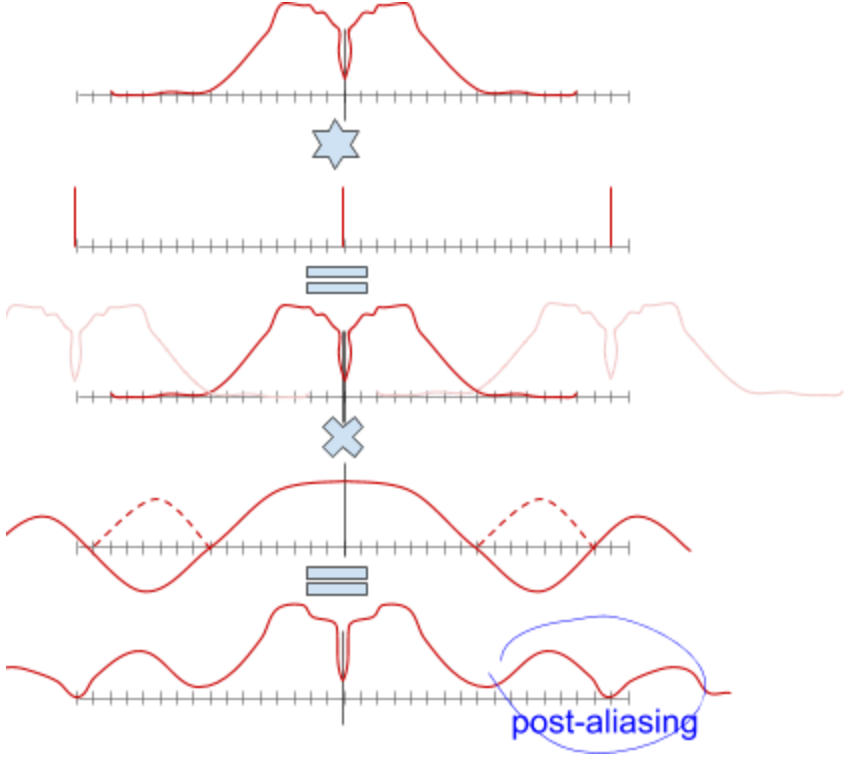




Smooth sampling/reconstruction: Signal theory

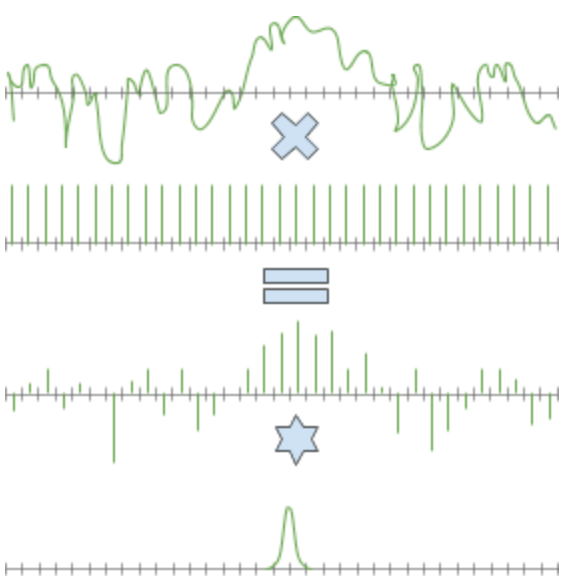
What when using Box filter :

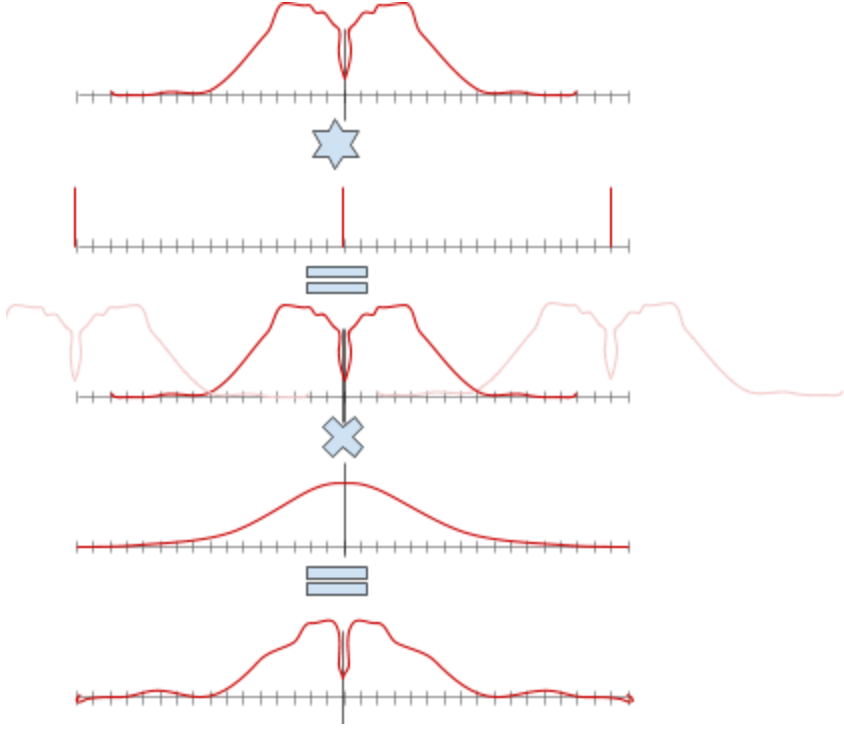




Smooth sampling/reconstruction: Signal theory

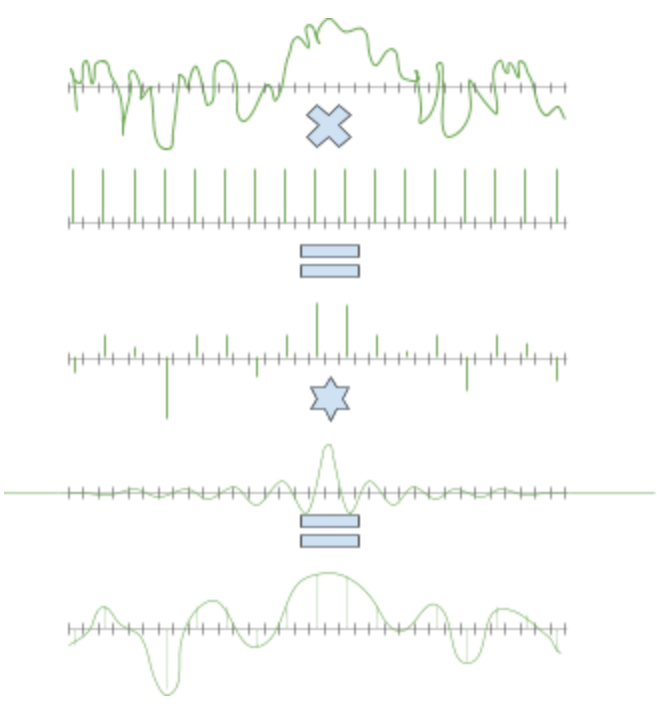
What when using Gaussian filter :

