UNIVERZA V LJUBLJANI
FAKULTETA ZA MATEMATIKO IN FIZIKO

Matematika – 2. stopnja

Katarina Zadražnik

Likovno upodabljanje slik

Magistrsko delo

Mentor: prof. dr. Andrej Bauer

Ljubljana, 2015

Podpisana Katarina Zadražnik izjavljam:

– da sem magistrsko delo z naslovom Likovno upodabljanje slik izdelala samostojno pod mentorstvom prof. dr. Andreja Bauerja, ter

– da Fakulteti za matematiko in fiziko Univerze v Ljubljani dovoljujem objavo elektronske oblike svojega dela na spletnih straneh.

Ljubljana, 26. december 2016 Podpis:

Zahvala

Kazalo

Program dela

Kraj, datum

prof. dr. Mentor Mentorski

prof. dr. Somentor Somentorski

Povzetek

Abstract

Math. Subj. Class. (MSC 2010): XXXXXX, YYYYYY
Ključne besede: foo, bar, baz
Keywords: foo, bar, baz

Poglavje 1

Obdelava slik

Pri obdelavi signalov in v teoriji filtrov je Fourierova transformacija še vedno zelo močno orodje, ki se ga med drugim lahko uporablja za odstranjevanje šuma, pohitritev računanja konvolucije dveh matrik in izboljšavo kakovosti signala.

V prvem razdelku tega poglavja si bomo zato najprej pogledali teoretično ozadje za Fourierovo transformacijo in njeno izboljšano različico (hitro Fourierovo transformacijo). V našem delu se bomo ukvarjali le z obdelavo slik, zato bomo v pretežni meri vso teorijo in izpeljane postopke aplicirali na slike, čeprav bo vse skupaj v prilagojenih različicah veljalo tudi za ostale vrste signalov.

V drugem razdelku bomo nato spoznali nekaj osnovnih pojmov in postopkov iz teorije filtrov. Našteli bomo osnovne vrste transformacij slike, se naučili postopkov za njihov izračun in nato predstavili osnovne filtrirne postopke (zaznava robov na sliki, zameglitev slike, glajenje robov na sliki ...), ki jih bomo uporabljali v algoritmih za likovno upodabljanje slik.

V tretjem razdelku se bomo posvetili barvnim prostorom RGB, CMY, HSV in YUV. Spoznali bomo tudi Kubelka-Monk barvni model, ki ga bomo uporabili v algoritmu za likovno upodabljanje slik z voščenkami.

Pri obdelavi slik je postopek odstranjevanja šuma zelo pomemben, zato si bomo v zadnjem razdelku tega poglavja pogledali različne algoritme za odstranjevanje šuma. Gaussovo filtriranje je postopek za odstranjevanje šuma, ki je sicer hiter, vendar pa nam večkrat vrne nezadovoljive rezultate. Na drugi strani pa bomo spoznali še nelokalno odstranjevanje šuma s slike, ki da zelo dobre rezultate, vendar pa ima kljub temu, da lahko nekatere izračune v algoritmu pohitrimo, še vedno kvadratno časovno zahtevnost. V algoritmih za likovno upodabljanje slik bomo odvisno od naših trenutnih prioritet (hitrost ali kakovost) izbrali ustrezni postopek za odstranjevanje šuma.

1.1 Fourierova transformacija

V tem razdelku bomo predstavili teorijo, ki nas bo vodila do algoritma za izračun dvodimenzionalne hitre (diskretne) Fourierove transformacije. Teorijo bomo povzeli po [**?**, 2. poglavje] in njene rezultate skupaj z dokazi zapisali za dvodimenzionalno različico ter jih opremili z lastnimi primeri.

1.1.1 Fourierova baza

Najprej bomo definirali N-dimenzionalni vektorski prostor $\ell^2(\mathbb{Z}_N)$ nad kompleksnimi števili:

$$\ell^2(\mathbb{Z}_N) \coloneqq \{z = (z(0), z(1), \ldots, z(N-1)) \mid z(j) \in \mathbb{C}, 0 \leq j \leq N-1\}\,.$$

Prostor bomo opremili s standardno evklidsko bazo $E = \{e_0, e_1, \ldots, e_{N-1}\}$, kjer za bazni element e_j velja: $e_j(n) = 1$ za $j = n$ in $e_j(n) = 0$ sicer. V nadaljevanju bomo definirali prostor $\ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2})$, ga opremili s standardno bazo in nato spoznali Fourierovo bazo.

