Sampling

Konstantin Tretjakov (kt@ut.ee)

21. november 2005

Täna räägime

- Diskretiseerimine ja taastamine *(sampling & reconstruction)*
- Sakilisus *(aliasing)*
- Fourier' analüüs, sagedusesitus
- Korrektne diskretiseerimine *(anti-aliasing)*
- Korrektne taastamine *(reconstruction filters)*

Küsimused

- Mis on pilt?
- Mis on piksel?

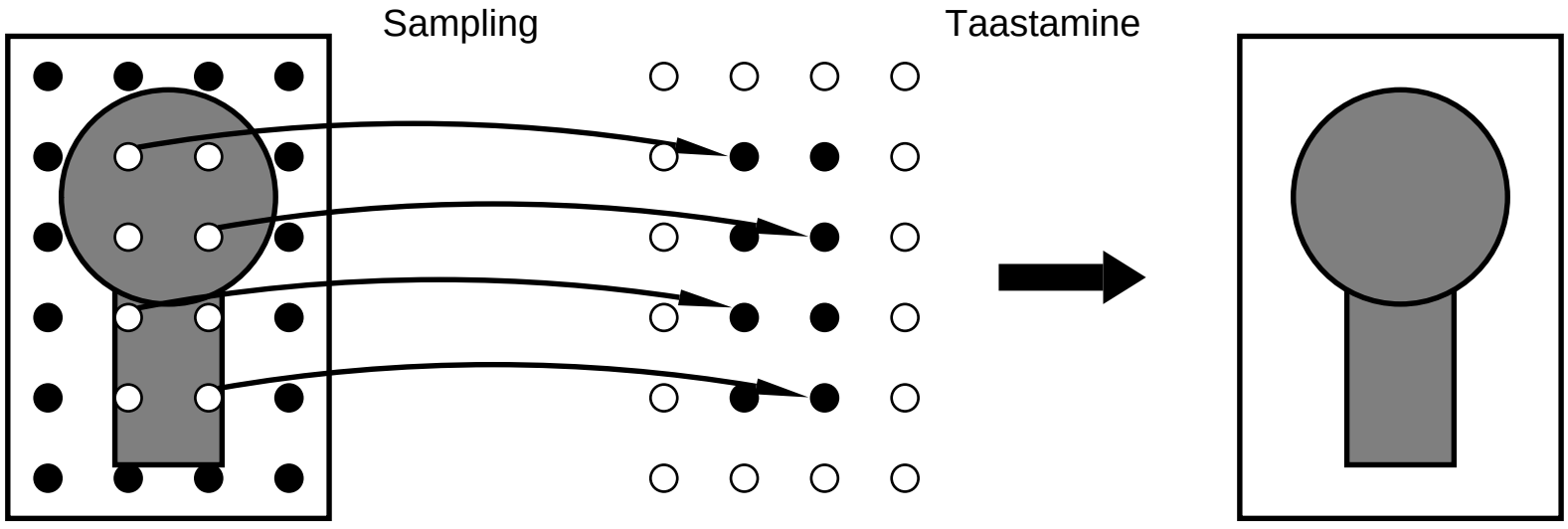
Vastused

- Pilt on *kahe muutuja funktsioon* $p(x, y)$.
- Üldiselt on pilt *pidevate* muutujate funktsioon $(x, y \in \mathbb{R})$.
- Analüütiliselt on pilte esitada võimatu, selle asemel salvestame me *valimi* pildi väärtustest teatud punktides $\{p(x_1, y_1), p(x_2, y_2), \ldots, p(x_n, y_n)\}$.
- Iga valimi punkt (x_i, y_i) koos vastava väärtusega $p(x_i, y_i)$ on *piksel*. Ehk: piksel on *pildi punkt*.
- Piksel ei ole midagi nähtavat.

Diskretiseerimine ja taastamine

- Pildi salvestamiseks konverteerime teda pideva muutuja funktsioonist diskreetse muutuja funktsiooniks ning salvestame teda väärtused tabelina.
- Me peame olema võimelised pärast sellest tabelist algset pildi taastada.
- Esimest protsessi nimetatakse *diskretiseerimine (sampling)*. Teist – *taastamine (reconstruction)*.
- Põhiküsimus: kuidas neid kõige õigemini teostada?

Diskretiseerimine ja taastamine



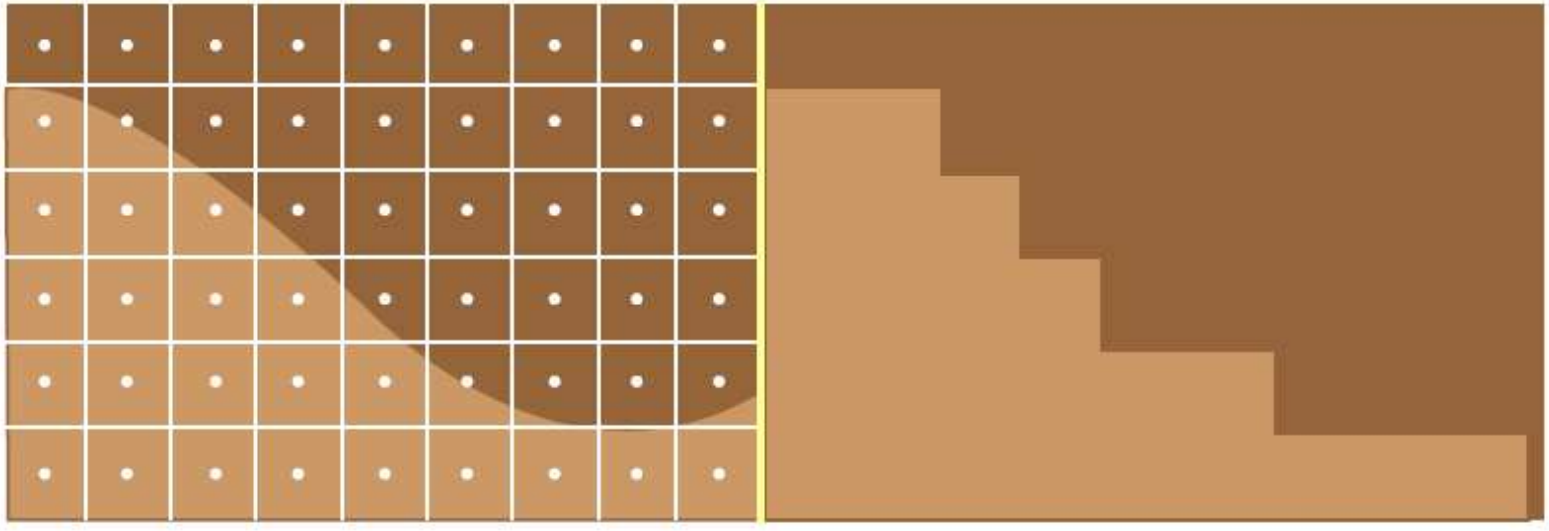
Diskretiseerimine ja taastamine

- Diskretiseerimise näited
  - Pildi pikslid (mitte päris ekraanipikslid!)
  - Kiired raytracing-algoritmis
  - Z-puhvri väärtused
  - Tekstuuri tekslid
  - Filmi kaadrid (ajaline diskretiseerimine)
- Taastamine
  - Pildi väljastamine ekraanile (CRT, LCD)
  - Video näitamine kaadrite jadana
  - Tekstuurimine, resampling

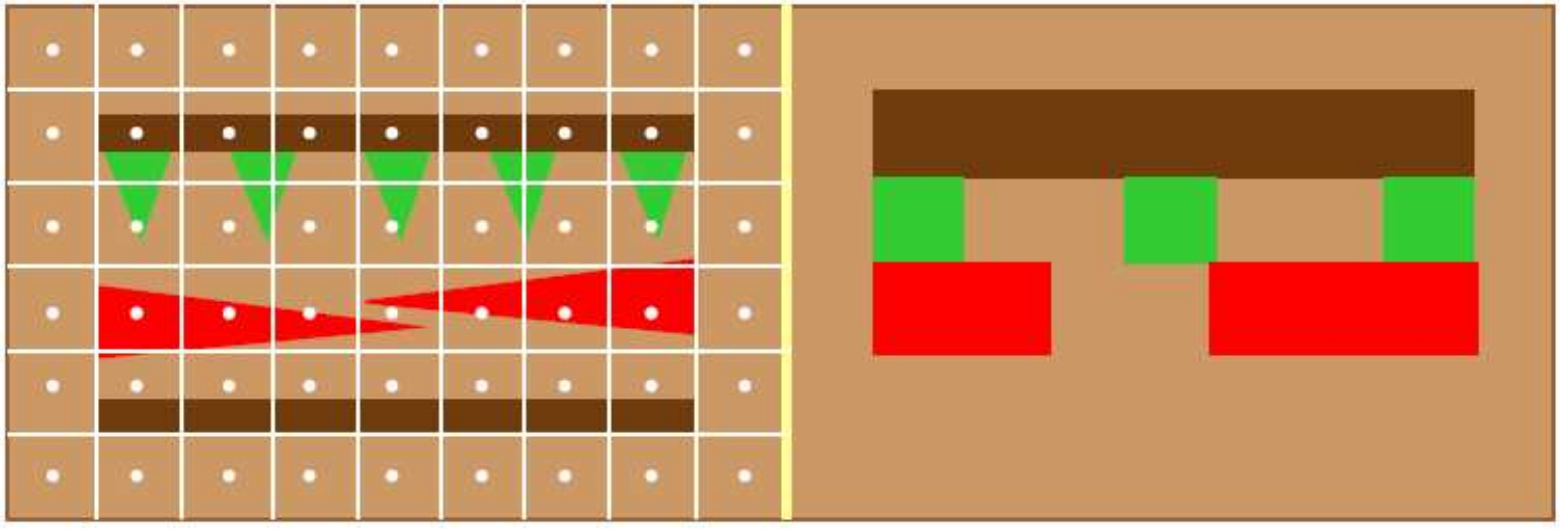
Diskretiseerimine ja taastamine

- Ideaalis soovime et pärast diskretiseerimist saaksime täpselt taastada originaalpildi.
- Kui seda ei saa, siis proovime vähemalt mitte tekitada asju mida algsel pildid polnud.
- Vale diskretiseerimise tõttu tekivaid vigu nimetatakse *sakilisuseks (aliasing)*:
  - Näited: joonte sakilised ääred, valesti renderdatud pisidetailid, moiré-efektid
- Vale taastamine tekitab natuke vähem olulisi vigu:
  - „Nähtavad pikslid", vilkuvad kaadrid filmis, jne.