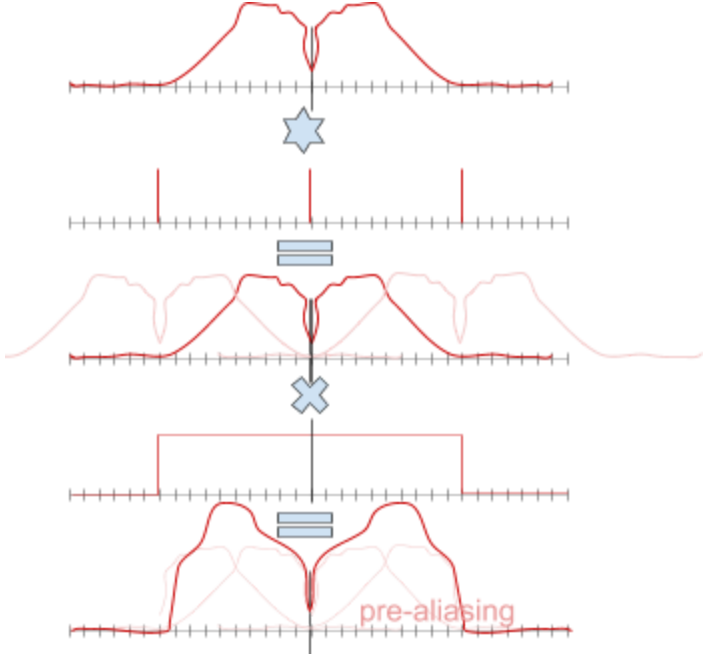




Smooth sampling/reconstruction: Signal theory

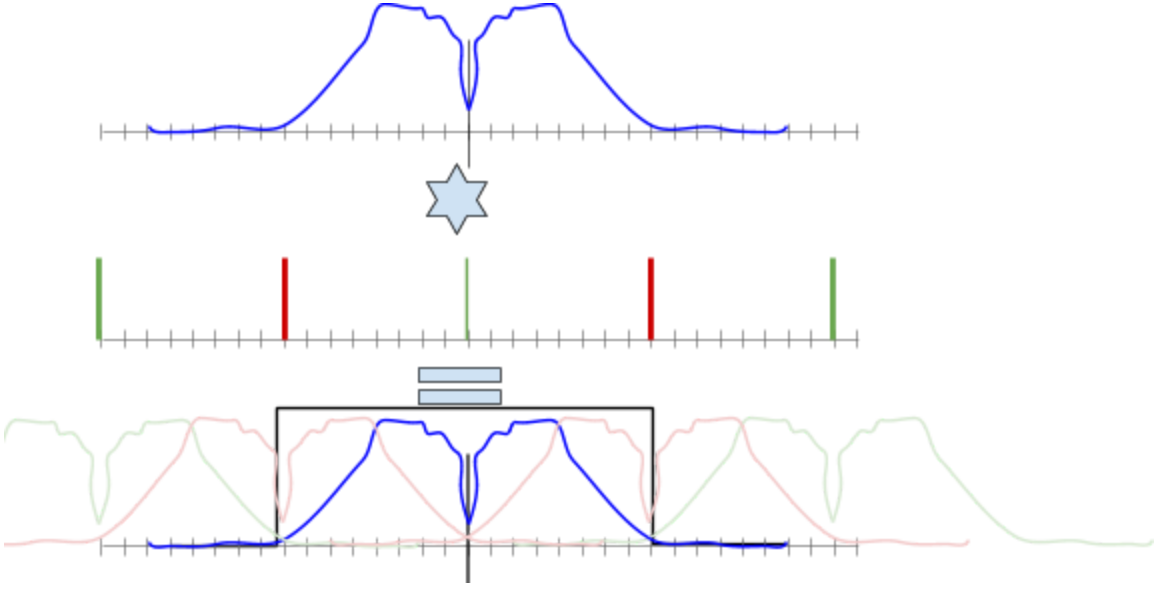
Back to Sinc: What with too low sampling :





Smooth sampling/reconstruction: Signal theory

Shannon-Nyquist condition *: sampling at least* ***twice*** *the max freq of signal*



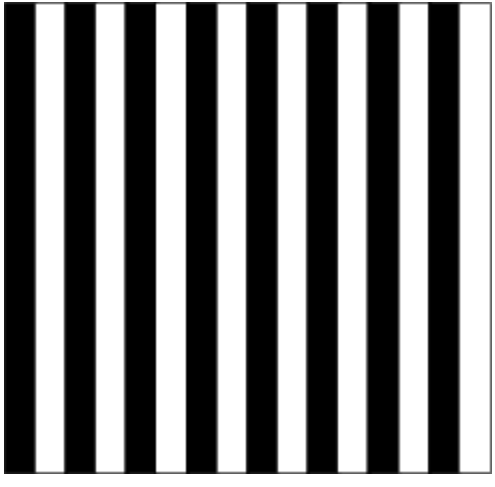
Smooth sampling/reconstruction: Signal theory

`Parenthesis:`

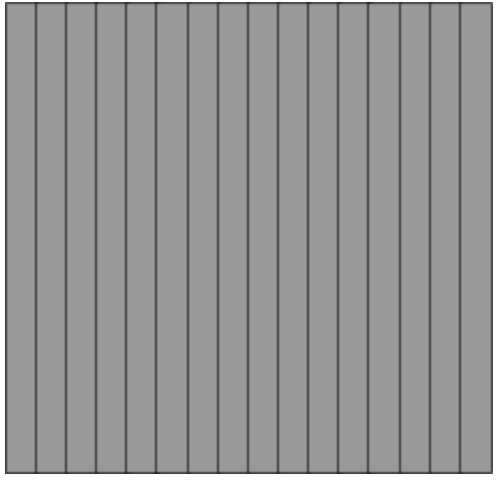
NB: this is **already** aliasing:

Smooth sampling/reconstruction: Signal theory

NB: this is **already** aliasing:



Proof: if grid under the paint offsetted by ½ pixel:
(same for rotation)

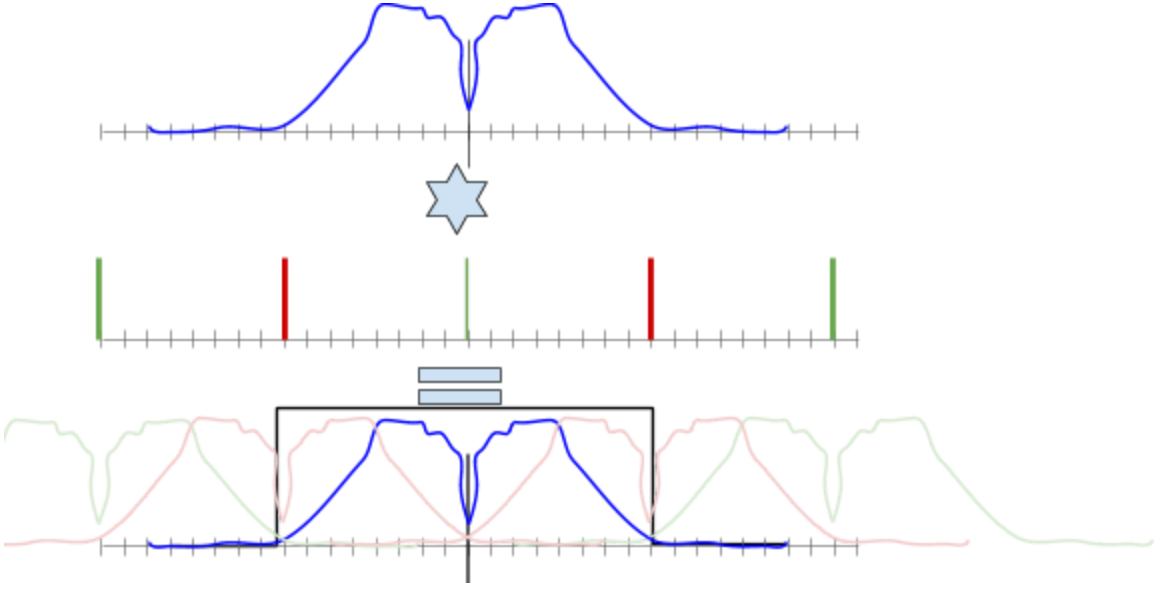


Good sampling : content identical whatever the sampling translation and rotation.

`:end parenthesis`

Smooth sampling/reconstruction: Signal theory

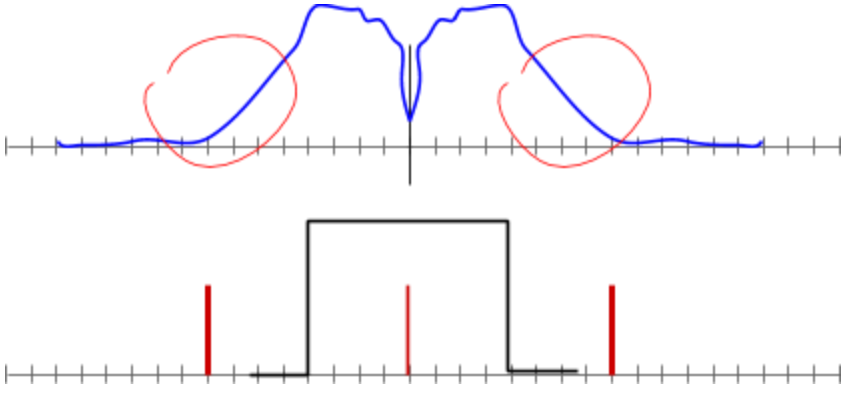
Shannon-Nyquist condition: sampling at least **twice** the max freq of signal



Filtering way: (pre)filter data so that *signal max freq **half** or less than sampling freq*

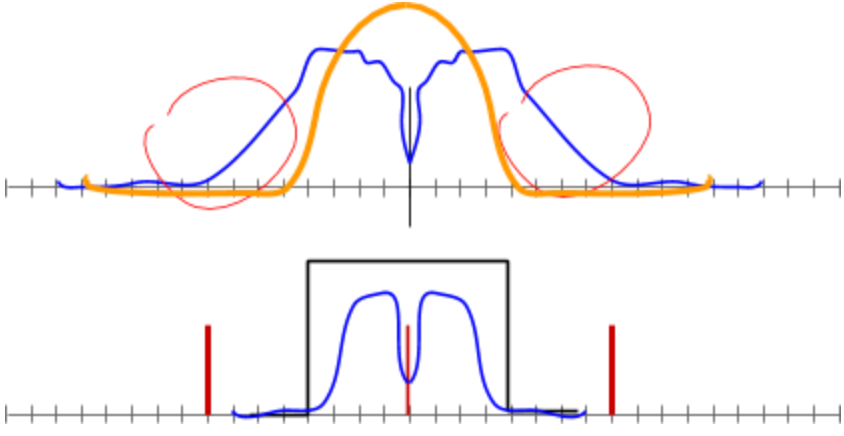
Smooth sampling/reconstruction: Signal theory

(Pre)filtering the data: left nothing out of the box !