Definicija 1.1.1 Za naravni števili N_1 in N_2 definiramo

$$\ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2}) \coloneqq \{z : \mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2} \to \mathbb{C}\}\,.$$

Elemente $z \in \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2})$ lahko zapišemo tudi z matriko

$$z = \begin{bmatrix} z(0,0) & z(0,1) & \ldots & z(0,N_2-1) \\ z(1,0) & z(1,1) & \ldots & z(1,N_2-1) \\ \vdots & \vdots & \ddots & \vdots \\ z(N_1-1,0) & z(N_1-1,1) & \ldots & z(N_1-1,N_2-1) \end{bmatrix},$$

kjer so za $0 \leq n_1 \leq N_1 - 1, 0 \leq n_2 \leq N_2 - 1$ vrednosti $z(n_1, n_2) \in \mathbb{C}$.

Z običajnim seštevanjem in množenjem s skalarjem po komponentah postane množica $\ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2})$ vektorski prostor nad $\mathbb{C}$. Za elementa $z, w \in \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2})$ definiramo kompleksni skalarni produkt

$$\langle z, w \rangle \coloneqq \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} z(n_1, n_2) \cdot \overline{w(n_1, n_2)}\,. \tag{1.1}$$

Trditev 1.1.2 *Prostor $\ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2})$ je N_1N_2-dimenzionalen.*

Dokaz. Dimenzija prostora je enaka razsežnosti njegove baze. Definiramo množico

$$E \coloneqq \{e_{i,j} \mid 0 \leq i \leq N_1 - 1, 0 \leq j \leq N_2 - 1\}\,,$$

kjer je $e_{i,j}(n_1, n_2) = 1$ za $(i, j) = (n_1, n_2)$ in $e_{i,j}(n_1, n_2) = 0$ sicer. Zanjo enostavno preverimo, da je linearno neodvisna in razpenja celoten prostor $\ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2})$, torej je N_1N_2-razsežna baza prostora $\ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2})$. ∎

Izrek 1.1.3 *Naj bosta $\{B_0, B_1, \ldots, B_{N_1-1}\}$ in $\{C_0, C_1, \ldots, C_{N_2-1}\}$ ortonormirani bazi za $\ell^2(\mathbb{Z}_{N_1})$ in $\ell^2(\mathbb{Z}_{N_2})$. Za $0 \leq m_1 \leq N_1 - 1$ in $0 \leq m_2 \leq N_2 - 1$ naj bo*

$$D_{m_1,m_2}(n_1, n_2) \coloneqq B_{m_1}(n_1) C_{m_2}(n_2)\,.$$

Tedaj je $\{D_{m_1,m_2}\}_{0 \leq m_1 \leq N_1-1, 0 \leq m_2 \leq N_2-1}$ ortonormirana baza prostora $\ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2})$.

Dokaz. Najprej preverimo, da velja enakost $\langle D_{m_1,m_2}, D_{k_1,k_2}\rangle = \langle B_{m_1}, B_{k_1}\rangle\langle C_{m_2}, C_{k_2}\rangle$. Z uporabo definicije skalarnega produkta (1.1) in s preureditvijo členov v dvojni vsoti dobimo:

$$\begin{aligned}
\langle D_{m_1,m_2}, D_{k_1,k_2}\rangle &= \sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1} D_{m_1,m_2}(n_1,n_2)\overline{D_{k_1,k_2}(n_1,n_2)} \\
&= \sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1} B_{m_1}(n_1)C_{m_2}(n_2)\overline{B_{k_1}(n_1)C_{k_2}(n_2)} \\
&= \sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1} B_{m_1}(n_1)\overline{B_{k_1}(n_1)}C_{m_2}(n_2)\overline{C_{k_2}(n_2)} \\
&= \sum_{n_1=0}^{N_1-1} B_{m_1}(n_1)\overline{B_{k_1}(n_1)}\sum_{n_2=0}^{N_2-1} C_{m_2}(n_2)\overline{C_{k_2}(n_2)} \\
&= \langle B_{m_1}, B_{k_1}\rangle\langle C_{m_2}, C_{k_2}\rangle.
\end{aligned}$$