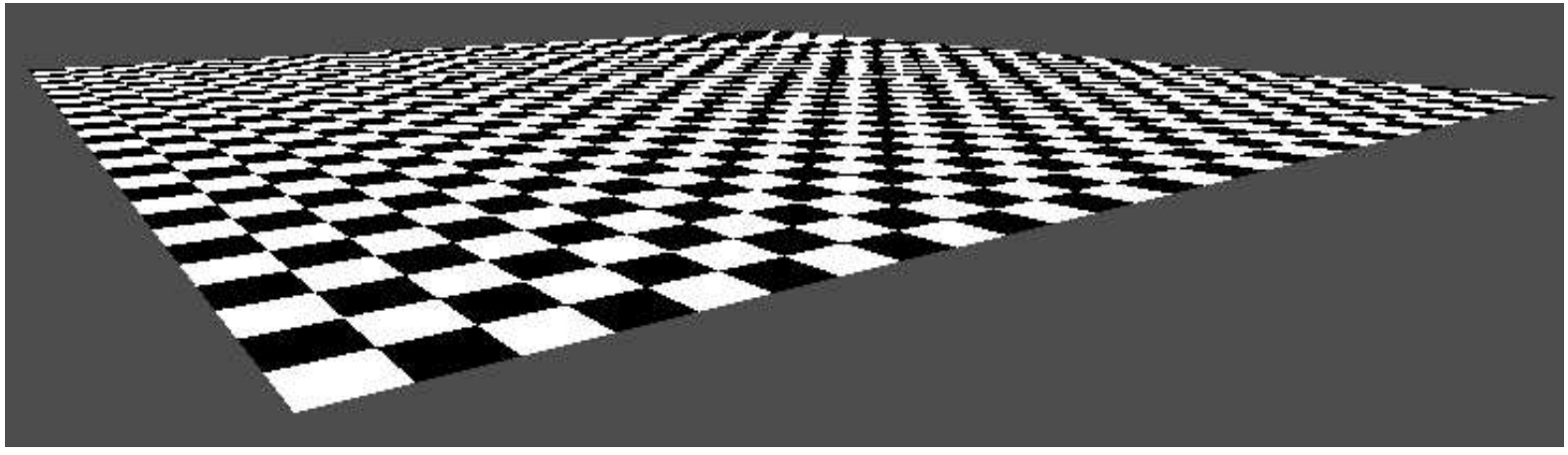
Aliasing: Sakilised jooned



Aliasing: Valed pisidetailid



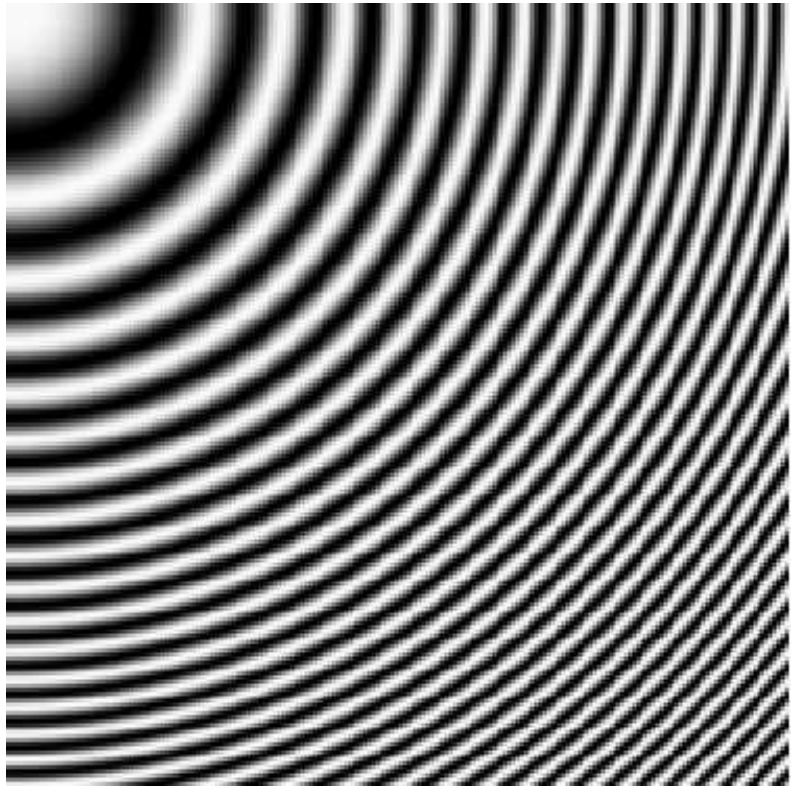
Aliasing: Vale tekstuurimine



Aliasing: Moiré efektid

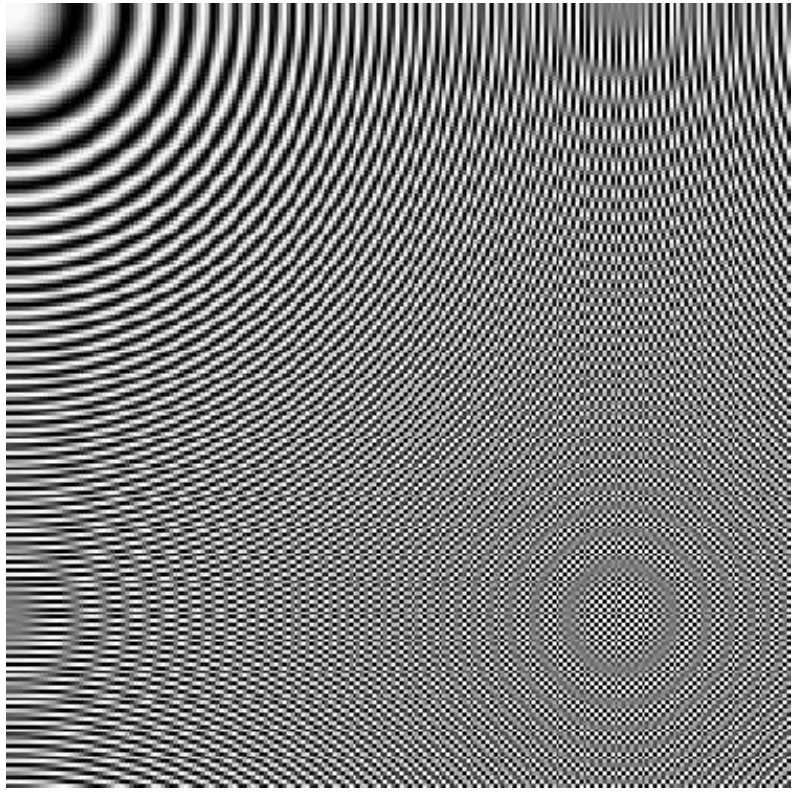
- Olgu meie „pilt” kirjeldatav analüütiliselt funktsioonina $p(x, y) = \cos(x^2 + y^2)$.
- Diskretiseerime seda 200×200 pikslite massiiviks erinevate resolutsioonidega:
  - Võtame sampleid sammuga 0.05 ristkülikus $[0, 10] \times [0, 10]$.
  - Samplime ka kaks korda suurema sammuga kaks korda suuremas ristkülikus.
  - ... ning veel kaks korda suurema sammuga (0.2) ristkülikus $[0, 40] \times [0, 40]$.