Smooth sampling/reconstruction: Signal theory

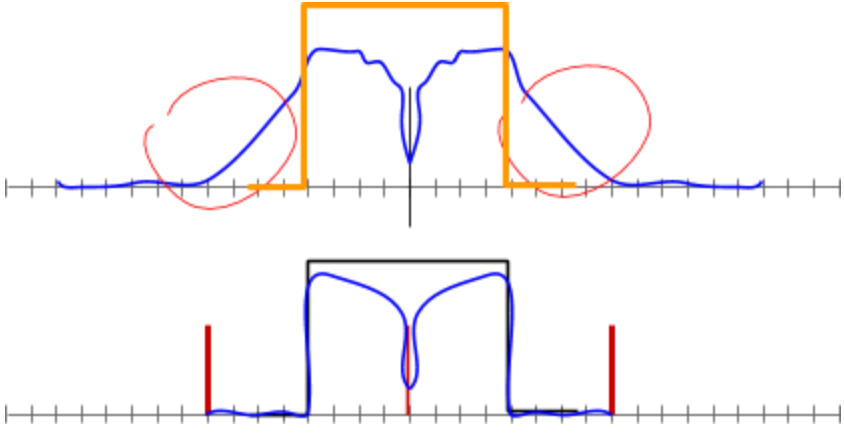
(Pre)filtering the data: Kernel that left nothing out of the box !



Smooth sampling/reconstruction: Signal theory

(Pre)filtering the data: Kernel that left nothing out of the box !

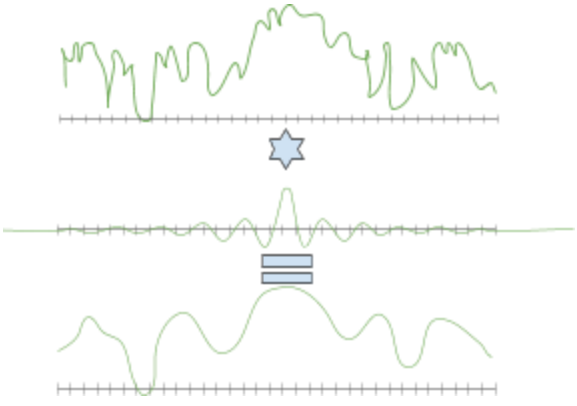
Then, why not using the box itself ?

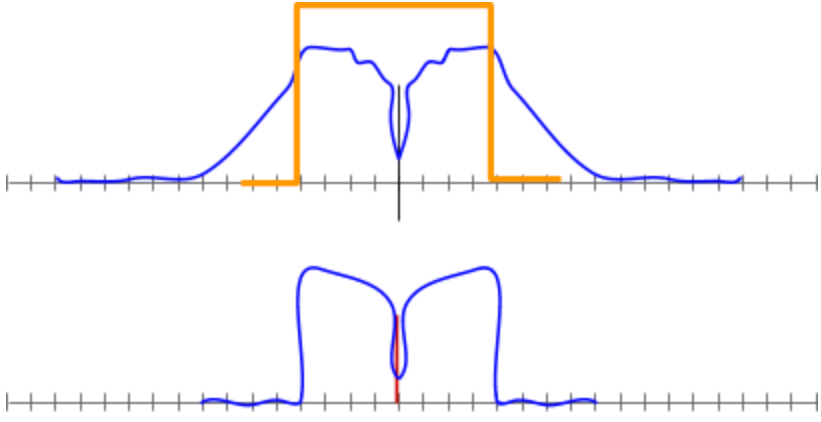


Smooth sampling/reconstruction: Signal theory

Then, why not using the box itself ?

= Sinc = $\frac{\sin(\pi x)}{\pi x}$. *The optimal filter according to signal theory*





Niceness of Sinc filter

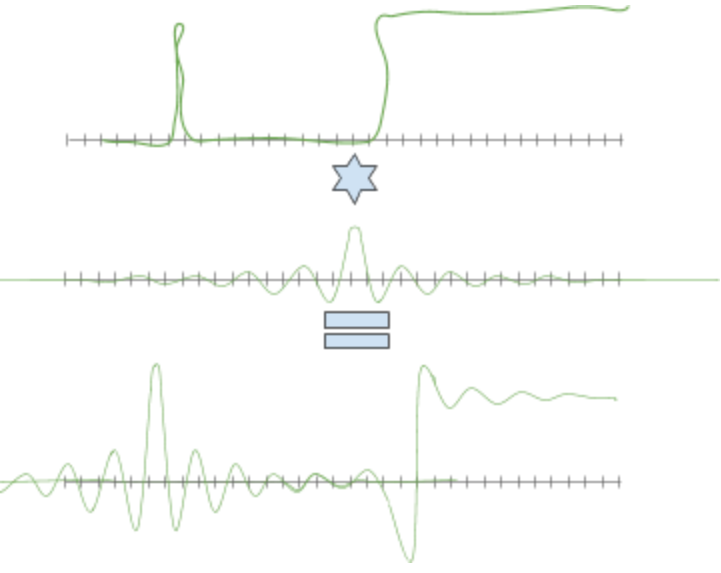
Sinc = $\frac{\sin(\pi x)}{\pi x}$.

- The optimal filter according to signal theory : keeps 100% of good, kills 100% of bad
- Interpolates data exactly (so $Sinc_1$ is neutral).

Problems of Sinc filter

Then, why not using the box itself ?

= Sinc = $\frac{\sin(\pi x)}{\pi x}$. *The optimal filter according to signal theory*



Problem 1:

- image is signal ≥ 0 (or even, in [0,1])
- On hard peaks and steps, Sinc can give negatives, overshoots, ripples/ringing :-(
- variance map (LEAN) : we can have $\sigma^2 < 0$!!!

Problem 2: (implem)

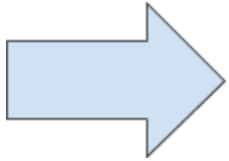
- filter infinitely large

Problems of Sinc filter

Then, why not using the box itself ?

= Sinc = $\frac{\sin(\pi x)}{\pi x}$. *The optimal filter according to signal theory*

Problem 3: Do you really like this optimality ? :-)

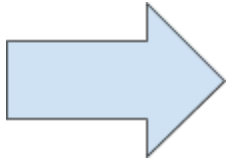


Sinc = razor: is perfectly… brutal !

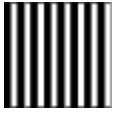
Brutal change = clandestine perceptual feature : people see a stop + a disk

Problem of smooth filter

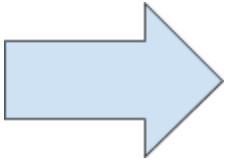
Turn grey too early : loose too much contrast

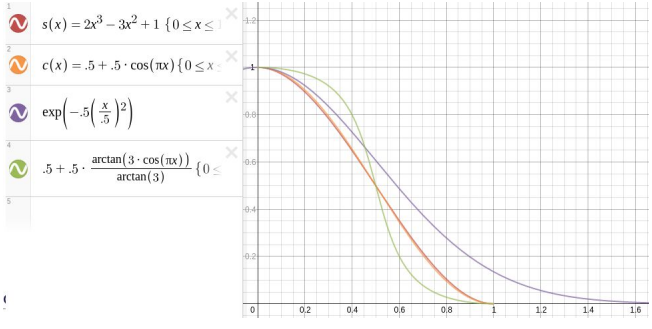


What's about a bit of aliasing ?



might be aliasing, but how lovely sharp ! :-) [Brown69]: a bit is ok.