Z računom

$$\langle D_{m_1,m_2}, D_{k_1,k_2}\rangle = \langle B_{m_1}, B_{k_1}\rangle\langle C_{m_2}, C_{k_2}\rangle = \begin{cases} 1 & \text{če } (m_1,m_2)=(k_1,k_2); \\ 0 & \text{sicer} \end{cases}$$

pokažemo, da je baza $\{D_{m_1,m_2}\}_{0\le m_1\le N_1-1, 0\le m_2\le N_2-1}$ res ortonormirana (v računu uporabimo dejstvo, da sta bazi $\{B_{m_1}\}_{0\le m_1\le N_1-1}$ in $\{C_{m_2}\}_{0\le m_2\le N_2-1}$ ortonormirani). ∎

V vektorskem prostoru $\ell^2(\mathbb{Z}_N)$ so ortonormirani bazni elementi za $m,n\in\{0,\dots,N-1\}$ definirani kot $E_m(n) \coloneqq \frac{1}{\sqrt{N}} e^{\frac{2\pi\imath mn}{N}}$. Za $m_1\in\{0,\dots,N_1-1\}$ in $m_2\in\{0,\dots,N_2-1\}$ definiramo elemente $E_{m_1,m_2}\in\ell^2(\mathbb{Z}_{N_1}\times\mathbb{Z}_{N_2})$ s formulo

$$E_{m_1,m_2}(n_1,n_2) \coloneqq \frac{1}{\sqrt{N_1N_2}} e^{\frac{2\pi\imath m_1n_1}{N_1}} e^{\frac{2\pi\imath m_2n_2}{N_2}}.$$

Spodnja trditev je direktna posledica zgornjega izreka.

Trditev 1.1.4 *Množica* $\{E_{m_1,m_2}\}_{0\le m_1\le N_1-1, 0\le m_2\le N_2-1}$ *je ortonormirana baza vektorskega prostora* $\ell^2(\mathbb{Z}_{N_1}\times\mathbb{Z}_{N_2})$.

Bazne elemente E_{m_1,m_2} pomnožimo z $\frac{1}{\sqrt{N_1N_2}}$ in dobimo nove bazne elemente

$$F_{m_1,m_2}(n_1,n_2) \coloneqq \frac{1}{N_1N_2} e^{\frac{2\pi\imath m_1n_1}{N_1}} e^{\frac{2\pi\imath m_2n_2}{N_2}},$$

ki pa posledično niso več normirani.

Definicija 1.1.5 Množica $F \coloneqq \{F_{m_1,m_2}\}_{0\le m_1\le N_1-1, 0\le m_2\le N_2-1}$ je *Fourierova baza* prostora $\ell^2(\mathbb{Z}_{N_1}\times\mathbb{Z}_{N_2})$.

Z uporabo zveze $e^{\imath\varphi} = \cos\varphi + \imath \cdot \sin\varphi$ lahko Fourierove bazne elemente zapišemo še na drug način s formulo

$$F_{m_1,m_2}(n_1,n_2) = \frac{1}{N_1N_2}\left(\cos\left[2\pi\left(\frac{m_1n_1}{N_1}+\frac{m_2n_2}{N_2}\right)\right] + \imath\sin\left[2\pi\left(\frac{m_1n_1}{N_1}+\frac{m_2n_2}{N_2}\right)\right]\right) .$$