Aliasing: Moiré efektid



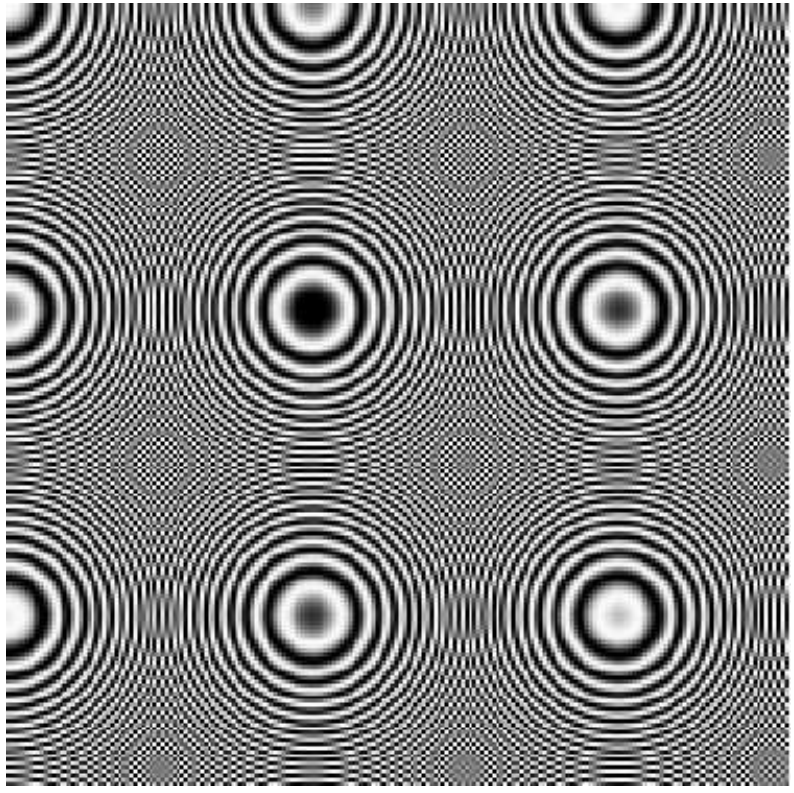
$$p_{ij} = p(0.05i, 0.05j)$$

Aliasing: Moiré efektid



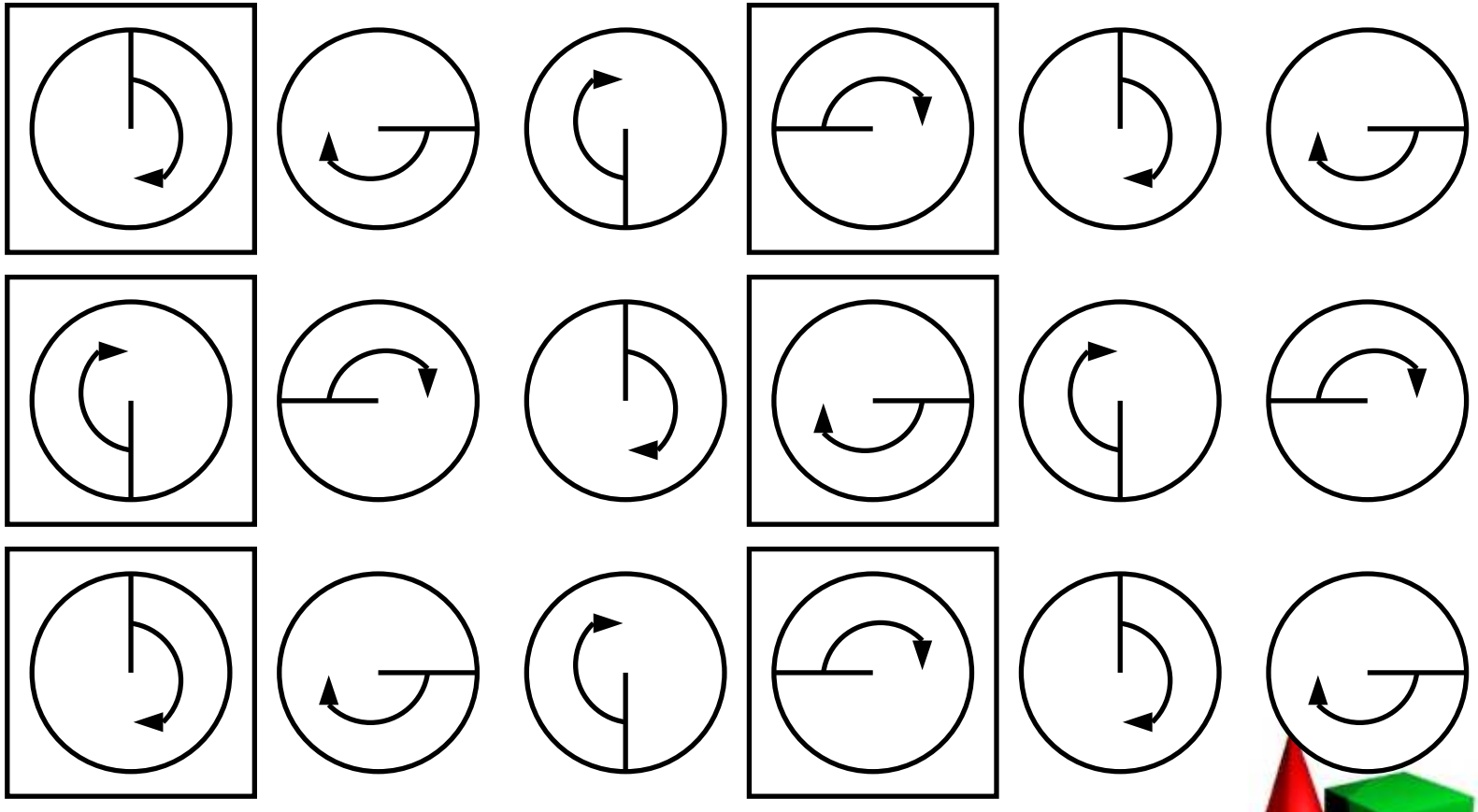
$$p_{ij} = p(0.1i, 0.1j)$$

Aliasing: Moiré efektid



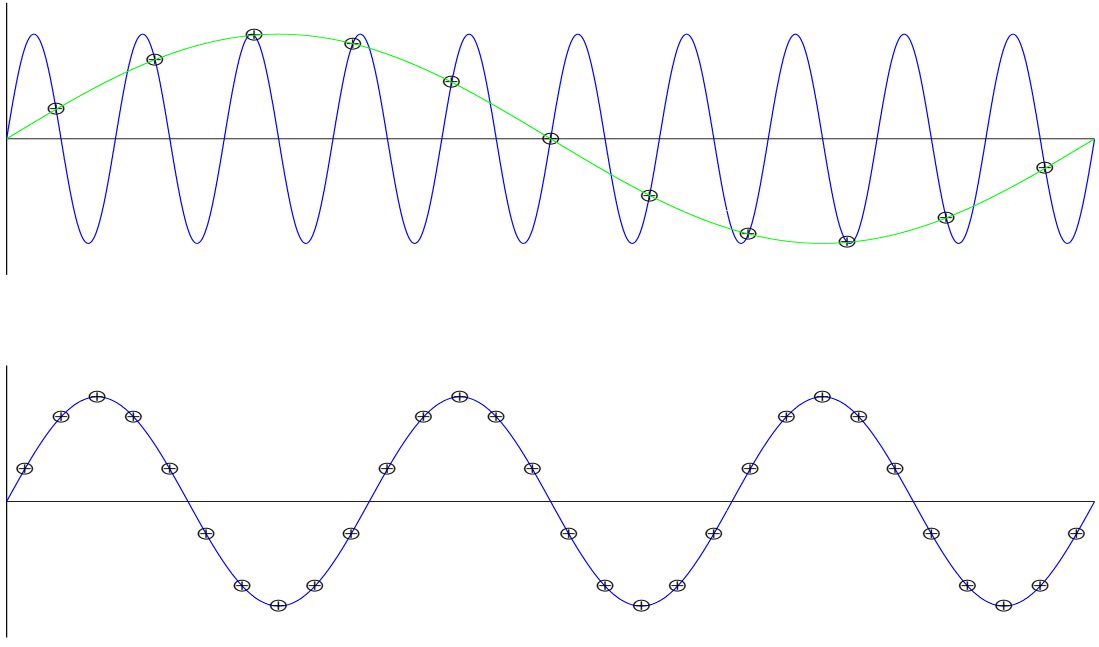
$$p_{ij} = p(0.2i, 0.2j)$$

Temporal aliasing



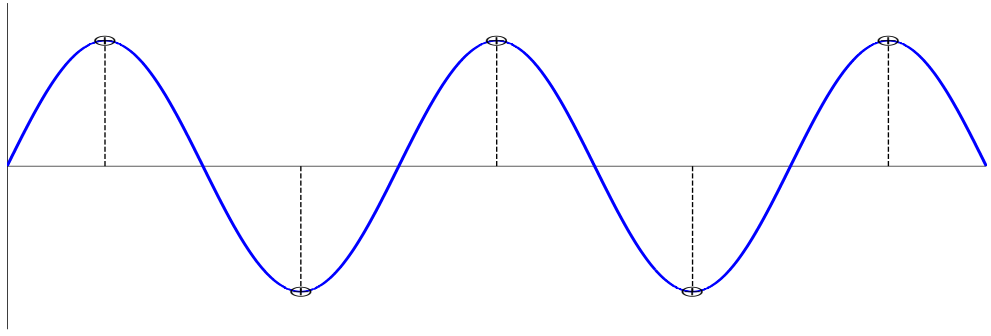
Milles siis viga?

- Kiiresti muutuva signaali ei tohi diskretiseerida liiga hõredalt!



Nyquist'i teoreem

- Osutub et *selleks et korrektselt diskretiseerida signaali, peab diskretiseerimise sagedus rohkem kui kaks korda kõrgem signaali sagedusest*



$$\omega_{\text{sampling}} > 2\max(\omega_{\text{signal}})$$

Korrektne diskretiseerimine

- Valime diskretiseerimise sageduse rohkem kui kaks korda kõrgem pildi kõrgeimast sagedusest.
- Mõnikord see ei ole võimalik sest
  - Pilt on liiga kõrgete sagedustega ja meil ei ole piisavalt resurssi selleks et seda nii tihedalt diskretiseerida.
  - Pildi sageduse spekter võib üldse olla lõpmatu.
  - Sagedusspektri teadasaamine on liigne ajakulu.
- Siis peame me eelnevalt töötlema pildi ning eemaldama sellest need kõrged sagedused mida me korrektselt samplida niivõinaa ei saa.

Sagedusesitus

- Asjade formaalse käsitluse jaoks peame sisse tuua Fourier' pööre ning sagedusesituse mõistet.

$$f(t) = \int_{-\infty}^{\infty} F(w) e^{i2\pi wt} df$$

$$F(w) = \int_{-\infty}^{\infty} f(t) e^{-i2\pi wt} df$$

Fourier' esitus

- Nagu teame, n-mõõtmelist vektorit $\mathbf{v}$ saame esitada valitud baasis $\mathcal{B} = \{\mathbf{b}_1, \mathbf{b}_2, \dots\}$:

$$\mathbf{v} = \sum_i v_i^{\mathcal{B}} \mathbf{b}_i$$

kus koordinaadid $v_i^{\mathcal{B}}$ on leitavad kui vektori $\mathbf{v}$ projektsioonid (ühik-)baasivektoritele:

$$v_i^{\mathcal{B}} = \langle \mathbf{v}, \mathbf{b}_i \rangle = \textstyle\sum_j v_j^{\mathcal{A}} b_j^{\mathcal{A}}$$

kus $\mathcal{A}$ on mõni ortonormaalne baas.

Fourier' esitus

Funktsioonidega on asi analoogne:

	Vektor $\mathbf{v} : \{1 \dots n\} \to \mathbb{R}$	Funktsioon $f : \mathbb{R} \to \mathbb{R}$
Baas $\mathcal{B}$:	$\{\mathbf{b}_1, \dots, \mathbf{b}_n\}$	$\{b_1, \dots\}$ (lõpmatu)
Esitus baasis $\mathcal{B}$:	$\mathbf{v} = \sum_i v_i^{\mathcal{B}} \mathbf{b}_i$	$f(t) = \sum_i F_i^{\mathcal{B}} b_i(t)$ $f(t) = \int F^{\mathcal{B}}(i) b(i,t) di$
Projektsioon:	$v_i^{\mathcal{B}} = \langle \mathbf{v}, \mathbf{b}_i \rangle$ $= \sum_i v_j b_{ij}$	$F_i^{\mathcal{B}} = \langle f, b_i \rangle$ $= \int f(t) b_i(t) dt$

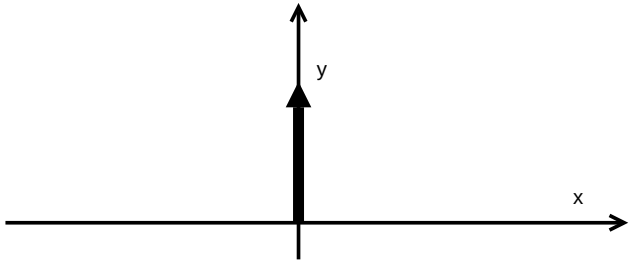
Dirac'i delta-funktsioon

- Veel üks oluline analoogia: vektori $\mathbf{v}$ i-s koordinaat v_i on projektsioon vektorile $(0, 0, \ldots, 1, 0, \ldots, 0)$.
- Funktsiooni puhul analoogselt: tema väärtus punktis t_0 on projektsioon funktsioonile $\delta(t - t_0)$:

$$f(t_0) = \int_{-\infty}^{\infty} f(t)\delta(t - t_0)dt$$

Dirac'i delta-funktsioon $\delta(t)$

$$\delta(t) = \begin{cases} \infty, \text{ kui } t = 0 \\ 0, \text{ otherwise} \end{cases} \qquad \text{ning} \int_a^b \delta(t)dt = 1 \text{ kui } 0 \in [a, b]$$



- Rangelt öeldes on δ mitte funktsioon vaid nn. *üldistatud funktsioon* või *mõõt*.

Dirac'i delta-funktsioon $\delta(t)$

- Delta-funktsioon vastab lõpmatult lühikesele ühikimpulssile, ning tema põhiomadus:

$$f(t_0) = \int_{-\infty}^{\infty} f(t)\delta(t - t_0)dt \quad \forall f$$

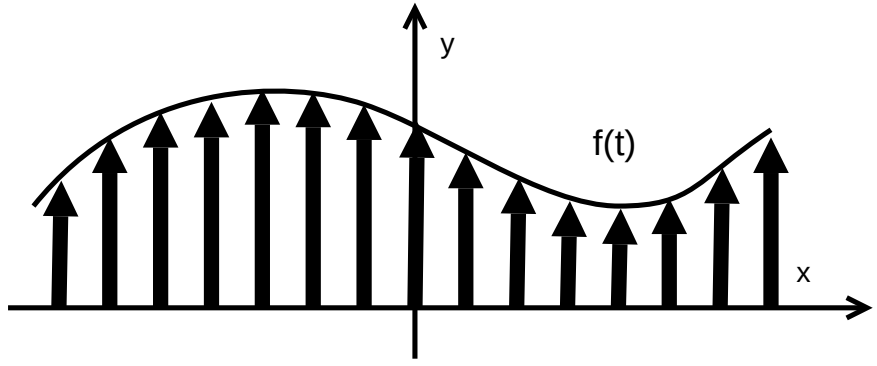
määrab teda teatud mõttes üheselt.