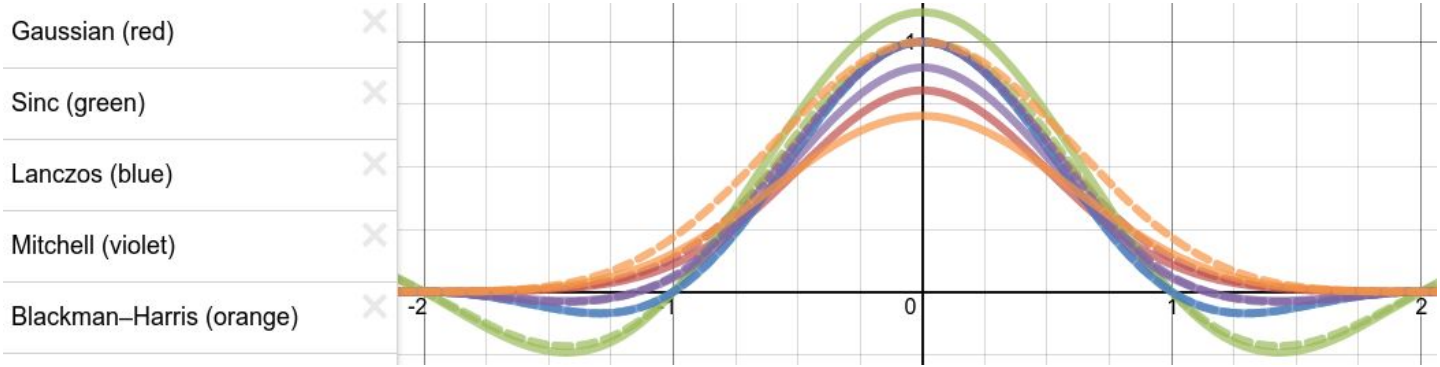


rows: Filter scaling = 200% (aliasing !), 160% , 120 %

Box iso (=Sinc) *Gaussian* *Spline iso* *Smoothbox(3) iso* *Smoothbox(3) separable* *Smoothbox(1.5) iso*

When visually evaluating the quality,
take care of color space !!! (gamma)
→ sRGB = space where 1+1=2 visually
(tone mapping kills physicality of intensity)

More filters

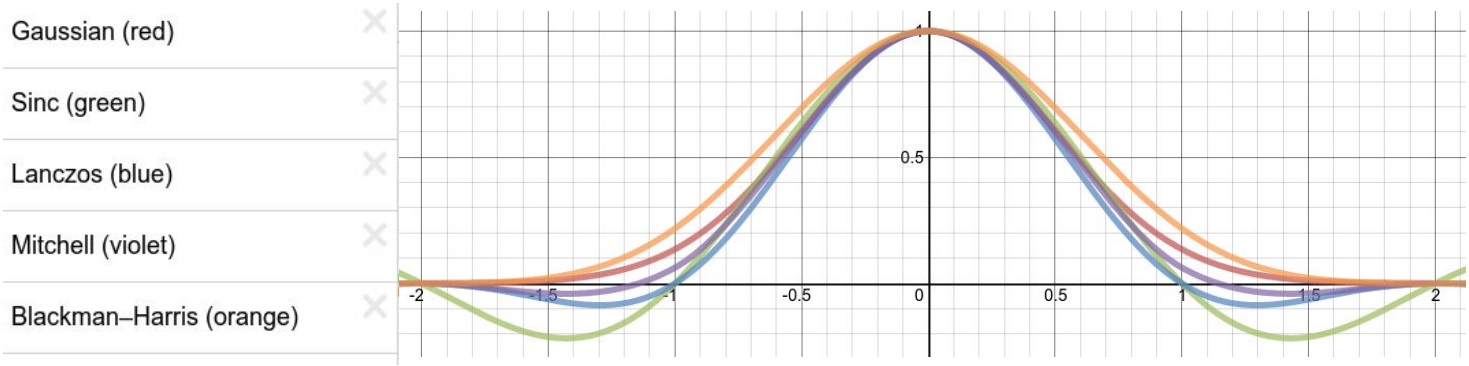


even more: wikipedia

- Lanczos: the taste of Sinc with less issues
- Some positives, some not
- Different fall-off
- Different filter size / evaluation cost (e.g. separable: Gauss)
- unique properties: - Sinc(correct signal), CR spline → unchanged (interpolant)
  - for Sinc & ~Gauss, $F_{½} * F_{1} = F_{½}$: cascade undistorted

More filters

(normalized as F(0)=1)



even more: wikipedia

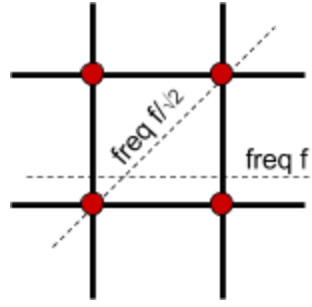
- Lanczos: the taste of Sinc with less issues
- Some positives, some not
- Different fall-off
- Different filter size / evaluation cost (e.g. separable: Gauss)
- unique properties: - Sinc(correct signal), CR spline → unchanged (interpolant)
  - for Sinc & ~Gauss, $F_{½} * F_{1} = F_{½}$: cascade undistorted

BTW: in 2D, isotropic or anisotropic ?

Pixel grid (i.e., sampling) is anisotropic → optimal filter should be squarish as well.

Remember the lesson: **signal theory optimal is not always our problem's optimal**.

isotropic anisotropic



. & aniso → separable

. → cheaper !

Filter squarish → result diamondish

Here, ***anisotropy (angular change) = clandestine perceptual feature***

BTW: in 2D, isotropic or anisotropic ?

Remember the lesson: signal theory optimal **is not always or problem**'s optimal.
Here, *anisotropy (angular change) = clandestine perceptual feature*

It's not because we can store more information that we should do it.

***Keeping perceptual property** in position / angle / zoom / time **is WAY MORE important**.*
/ Not creating

BTW: in 2D, isotropic or anisotropic ?

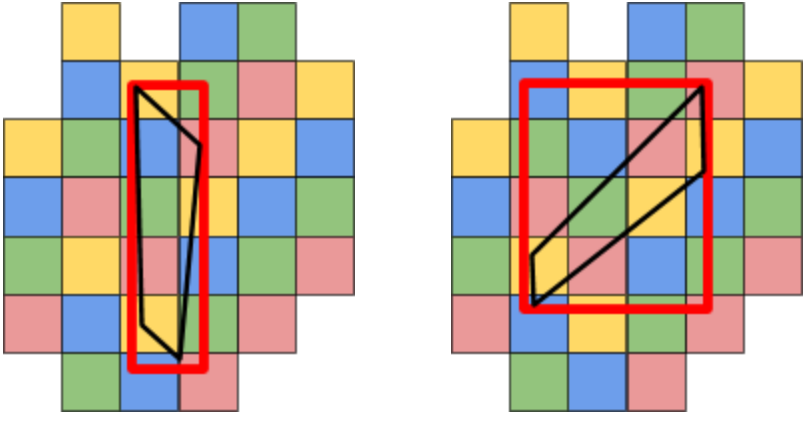
Remember the lesson: signal theory optimal is not always or problem's optimal.
Here, *anisotropy (angular change) = clandestine perceptual feature*

It's not because we can store more information that we should do it.

***Keeping perceptual property** in position / angle / zoom / time **is WAY MORE important**.*

Remember the promising "*Summed Area Table*" [Crow84]

- Always better that MIPmapping under any condition.
- But contrast varies with angle on a turntable. → crippling (perceive a pulse).