Primer 1.1.6 Fourierova baza prostora $\ell^2(\mathbb{Z}_2 \times \mathbb{Z}_3)$ je množica

$$F = \{F_{0,0}, F_{0,1}, F_{0,2}, F_{1,0}, F_{1,1}, F_{1,2}\},$$

kjer so bazni elementi enaki:

$$F_{0,0} = \tfrac{1}{6}\cdot\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}, \qquad F_{1,0} = \tfrac{1}{6}\cdot\begin{bmatrix}1 & 1 & 1\\ -1 & -1 & -1\end{bmatrix},$$

$$F_{0,1} = \tfrac{1}{6}\cdot\begin{bmatrix}1 & -\frac{1}{2}+\frac{\imath\sqrt{3}}{2} & -\frac{1}{2}-\frac{\imath\sqrt{3}}{2}\\ 1 & -\frac{1}{2}+\frac{\imath\sqrt{3}}{2} & -\frac{1}{2}-\frac{\imath\sqrt{3}}{2}\end{bmatrix}, \quad F_{1,1} = \tfrac{1}{6}\cdot\begin{bmatrix}1 & -\frac{1}{2}+\frac{\imath\sqrt{3}}{2} & -\frac{1}{2}-\frac{\imath\sqrt{3}}{2}\\ -1 & \frac{1}{2}-\frac{\imath\sqrt{3}}{2} & \frac{1}{2}+\frac{\imath\sqrt{3}}{2}\end{bmatrix},$$

$$F_{0,2} = \tfrac{1}{6}\cdot\begin{bmatrix}1 & -\frac{1}{2}-\frac{\imath\sqrt{3}}{2} & -\frac{1}{2}+\frac{\imath\sqrt{3}}{2}\\ 1 & -\frac{1}{2}-\frac{\imath\sqrt{3}}{2} & -\frac{1}{2}+\frac{\imath\sqrt{3}}{2}\end{bmatrix}, \quad F_{1,2} = \tfrac{1}{6}\cdot\begin{bmatrix}1 & -\frac{1}{2}-\frac{\imath\sqrt{3}}{2} & -\frac{1}{2}+\frac{\imath\sqrt{3}}{2}\\ -1 & \frac{1}{2}+\frac{\imath\sqrt{3}}{2} & \frac{1}{2}-\frac{\imath\sqrt{3}}{2}\end{bmatrix}.$$

◊

1.1.2 Dvodimenzionalna diskretna Fourierova transformacija

Dvodimenzionalna diskretna Fourierova transformacija (kratica 2DFT) je preslikava

$$\mathcal{F} : \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2}) \to \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2}),$$

ki je za $z \in \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2})$ definirana s formulo

$$\mathcal{F}(z)(m_1,m_2) \coloneqq \sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1} z(n_1,n_2) e^{\frac{-2\pi\imath m_1n_1}{N_1}} e^{\frac{-2\pi\imath m_2n_2}{N_2}} .$$

Dvodimenzionalna diskretna inverzna Fourierova transformacija (kratica 2IFT) je preslikava

$$\mathcal{F}^{-1} : \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2}) \to \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2}),$$

ki je za $w \in \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2})$ definirana s formulo

$$\mathcal{F}^{-1}(w)(n_1,n_2) \coloneqq \frac{1}{N_1N_2}\sum_{m_1=0}^{N_1-1}\sum_{m_2=0}^{N_2-1} w(m_1,m_2) e^{\frac{2\pi\imath m_1n_1}{N_1}} e^{\frac{2\pi\imath m_2n_2}{N_2}} .$$

Enostavno lahko preverimo, da sta zgornji preslikavi dobro definirani in res slikata nazaj v isti prostor.

Ker je $\{E_{m_1,m_2}\}_{0\le m_1\le N_1-1, 0\le m_2\le N_2-1}$ ortonormirana baza, lahko $z \in \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2})$ razvijemo po tej bazi kot

$$z = \sum_{m_1=0}^{N_1-1}\sum_{m_2=0}^{N_2-1} \langle z, E_{m_1,m_2}\rangle E_{m_1,m_2} . \tag{1.2}$$

Pri tem je

$$\langle z, E_{m_1,m_2}\rangle = \sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1} z(n_1,n_2)\overline{\frac{1}{\sqrt{N_1N_2}}e^{\frac{2\pi\imath m_1 n_1}{N_1}}e^{\frac{2\pi\imath m_2 n_2}{N_2}}}\ . \tag{1.3}$$

S pomočjo tega razvoja bomo preprosto pokazali, da velja naslednja trditev.