Dirac'i delta-funktsioon $\delta(t)$

- Võib öelda et iga funktsioon on iseenda baasi esitus δ-funktsioonide baasis:

$$\mathcal{B} = \{\delta(t - t_0) \,|\, t_0 \in \mathbb{R}\}$$

.



Fourier-esitus

- Funktsioone esitamine teistes baasides võib anda kasulikku infot.
- Üks populaarematest on *trigonomeetriline baas*:

$$\left\{ \frac{1}{2\pi},\ \frac{1}{\pi}\sin(kt),\ \frac{1}{\pi}\cos(kt) \,|\, k \in \{1, 2, \dots\} \right\}$$

- Selles baasis saab esitada kõik mõistlikud 2π-perioodilised funktsioonid.
- Funktsiooni esitust selles baasis nimetatakse *trigonomeetriliseks Fourier' reaks.*

Trigonomeetriline Fourier' rida

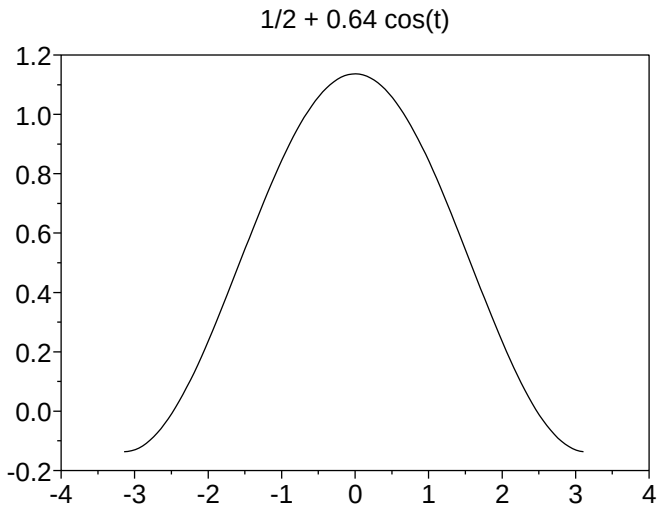
$$f(t) = \frac{a_0}{2} + \sum_{k=1}^{\infty}(a_k \cos(kt) + b_k \sin(kt))$$

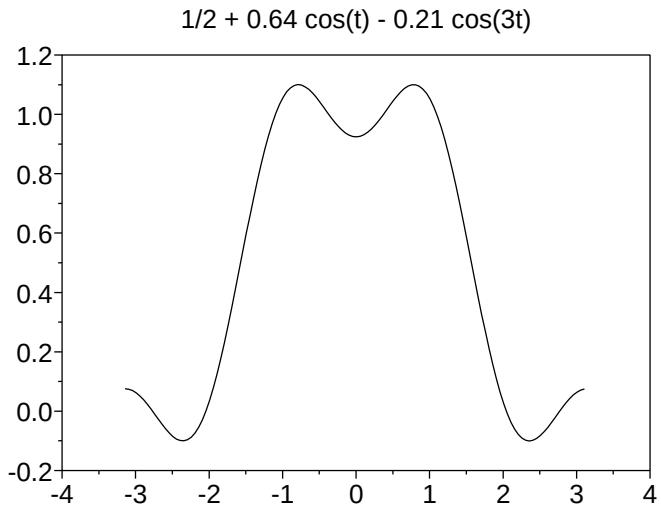
$$a_k = \frac{1}{\pi}\int_{-\pi}^{\pi} f(t)\cos(kt)dt$$

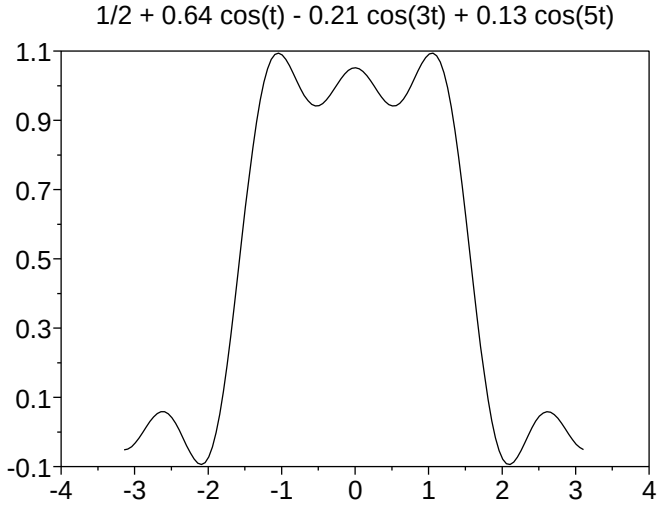
$$b_k = \frac{1}{\pi}\int_{-\pi}^{\pi} f(t)\sin(kt)dt$$

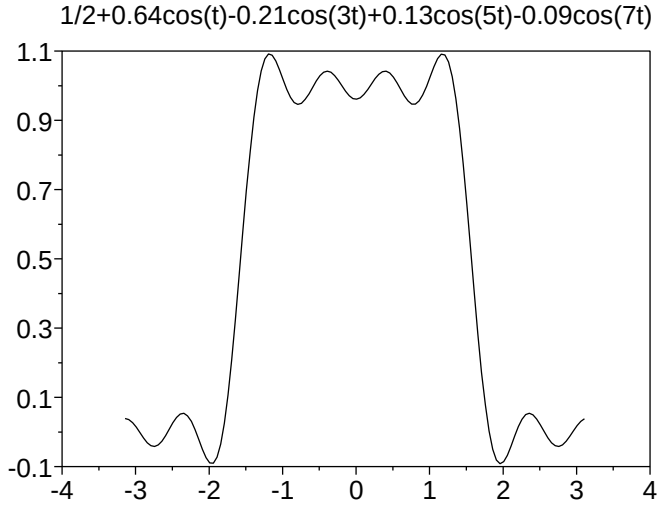
- Suuruste a_k ning b_k järgi saab näha kui palju peendetaile sisaldab funktsioon f.

Trigonomeetriline Fourier' rida









Fourier' rida komplekskuju

Kasutades valemeid:

$$\cos t = \frac{e^{it} + e^{-it}}{2} \qquad \sin t = \frac{e^{it} - e^{-it}}{2i}$$

saame viia trigonomeetrilist Fouier' rea *komplekskujule*:

$$f(t) = \sum_{k=-\infty}^{\infty} c_k e^{ikt}$$

$$c_k = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(t) e^{-ikt} dt$$

Fourier' rida perioodiga T

Periood ei pea olema 2π. Lihtsa skaleerimisega saame kätte Fourier' rea funktsioonile suvalise perioodiga T:

$$f(t) = \sum_{k=-\infty}^{\infty} c_k e^{ikt\frac{2\pi}{T}}$$

$$c_k = \frac{1}{T}\int_{-T/2}^{T/2} f(t)e^{-ikt\frac{2\pi}{T}}\,dt$$

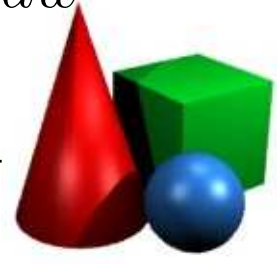
Fourier' integral

Mitteperioodiliste funktsioonide esitamiseks laseme $T \to \infty$, ning saame:

$$f(t) = \sum_{k=-\infty}^{\infty} c_k e^{ikt\frac{2\pi}{T}} =$$

$$= \sum_{k=-\infty}^{\infty} \frac{1}{T} \left(\int_{-T/2}^{T/2} f(t) e^{-ikt\frac{2\pi}{T}} dt \right) e^{ikt\frac{2\pi}{T}}$$

$$\to \int_{-\infty}^{\infty} \left(\int_{-\infty}^{\infty} f(t) e^{-i2\pi wt} dt \right) e^{i2\pi wt} dw$$



Fourier' pööre

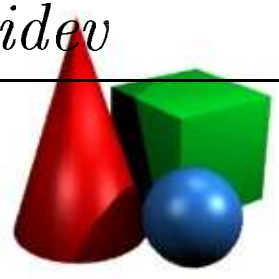
- Fourier' pööre *(Fourier' transform)*:

$$F(w) = \int_{-\infty}^{\infty} f(t) e^{-i2\pi wt} dt$$

- Tagurpidi pööre *(inverse Fourier' transform)*:

$$f(t) = \int_{-\infty}^{\infty} F(w) e^{i2\pi wt} dw$$

Mitteperioodiliste funktsioonide spekter on *pidev*



Fourier' pööre

- Peaaegu iga mõistliku funktsiooni puhul omab Fourier pööre mõtet (funktsioon peab rahuldama nn. *Dirichlet' tingimusi*).
- Funktsiooni „ajaline esitus" $f(t)$ ja *spektraalesitus* $F(w)$ on *samaväärsed*, s.t. sisaldavad sama palju informatsiooni (nad on ühe asja esitused erinevates baasides!).
- Kasutame tähistust $F = \mathcal{F}\{f\}$ ning ka

$$f \leftrightarrow F$$

Näide: Kastfunktsioon

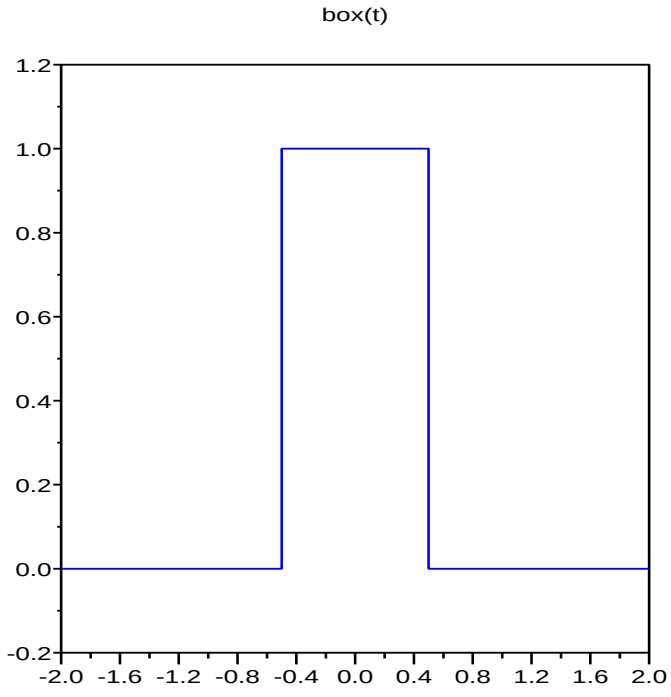
- Kastfunktsioon: $\Pi_a(t) = \begin{cases} 1, & \text{kui } t \in [-a, a] \\ 0, & \text{mujal} \end{cases}$