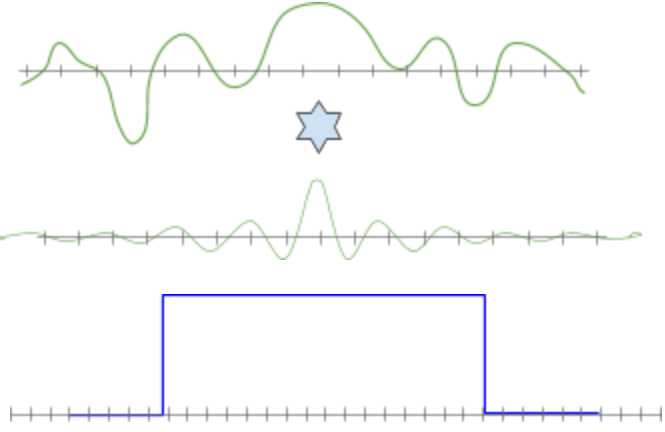
Criteria

Effect of CR spline parameters [Mitchell'88]

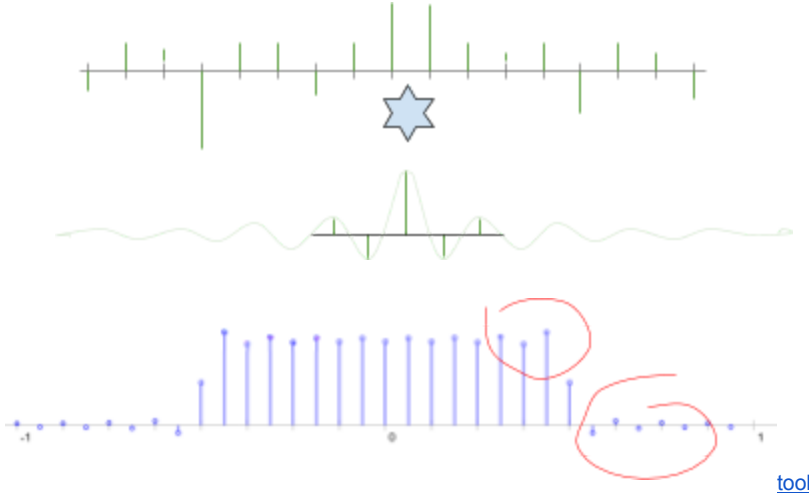
Story not finished: We use **finite** **discrete** *filters !*

oiio: 8x8

not this :



but this :



tool

then cascading (for MIP-map) [wrong: spectrum continuous !]

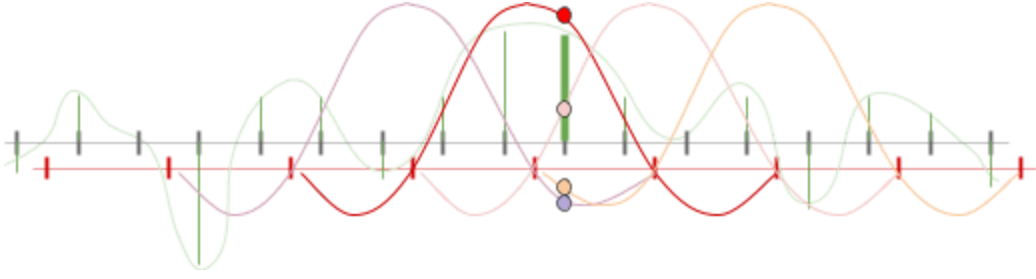
→ get then more and more distorted

ends up as box filters ! (→ ***forget cascading***)

Story not finished: We use finite discrete *filters !*

In 1D :

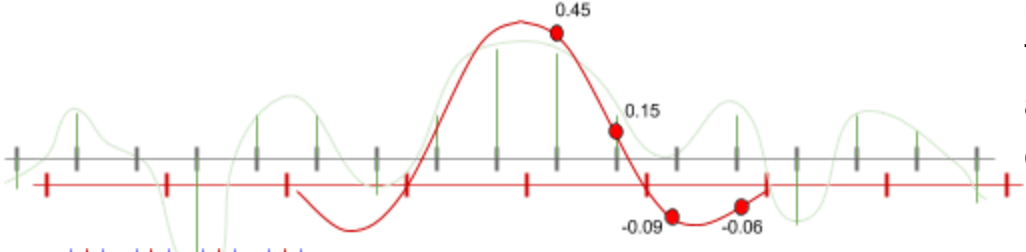
oiio: 8x8 : **staggered**



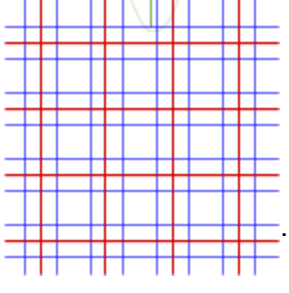
staggered discrete Sinc:
negatives weight +12% than continuous Sinc , and +100% than centered Sinc !

.

.

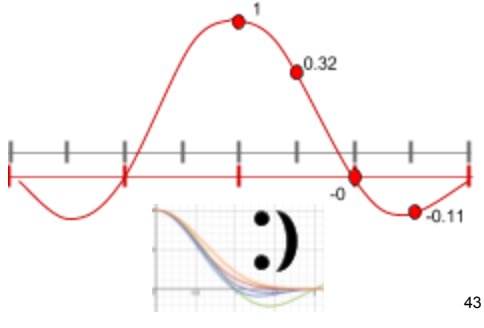


In 2D :



.

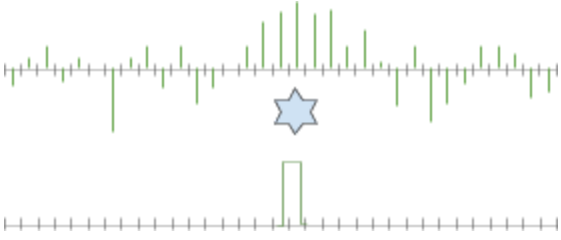
(impulse response)

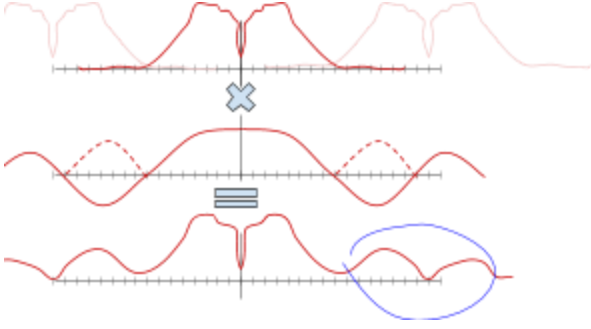


Story not finished: *We use* finite discrete *filters !*

Reconstruction: (pixel interpolation / Mag filter) can create post-aliasing

Nearest (= box)

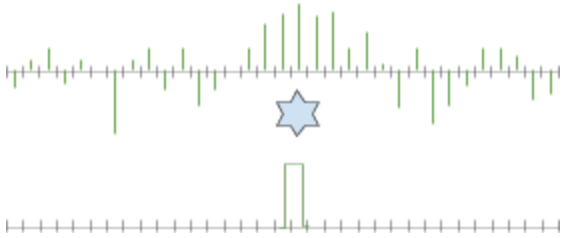


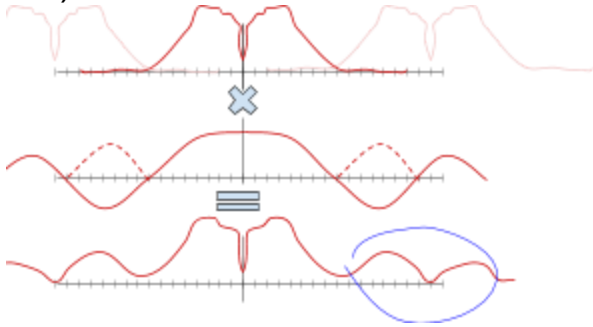


Story not finished: *We use* finite discrete *filters !*

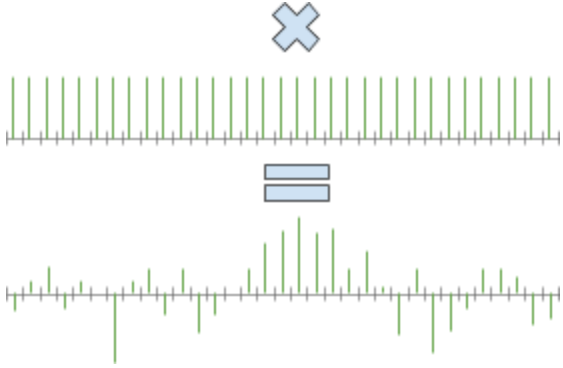
Reconstruction: (pixel interpolation / Mag filter) can create post-aliasing

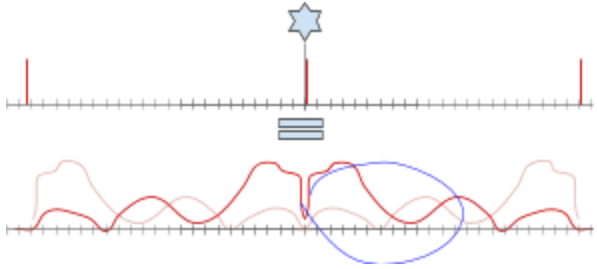
Nearest (= box)