Trditev 1.1.7 *Za vsak $z \in \ell^2(\mathbb{Z}_{N_1}\times\mathbb{Z}_{N_2})$ velja:*

$$z = (\mathcal{F}^{-1}\mathcal{F})(z).$$

Dokaz. Z upoštevanjem formul (1.2) in (1.3) naredimo naslednji izračun:

$$\begin{aligned}
&(\mathcal{F}^{-1}\mathcal{F})(z)(k_1,k_2) = \\
&= \frac{1}{N_1N_2}\sum_{m_1=0}^{N_1-1}\sum_{m_2=0}^{N_2-1}\left(\sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1} z(n_1,n_2)e^{\frac{-2\pi\imath m_1 n_1}{N_1}}e^{\frac{-2\pi\imath m_2 n_2}{N_2}}\right)e^{\frac{2\pi\imath k_1 m_1}{N_1}}e^{\frac{2\pi\imath k_2 m_2}{N_2}} \\
&= \sum_{m_1=0}^{N_1-1}\sum_{m_2=0}^{N_2-1}\left(\sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1} z(n_1,n_2)\frac{1}{\sqrt{N_1N_2}}e^{\frac{-2\pi\imath m_1 n_1}{N_1}}e^{\frac{-2\pi\imath m_2 n_2}{N_2}}\right)\frac{1}{\sqrt{N_1N_2}}e^{\frac{2\pi\imath k_1 m_1}{N_1}}e^{\frac{2\pi\imath k_2 m_2}{N_2}} \\
&= \sum_{m_1=0}^{N_1-1}\sum_{m_2=0}^{N_2-1}\langle z, E_{m_1,m_2}\rangle E_{m_1,m_2}(k_1,k_2) = z(k_1,k_2)\ .
\end{aligned}$$

Ker ta izračun velja za vsak par $(k_1,k_2)\in\ell^2(\mathbb{Z}_{N_1}\times\mathbb{Z}_{N_2})$, je trditev s tem dokazana. ∎

Primer 1.1.8 Naj bo $z\in\ell^2(\mathbb{Z}_2\times\mathbb{Z}_3)$, npr.

$$z = \begin{bmatrix} 3\imath & 10 & 10-\imath \\ 2 & 0 & -\imath \end{bmatrix}.$$

Na tem primeru bomo sedaj preverili, da velja trditev 1.1.7. Za vsak par $(m_1,m_2)\in\mathbb{Z}_2\times\mathbb{Z}_3$ izračunamo vrednost $\mathcal{F}(z)(m_1,m_2)$ in dobimo matriko

$$\mathcal{F}(z) = w = \begin{bmatrix} 22+\imath & -8+\sqrt{3}+4\imath & -8-\sqrt{3}+4\imath \\ 18+3\imath & -12+3\imath & -12+3\imath \end{bmatrix}.$$

Če na matriki w sedaj uporabimo 2IFT, dobimo nazaj matriko z. Torej res velja $z = (\mathcal{F}^{-1}\mathcal{F})(z)$. ◊

Bolj kot razvoj po standardni bazi bo za nas zanimiv razvoj po Fourierovi bazi. Iz izračuna v zadnjem dokazu je razvidno, da je

$$z = \sum_{m_1=0}^{N_1-1}\sum_{m_2=0}^{N_2-1}\left(\sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1} z(n_1,n_2)\frac{1}{\sqrt{N_1N_2}}e^{\frac{-2\pi\imath m_1 n_1}{N_1}}e^{\frac{-2\pi\imath m_2 n_2}{N_2}}\right)E_{m_1,m_2}. \tag{1.4}$$

S preureditvijo členov in uporabo definicije Fourierove baze dobimo, da je

$$\begin{aligned}
z &= \sum_{m_1=0}^{N_1-1}\sum_{m_2=0}^{N_2-1}\left(\sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1} z(n_1,n_2)e^{\frac{-2\pi\imath m_1 n_1}{N_1}}e^{\frac{-2\pi\imath m_2 n_2}{N_2}}\right)F_{m_1,m_2} \\
&= \sum_{m_1=0}^{N_1-1}\sum_{m_2=0}^{N_2-1}\mathcal{F}(z)(m_1,m_2)F_{m_1,m_2}.
\end{aligned}$$