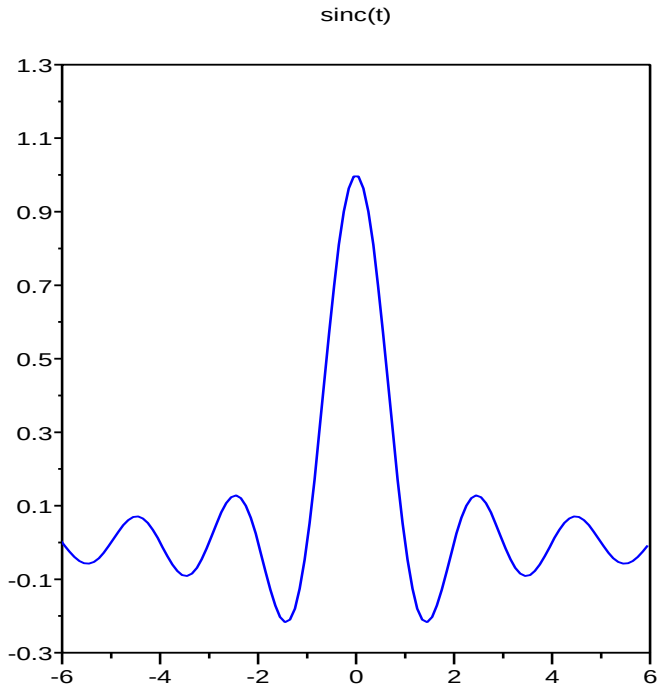
$$\mathcal{F}\{\Pi_a(t)\} = \int_{-a}^{a} e^{-i2\pi wt} dt == \left(\frac{e^{-i2\pi wt}}{-i2\pi w} \right) \Bigg|_{t=-a}^{a} =$$

$$= \frac{e^{-i2\pi aw} - e^{-i2\pi(-a)w}}{-i2\pi w} = \frac{\sin(2\pi aw)}{\pi w} =$$

$$= 2a \operatorname{sinc}(2aw)$$

Näide: Kastfunktsioon





$$\Pi_{1/2}(t) \leftrightarrow \mathrm{sinc}(w)$$

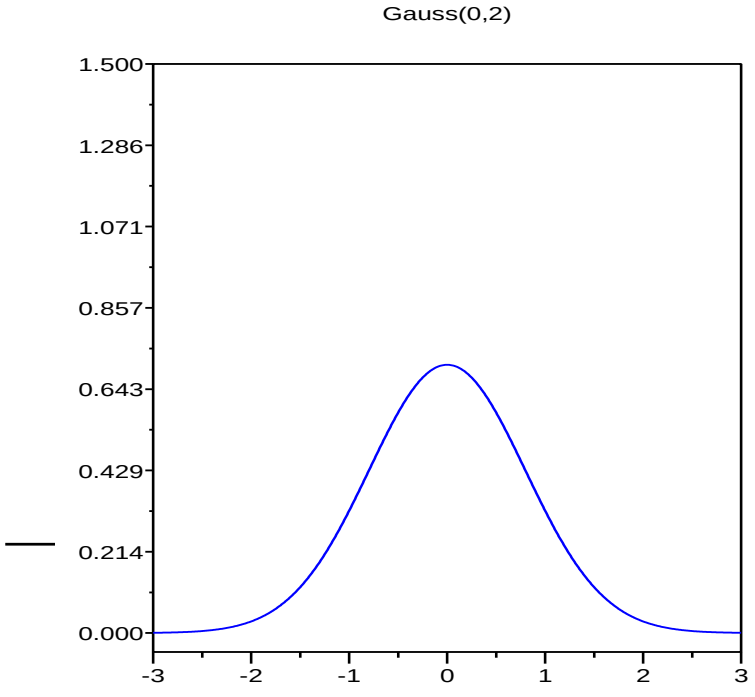
Näide: δ-funktsioon

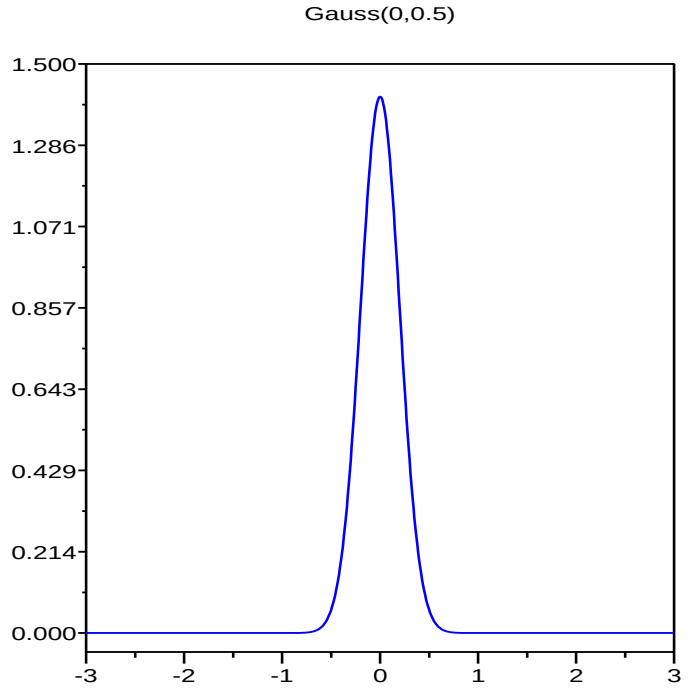
$$\mathcal{F}\{\delta(t)\} = \int_{-\infty}^{\infty} \delta(t) e^{-2\pi w t} dt = e^{-2\pi w 0} = 1$$

Näide: gaussiaan

Gaussiaan jääb gaussiaaniks:

$$\frac{1}{\sqrt{a}} e^{-\frac{\pi t^2}{a^2}} \leftrightarrow \sqrt{a} e^{-\pi a^2 w^2}$$

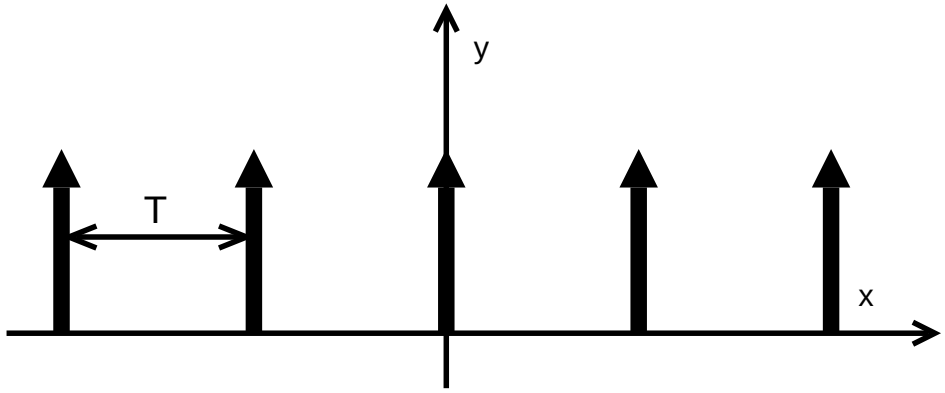




Näide: δ-kamm

Vaatleme δ-funktsioonide „kammi” sammuga T:

$$\delta_T^*(t) = \sum_{k=-\infty}^{\infty} \delta(t - kT)$$



Näide: δ-kamm

$$\mathcal{F}\{\delta_T^*(t)\} = \mathcal{F}\left\{\sum_{k=-\infty}^{\infty} \delta(t-kT)\right\} =$$

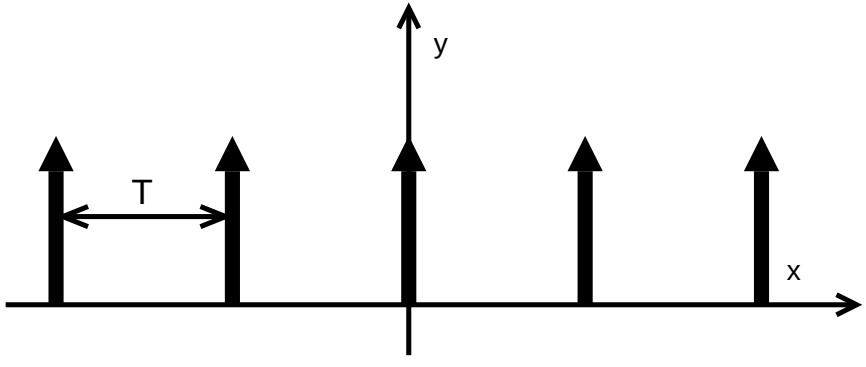
$$= \sum_{k=-\infty}^{\infty} \mathcal{F}\{\delta(t-kT)\} = \sum_{k=-\infty}^{\infty} e^{-i2\pi wkT} =$$

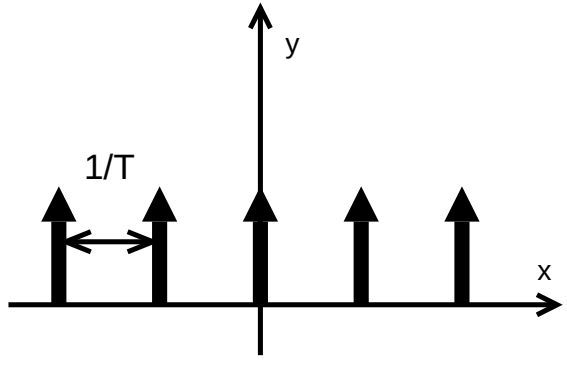
$$= \frac{1}{T}\sum_{k=-\infty}^{\infty} \delta\left(w - \frac{k}{T}\right) = \frac{1}{T}\delta_{1/T}^*(w)$$

Näide: δ-kamm

Kamm jääb kammiks:

$$\delta_T^*(t) \leftrightarrow \delta_{1/T}^*(w)$$





Fourier' pööre 2D

- Need olid kõik ühemõõtmelised näited, kuid pildid millega me töötame on kahe muutuja funktsioonid.
- Õnneks on kahemõõtmeline Fourier' pööre täiesti analoogne ühemõõtmelisele ning täpselt samasuguste omadustega:

$$F(w_x, w_y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) e^{-i2\pi(w_x x + w_y y)} dx dy$$

$$f(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(w_x, w_y) e^{i2\pi(w_x x + w_y y)} dw_x dw_y$$

Fourier' pööre omadused

- Fourier pööre on *lineaarteisendus*:

$$\alpha f(t) + g(t) \leftrightarrow \alpha F(w) + G(w)$$

- Fourier pööre on „peaaegu" iseenda pöördteisendus:

$$\mathcal{F}\{\mathcal{F}\{f(t)\}\} = f(-t)$$

- Mida „kiirem" on funktsioon, seda laiem on tema spekter ja vastupidi:

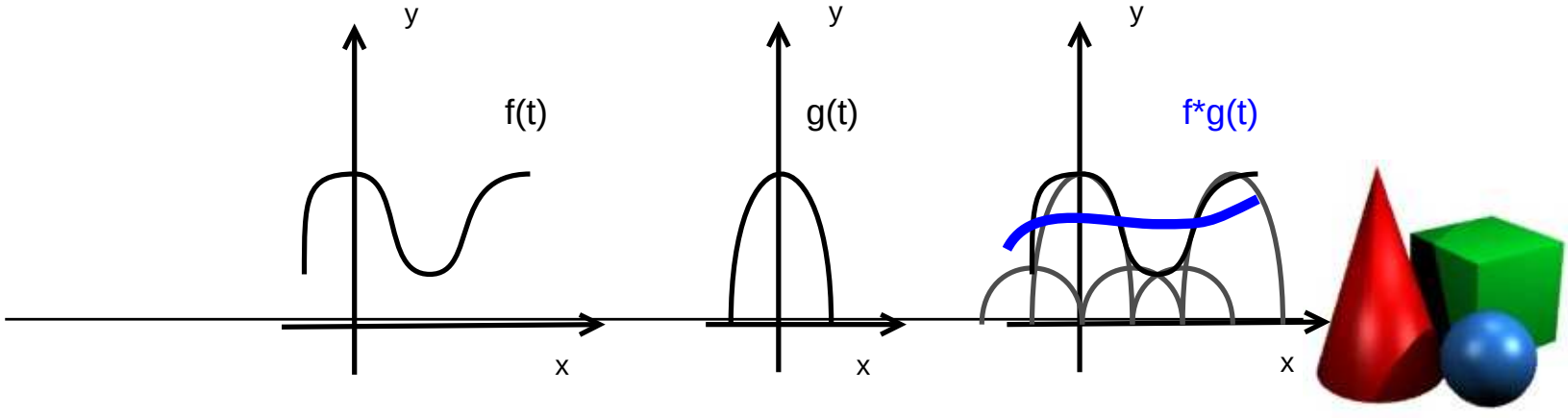
$$\mathcal{F}\{f(at)\} = \frac{1}{a}F(\frac{w}{a})$$

Konvolutsioon

Korrutise Fourier' pööre on Fourier' pöörete *konvolutsioon* ja vastupidi:

$$f(t)g(t) \leftrightarrow F(w) * G(w) \qquad f(t) * g(t) \leftrightarrow F(w)G(w)$$

$$(f * g)(t) = \int_{-\infty}^{\infty} f(x)g(t-x)dx = \int_{-\infty}^{\infty} g(x)f(t-x)dx$$

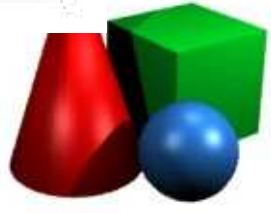


Konvolutsioon

- Konvolutsioon on oluline signaalitöötluse operatsioon.
- Konvolutsiooni nimetatakse tihti ka *lineaarfiltreerimiseks*, diskreetsel juhul ka *libiseva akna meetodiks*.
- Pilditöötluses tuntud filtrid on piltide „udusemaks" tegija *(blur)*, „äärte leidja" *(edge detection)*, jne.
- Helitöötluses teostab konvolutsiooni näiteks ekvalaiser (ja ka enamus muid heli töötlevaid filtreid).

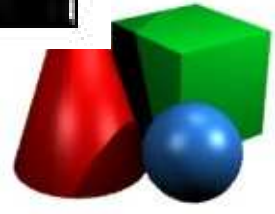
Konvolutsioon: Kastfilter

Pildi diskreetne konvolutsioon 5x5 kastfiltriga:



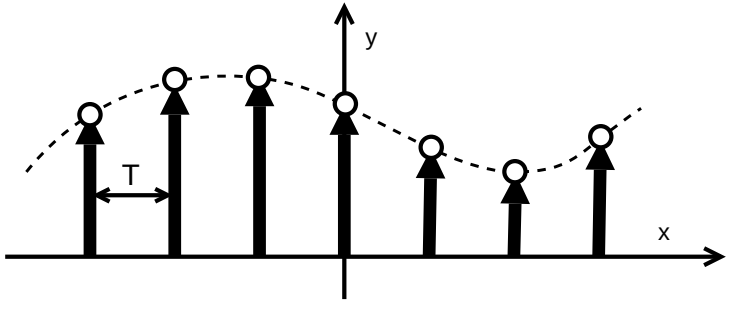
Konvolutsioon: Edge detection

Pildi diskreetne konvolutsioon filtriga $(-1\,0\,1)$:



Fourier' vaade diskretiseerimisele

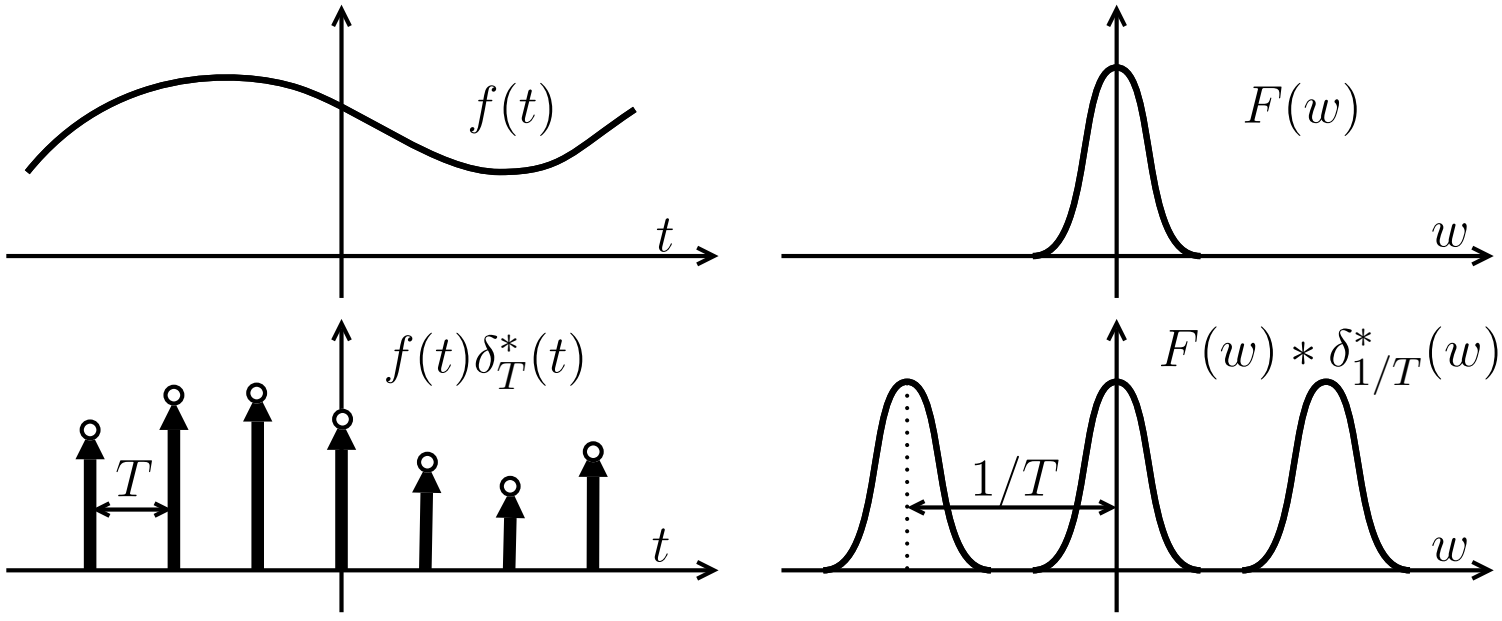
- Nüüd tuleme tagasi diskretiseerimisülesandele.
- Diskretiseerimist esitame kui *korrutamist δ-kammiga*:



$$f_{\text{sampled}}(t) = f(t)\delta_T^*(t)$$

- Uurime f ja f_{sampled} spektrite vahekorra.

Fourier' vaade diskretiseerimisele

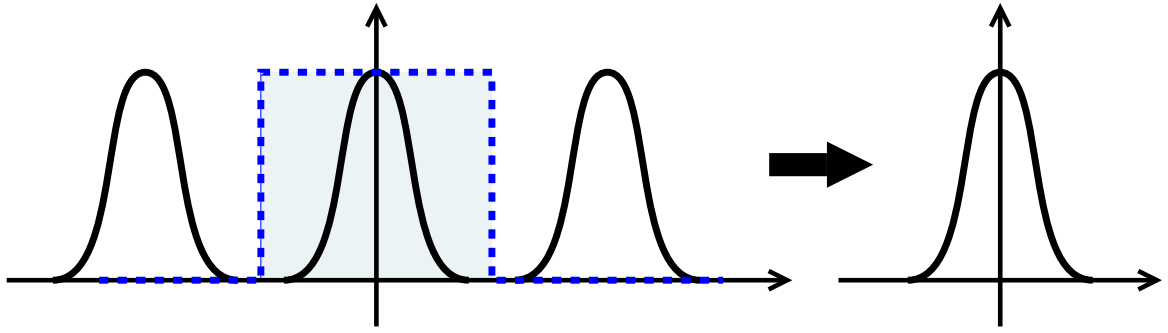


$$f(t) \leftrightarrow F(w)$$

$$f(t)\delta_T^*(t) \leftrightarrow F(w) * \delta_{1/T}^*(w)$$

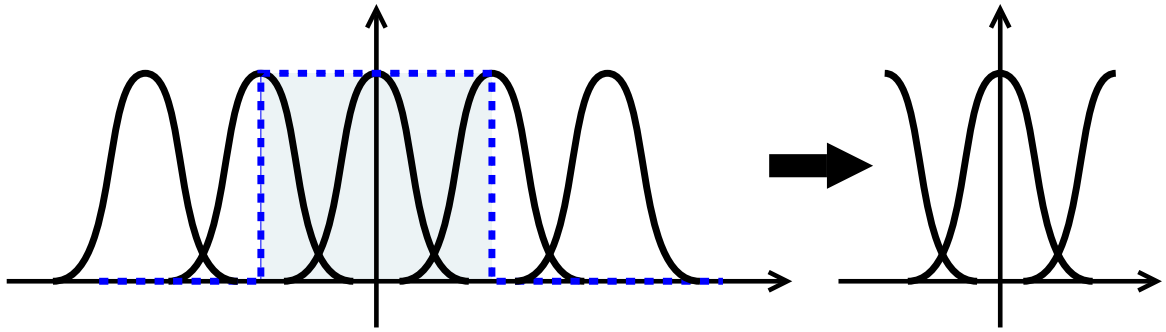
Fourier' vaade diskretiseerimisele

- Kui periood $1/T$ mahutab $F(w)$ täielikult, on võimalik diskretiseeritud signaali spektrist taastada originaalsignaali spektrit, korrutades teda sobiva kastfunktsiooniga:



Fourier' vaade diskretiseerimisele

- Kui periood $1/T$ on liiga väike, originaalspektri $F(w)$ ei ole võimalik taastada:



- Sel juhul satuvad taastatud funktsiooni kõrged sagedused ekslikult tema madalate sageduste hulka ja kõrged sagedused: madalate sageduste hulka. Sellest ka nimetus: *aliasing*.