Final resample

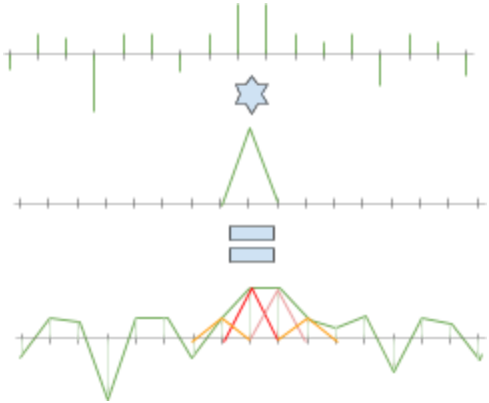


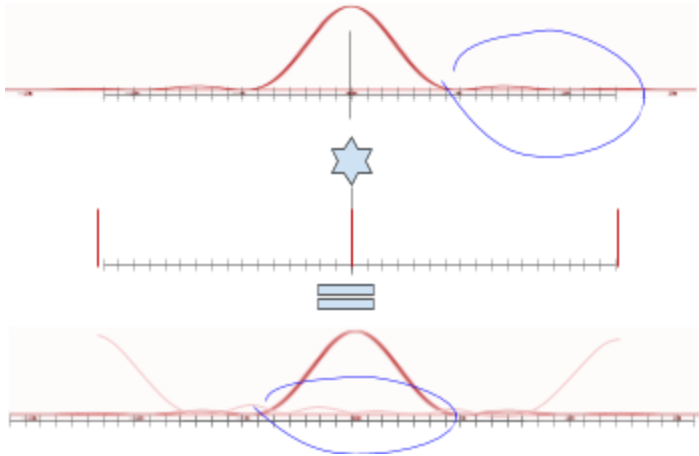


Story not finished: *We use* finite discrete *filters !*

Reconstruction: (pixel interpolation / Mag filter) can create post-aliasing

Linear interpolation (= tent)

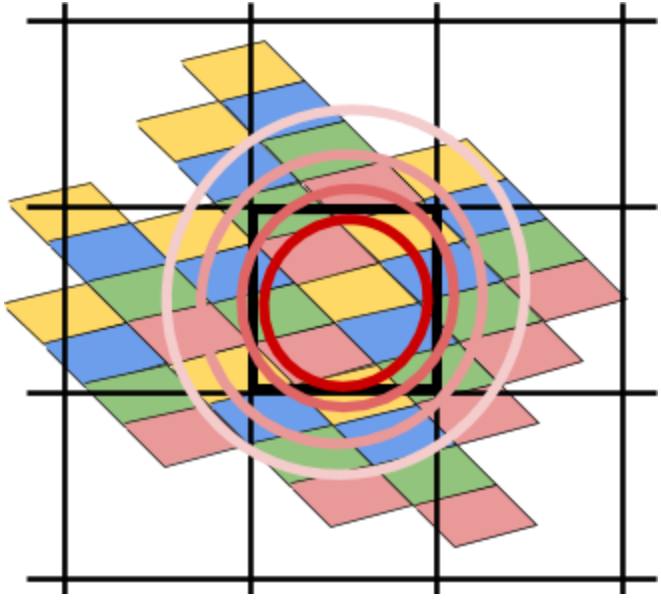


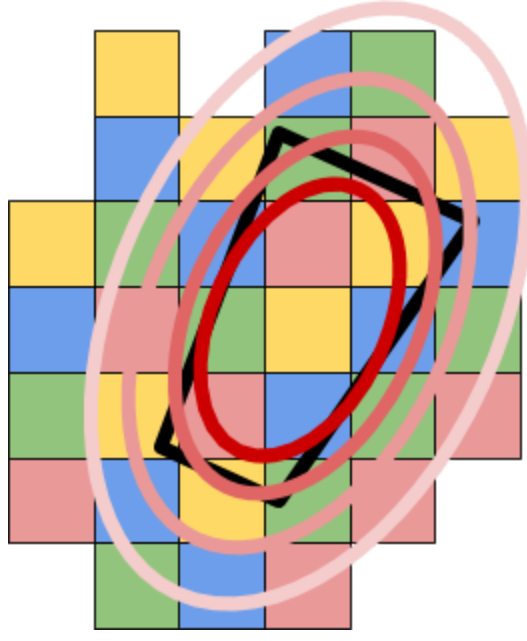


. still not perfect, with a bit of post-aliasing

. → CR Spline better (or Sinc :-))

Filter “size”: what’s about our pixel footprint ?





View-dep: cannot naively precalculate.

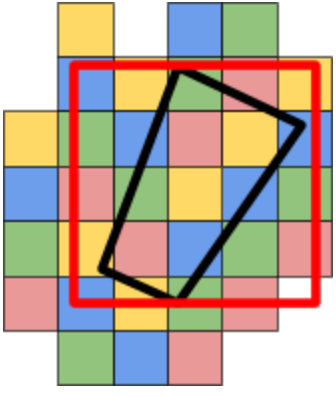
Can we avoid one full filter computation per pixel (/ dice vertex) ?

Reconstruction using MIP-map

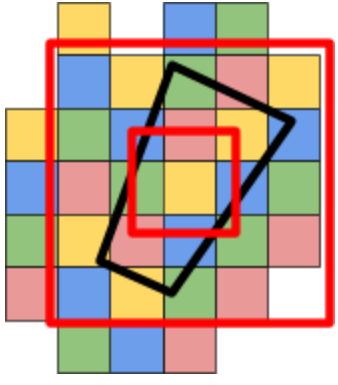
Can we avoid one full filter computation per pixel ?

MIP-map precomputation principle:

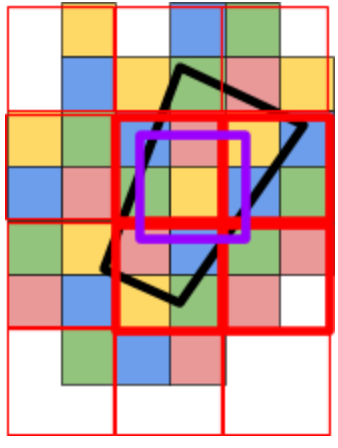
idea 1: B square



idea 2: B levels



idea 3: quadtree

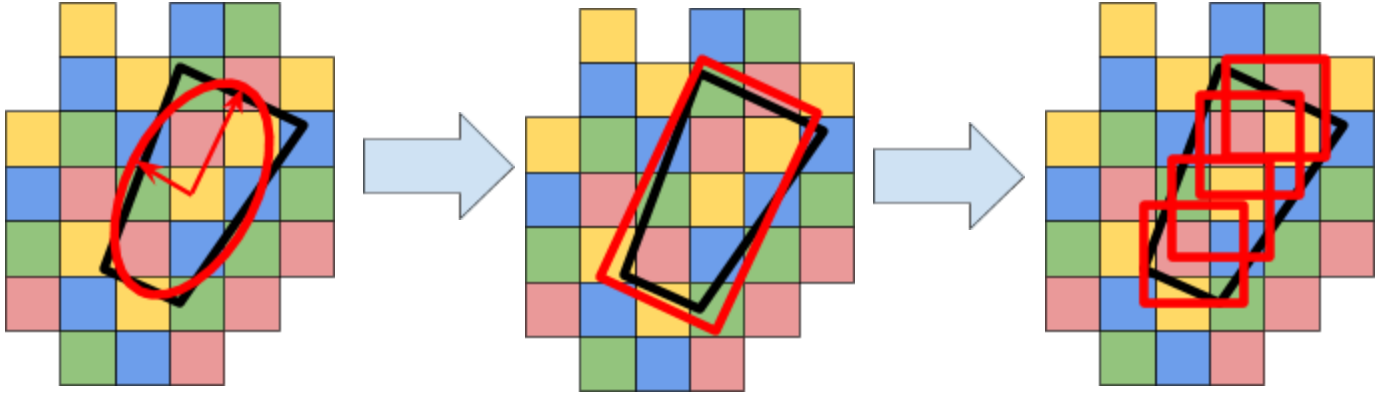


Kind of ugly… not even anisotropic !

(disclaimer: accumulation of filters → ~ Gauss :-))

Reconstruction using MIP-map

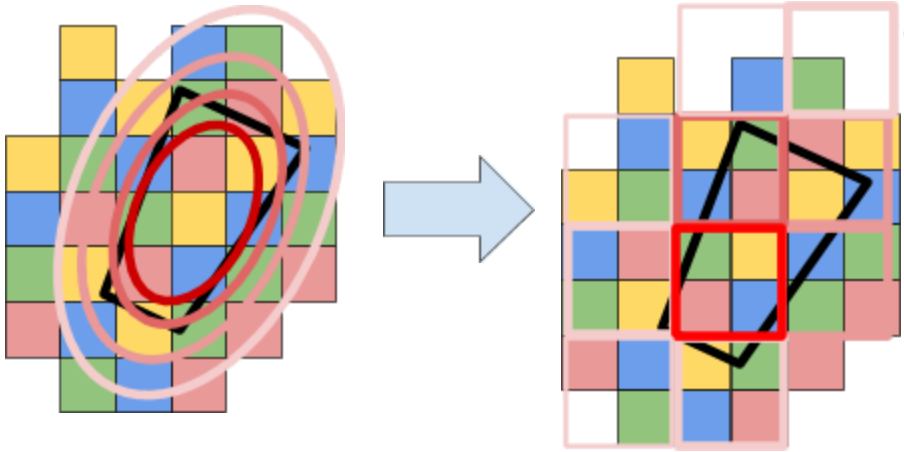
Anisotropic approx using MIP-map: (GPU aniso x4)



Reconstruction using MIP-map

Anisotropic approx using MIP-map: coarse filter * MIPmap = $F_1(F_2(\text{data}))$ or = $F_1(\text{LF data})$

→ we need the LF to not be stupid
(aliased, too LF, ...)

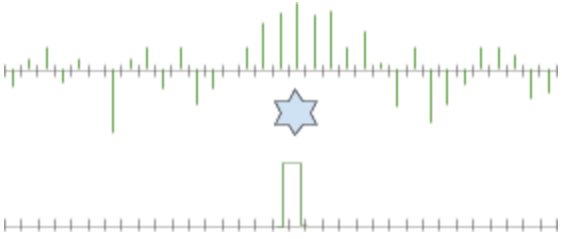


BTW:

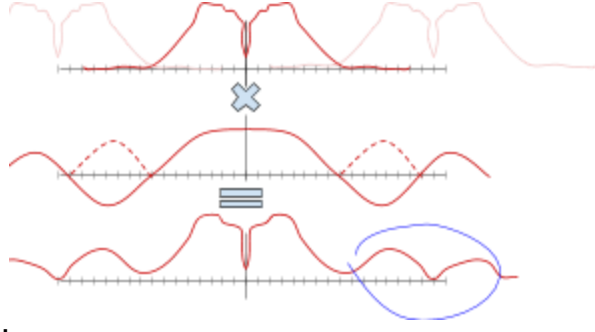
Might ellipse integration used in or upstream oiio ?

Cascaded filtering-subsampling (MIPmap)

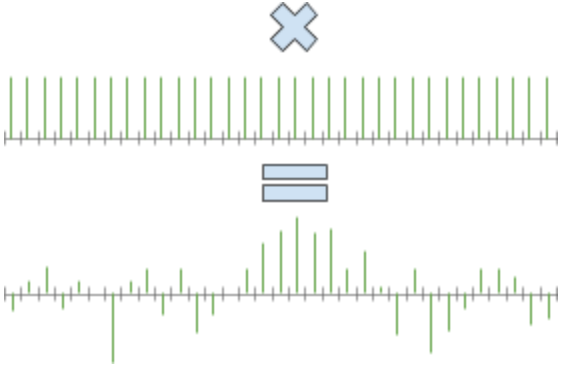
Level 0

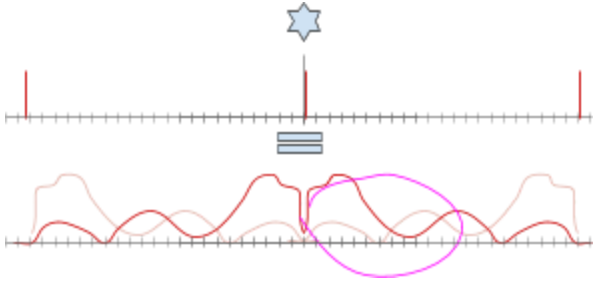


Nearest (= box)



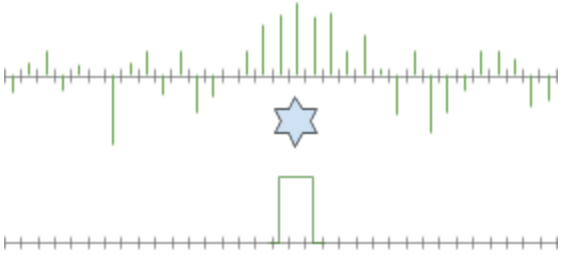
resample



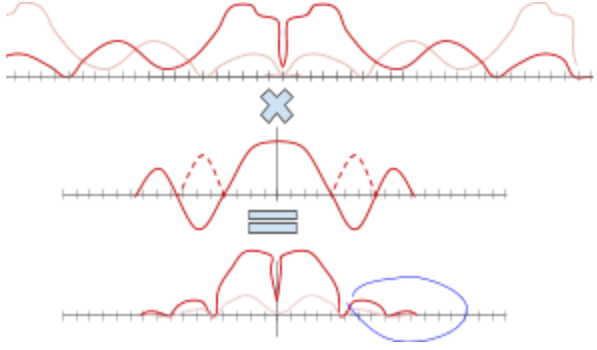


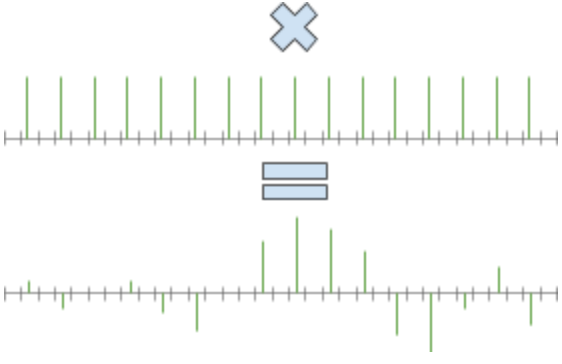
Cascaded filtering-subsampling (MIPmap)

Level 1

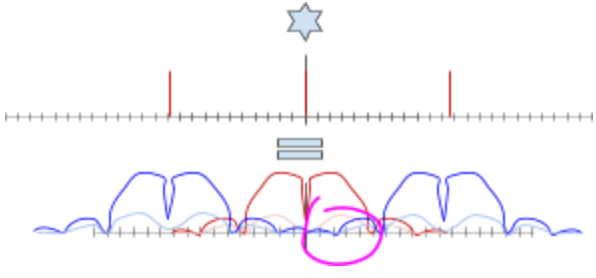


Nearest (= box)



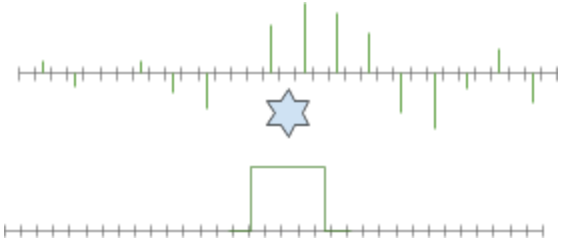


resample

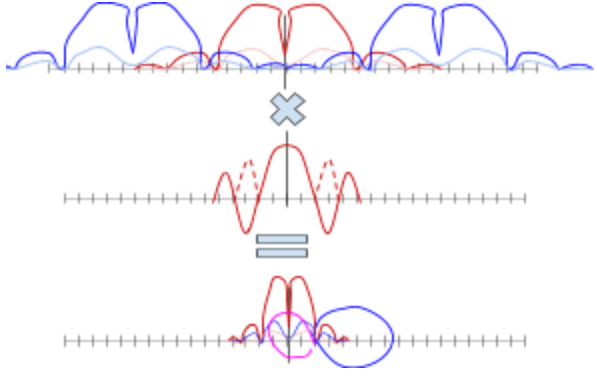


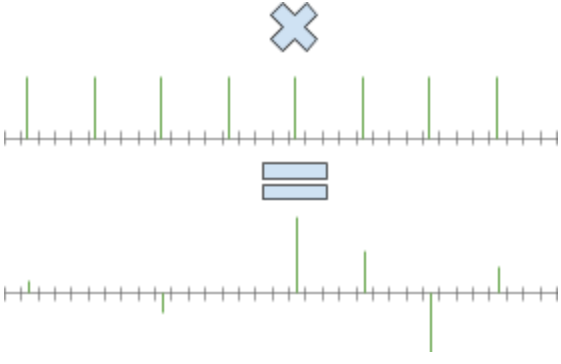
Cascaded filtering-subsampling (MIPmap)