Nyquist'i teoreem

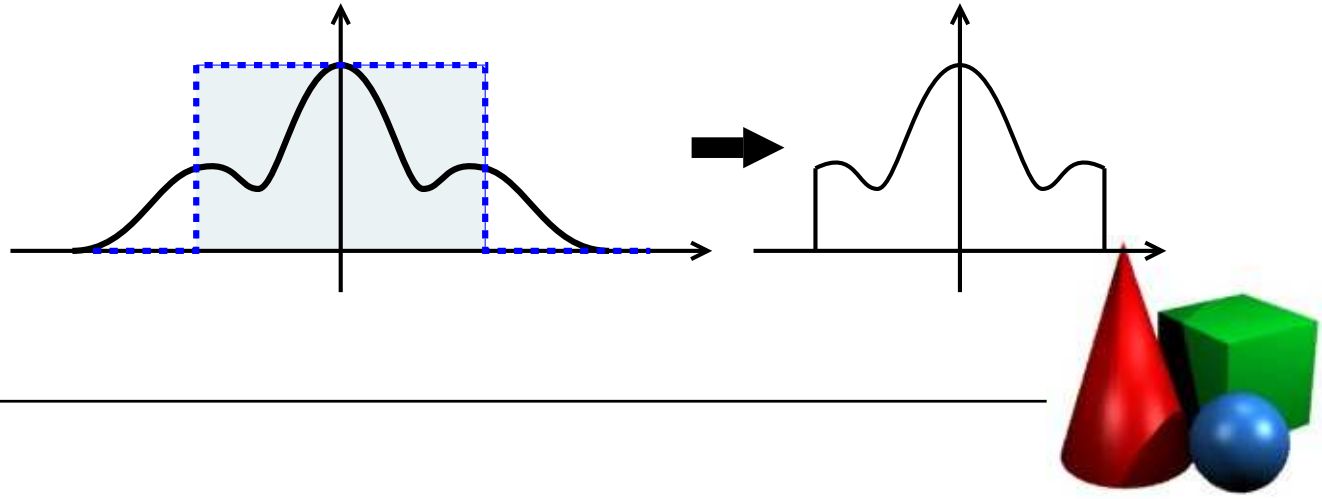
- Niiet selleks et $f(t)$ oleks taastatav peab $F(w)$ „mahtuma" ühte perioodi pikkusega $1/T$. Teisisõnu, peab $1/T$ (ehk diskretiseerimise sagedus) olema vähemalt kaks korda suurem kui signaali maksimaalne sagedus. See ongi Nyquist'i teoreem:

$$w_{\text{sampling}} > 2\max(w_f)$$

- Jääb viimane küsimus: mis siis kui me ei saa nii tihedalt diskretiseerida, kuidas saab pildi kõrgeid sagedusi „maha lõigata"?

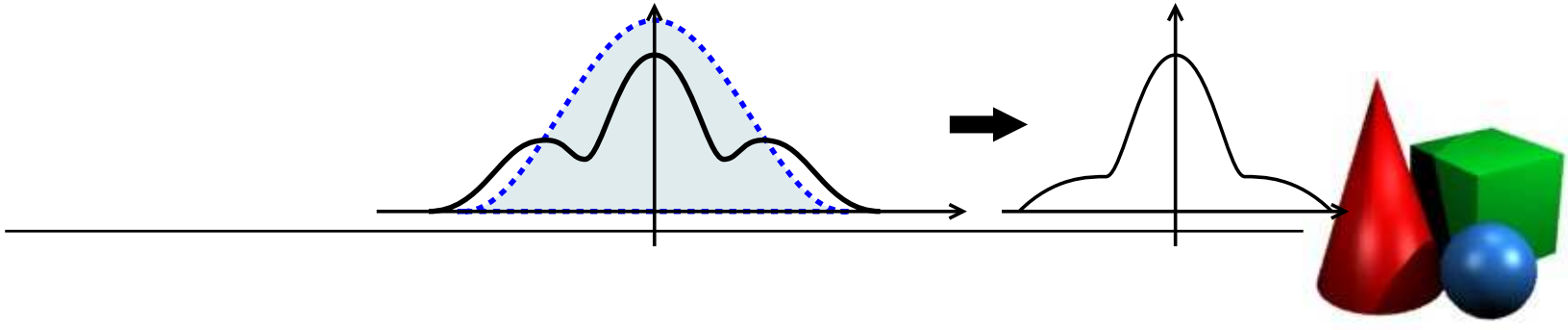
Bandlimiting

- Liiga laia spektriga pilte spektrit peame enne diskretiseerimist maha lõikama.
- Näiteks teravate äärtega piltide spekter on lõpmata! (vt. kastfunktsiooni spekter).
- Ideaalis peaksime liiga laia spektri korrutama kastfunktsiooniga:



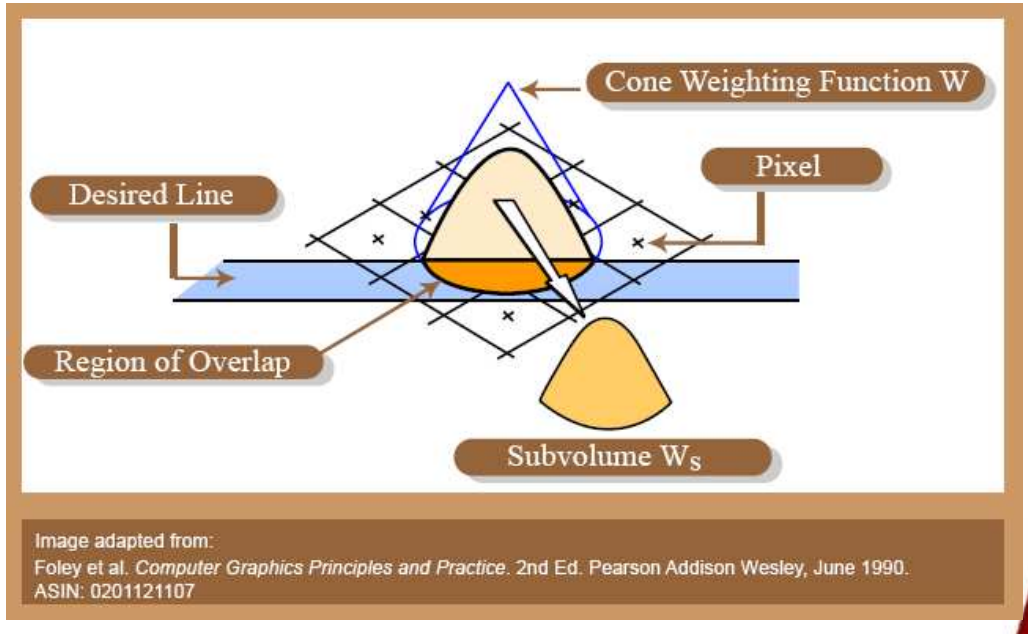
Bandlimiting

- Pildi spektri korrutamine kastfunktsiooniga vastab pildi enda konvolutsioonile sinc funktsiooniga.
- sinc funktsiooniga konvolutsiooni arvutamine on ebaefektiivne kuna tema määramispiirkond on lõpmata.
- Selle asemel lõikame sagedusi gaussiaaniga. Sellele vastab konvolutsioon ka gaussiaaniga (ehk „blur”-filtreerimine):



Anti-aliasing

- Konvolutsioon gaussiaaniga või temale sarnase funktsiooniga ongi kõige tüüpilisem praktiline meetod heade piltide genereerimiseks.



Anti-aliasing

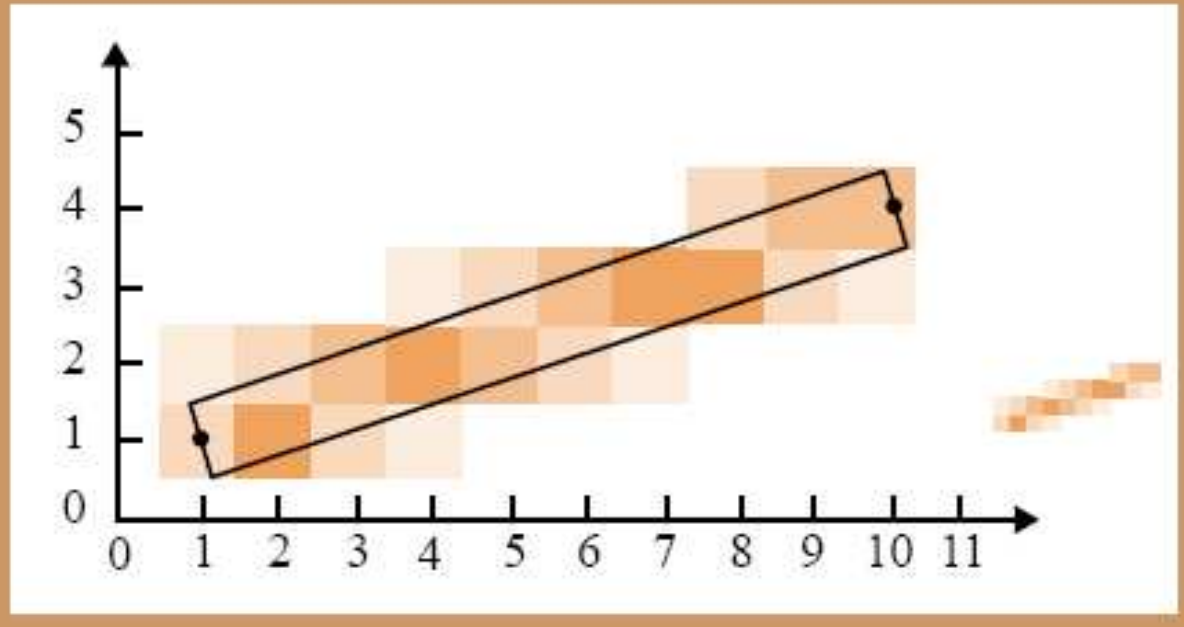


Image adapted from:
Foley et al. *Computer Graphics Principles and Practice*. 2nd Ed. Pearson Addison Wesley, June 1990. ASIN: 0201121107

Anti-aliasing

- Üks lihtne meetod teostada konvolutsiooni gaussiaaniga pildi renderdamise käigus on kasutada akkumuleerimispuhvrit:
  - Liidame akkumuleerimispuhvris mitu kaadrit.
  - Iga kaader on nihutatud juhusliku nihe x, y võrra ning korrutatud gaussiaani väärtusega selles punktis $G(x, y)$.
- Analoogne idee töötab ka raytracingus.