Level 2

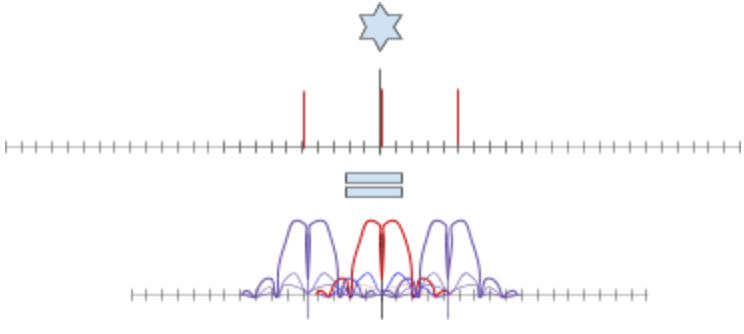


Nearest (= box)



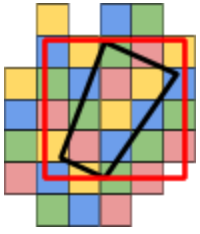


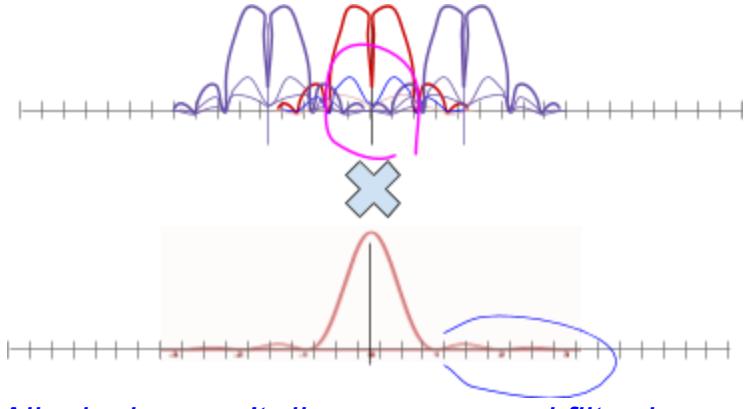
resample



Cascaded filtering-subsampling (MIPmap)

Fetch texel with interpolation :

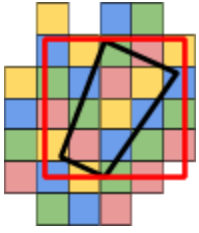


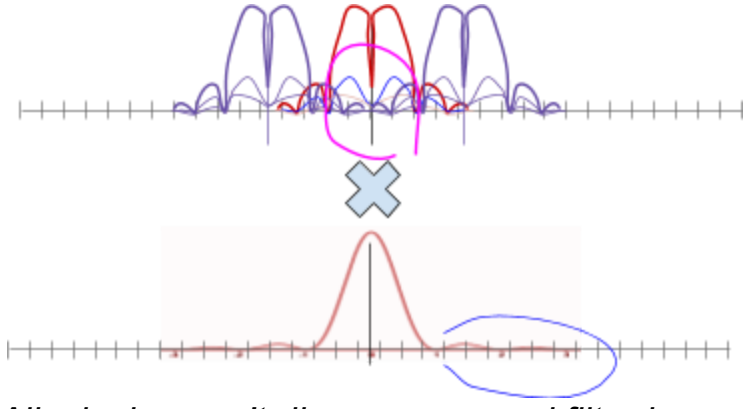


Aliasiasing can't disappear : good filter is crucial !

Cascaded filtering-subsampling (MIPmap)

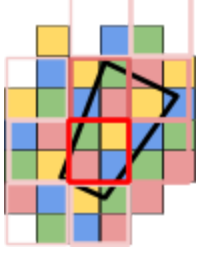
Fetch texel with interpolation :

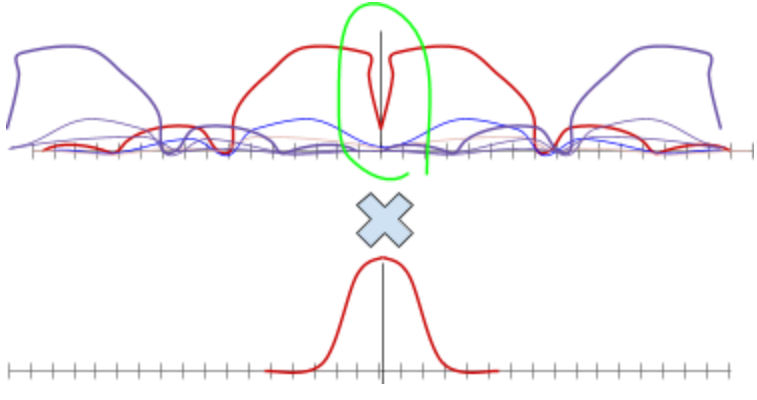




Aliasiasing can't disappear : good filter is crucial !

Integrate from thinner level :





Less garbage in LF

Filtering the pixel footprint: what should be “the right filter” ?

Sorry, no all-cooked recipe, many criterions and steps…

But plenty of ingredient for yours ! :-)

At least, a long list of pitfalls and traps to avoid ;-)

→ **Recap**

Filtering the pixel footprint: what should be "the right filter" ?

Recap : Cascade of Filters

- Filter at texture creation (e.g. Mari painter) - don't create junk, at first !
- Filter at MIP-map pyramid construction - or from base level. Filter dep use (σ ...)
- Filter at dice footprint integration (shading) - at least, fetch & tri-interpolate
- Filter at pixel reconstruction (PRman, Manuka)

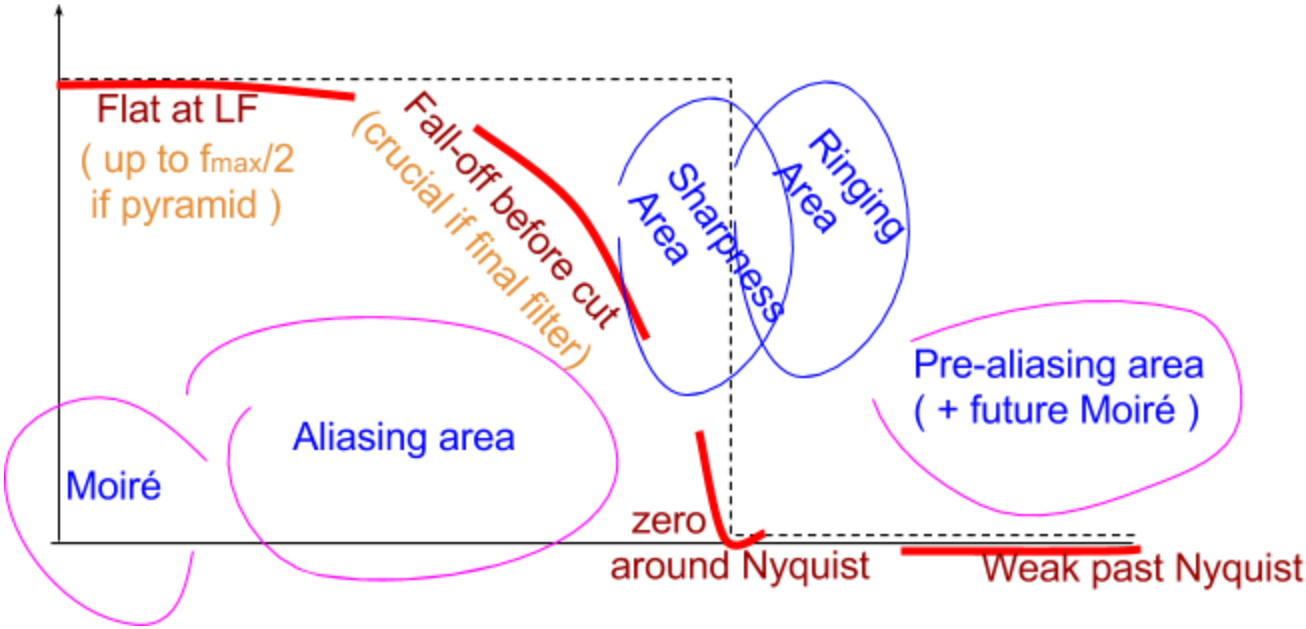
BTW:

Shading: dice footprint (+smooth deriv), not pixel ?

What is Manuka pixel Kernel for path tracing ?

Filtering the pixel footprint: what should be "the right filter" ?

Recap : Filter shape (Fourier space)



Filtering the pixel footprint: what should be “the right filter” ?

Sorry, no all-cooked recipe, many criterions and steps…

But plenty of ingredient for yours ! :-)

At least, a long list of pitfalls and traps to avoid ;-)