Diskretiseerimine: Kokkuvõte

- Diskretiseerimise sagedus peab olema rohkem kui kaks korda kõrgem pildi kõrgeimast sagedusest.
- Teravate äärtega pilte filtreerime eelnevalt madalsagedusfiltriga (ideaalis sinc, praktikas gaussiaan).
- Eelmistes loengutes ja praksides tegelesime algoritmidega ilma anti-aliasinguta. Tegelikult enamus „ilusaid" renderdamisalgoritme *peavad* sisaldama mõnda aliasingu-tõrjet.
- Tõsi küll, anti-aliasing on üsna kallis lõbu, ning mängudes sellest tihti loobutakse.

Taastamine (reconstruction)

- Peale diskretiseerimist peame suutma pildi õieti *taastama*. Pilditöötluse terminites on ülesanne pildi resolutsiooni suurendamises (ja diskretiseerimine vastas tegelikult pildi resolutsiooni vähendamise probleemile).

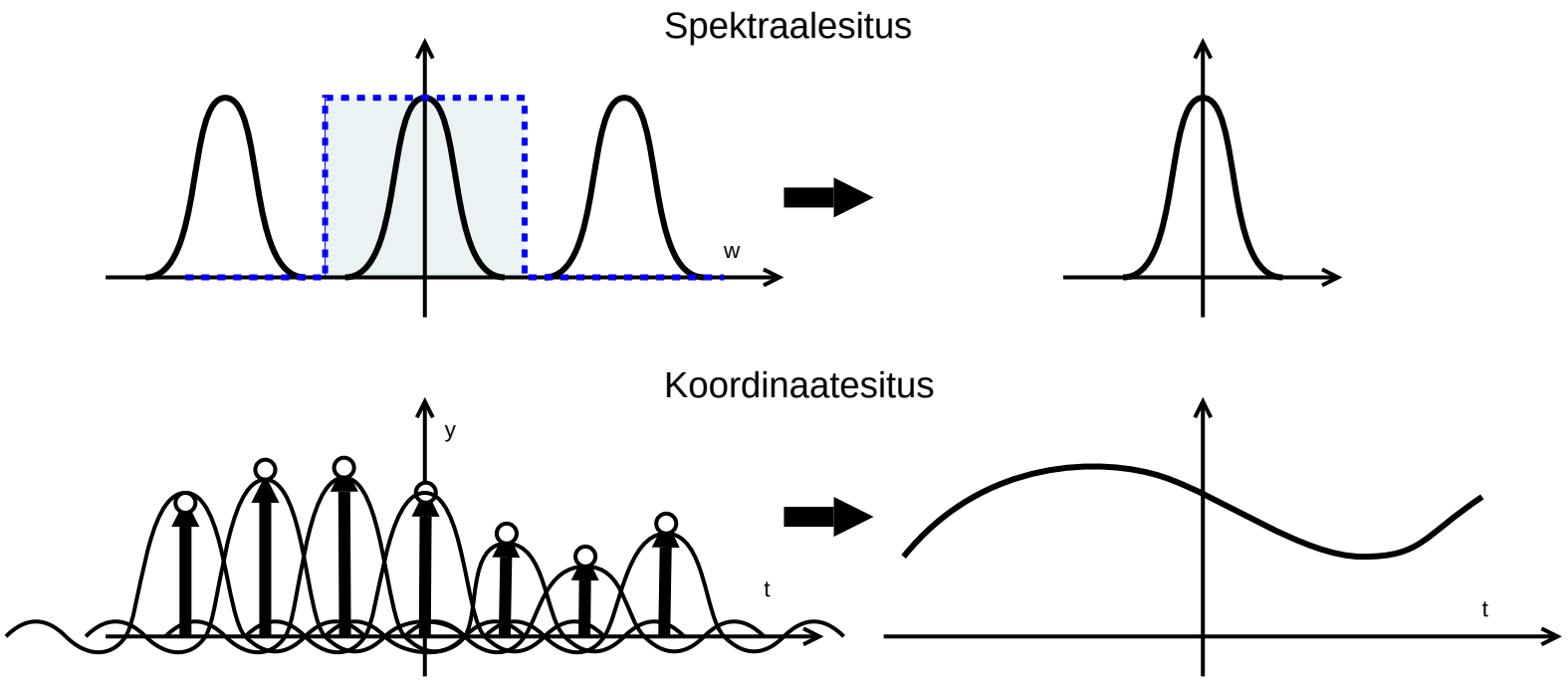
Nearest neighbor

- Ilmselt kõige levinum taastamise viis on nn. *lähima naabri meetod*, kus me oletame et pikslid on tegelikult väiksed ruudukesed. Muidugi pole see õige mõtlemisviis.

Kuidas tegelikult asi käima peab

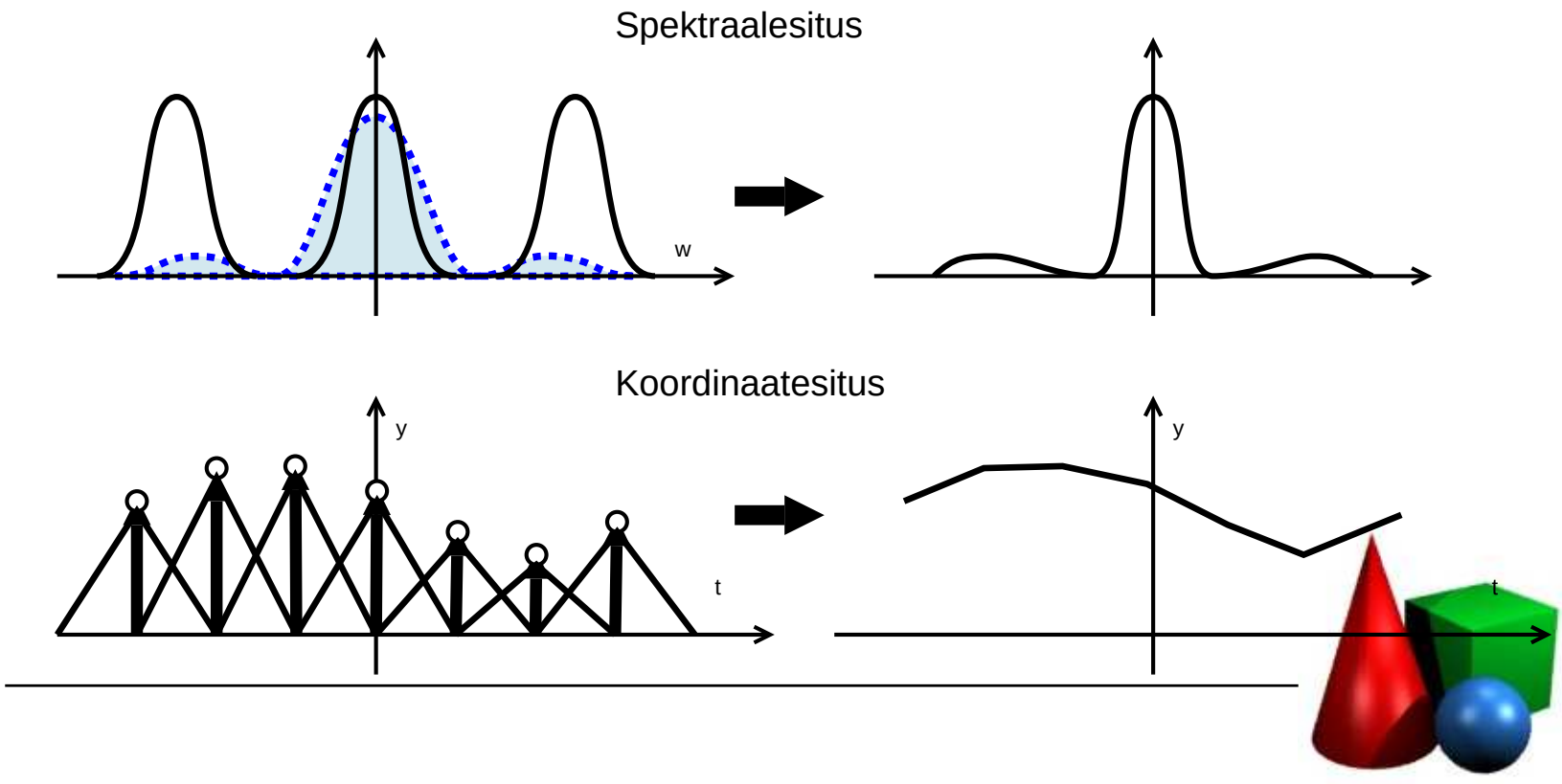
- Tegelikult peame me taastamiseks *korrutama diskretiseeritud signaali spektri sobiva kastfunktsiooniga.*
- See tähendab, ideaalis peame me taastamiseks *filtreerima diskretiseeritud signaali* sinc funktsiooniga.
- See tähendab, iga pildi punkti taastamiseks peaks tegelikult arvestama kõigi pikslite väärtustega.

Kuidas tegelikult asi käima peab



(Bi-)lineaarne filter

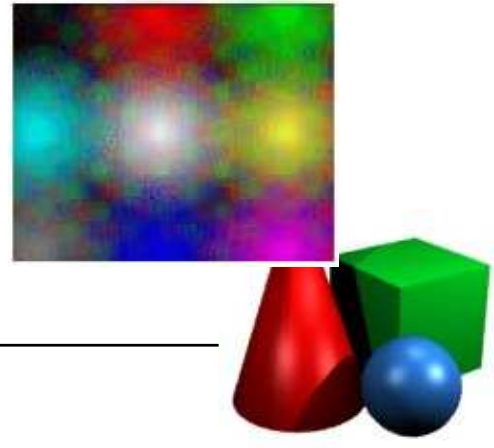
- Kasutame efektiivsemaid lahendusi. Näiteks lineaarset interpoleerimist:



Nearest neighbor vs Bilinear filter

Gaussian filter

- Veel üks alternatiiv on gaussi-filter. Natuke parem kui lineaarfilter, kuid arvutuslikult kallim.
- CRT monitorid teevad Gaussi-filtriga taastamist enda pikslitele ise:



Kokkuvõte

- Diskretiseerimine
  - Peab olema tehtud õige sagedusega.
  - Soovitavalt madalsagedusfiltriga
- Taastamine
  - On teostatav konvolutsiooniga (ehk filtreerimisega)
  - Hea oleks sinc-filter, kuid praktikas kasutatakse bilineaarset interpoleerimist või gaussi-filtrit.

Mõtlemiseks

- Tüüpiliselt tulevad taastamine ja diskretiseerimine ette kombinatsioonina, mille nimi on *resampling*:
  - Tekstuurimine
  - Pilditöötlus programmides piltide resolutsiooni (ehk „suuruse") muutmine, piltide pööramine mitte täisnurga võrra, jne.
- Mõelge kodus selle peale, kuidas ja mis filtritega resamplimist õieti realiseerima peaks ja võiks (näiteks: kuidas pöörata pildi 45 kraadi võrra).

Mõtlemiseks

- Kuidas lahendada ajalise aliasingu probleemi?

Mõtlemiseks

- Mis on pilt?
- Mis on piksel?

Küsimused

?