Kadar z razvijemo po Fourierovi bazi, pravimo, da smo ga zapisali v *frekvenčni domeni.*[1]

Primer 1.1.9 Element z iz primera 1.1.8,

$$z = \begin{bmatrix} 3\imath & 10 & 10-\imath \\ 2 & 0 & -\imath \end{bmatrix},$$

bomo zapisali v Fourierovi bazi:

$$\begin{aligned} z = (22+\imath)\cdot F_{0,0} + (-8+\sqrt{3}+4\imath)\cdot F_{0,1} + (-8-\sqrt{3}+4\imath)\cdot F_{0,2} + \\ + (18+3\imath)\cdot F_{1,0} + (-12+3\imath)\cdot F_{1,1} + (-12+3\imath)\cdot F_{1,2}, \end{aligned}$$

kjer so F_{n_1,n_2} Fourierovi bazni elementi prostora $\ell^2(\mathbb{Z}_2 \times \mathbb{Z}_3)$. ◊

Izrek 1.1.10 *Za $z, w \in \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2})$ veljata naslednji dve formuli:*

1. *Parsevalova formula:*
$$\langle z, w\rangle = \frac{1}{N_1 N_2}\langle \mathcal{F}(z), \mathcal{F}(w)\rangle;$$

2. *Plancherelova formula:*
$$\|z\|^2 = \frac{1}{N_1 N_2}\|\mathcal{F}(z)\|^2.$$

Dokaz. Najprej bomo dokazali Parsevalovo formulo. S primerjavo formul (1.2) in (1.4) opazimo, da velja $\mathcal{F}(z(m_1,m_2)) = \sqrt{N_1N_2}\cdot\langle z, E_{m_1,m_2}\rangle$. Z uporabo te ugotovitve in formule (1.1) za izračun skalarnega produkta dobimo, da je

$$\begin{aligned} \langle z, w\rangle &= \sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1} z(n_1,n_2)\cdot\overline{w(n_1,n_2)} \\ &= \sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1} \frac{1}{\sqrt{N_1N_2}}\mathcal{F}(z(n_1,n_2))\frac{1}{\sqrt{N_1N_2}}\mathcal{F}(w(n_1,n_2)). \end{aligned}$$

S tem smo pokazali veljavnost Parsevalove formule. Plancherelovo formulo enostavno dokažemo tako, da ponovimo postopek za $w = z$ in uporabimo definicijo norme. ∎

1.1.3 Translacijsko invariantne linearne transformacije

Transformacija je matematični zapis sistema, ki poskrbi za pretvorbo vhodnega signala v izhodnega. Sistemi, ki jih modeliramo s transformacijo, so običajno linearni (pripadajoča transformacija je linearna) in invariantni (v primeru časovno ali prostorsko zamaknjenega vhodnega signala, ima te lastnosti tudi izhodni signal). V našem delu bomo modelirali sivinske slike, zato se bomo sedaj osredotočili na dvodimenzionalne transformacije.

Naj bo sedaj z zaporedje, ki ga definiramo na množici $\mathbb{Z}\times\mathbb{Z}$ (elementi zaporedja so $z(n_1,n_2)\in\mathbb{C}$ za $(n_1,n_2)\in\mathbb{Z}\times\mathbb{Z}$). Za zaporedje z pravimo, da je *periodično* v prvi

[1] V razdelku **??** bomo videli fizikalno interpretacijo zapisa v Fourierovi bazi in naredili njihovo vizualno predstavitev.

spremenljivki s periodo N_1 in v drugi spremenljivki s periodo N_2, če za vse $n_1, n_2, j_1, j_2 \in \mathbb{Z}$ velja

$$z(n_1 + j_1 N_1, n_2 + j_2 N_2) = z(n_1, n_2) \, .$$

Definicija 1.1.11 Za $k_1, k_2 \in \mathbb{Z}$ je *translacijsko invariantna linearna transformacija*

$$R_{k_1,k_2} : \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2}) \to \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2})$$

definirana s formulo

$$(R_{k_1,k_2} z)(n_1, n_2) \coloneqq z(n_1 - k_1, n_2 - k_2) \, .$$

Pravimo, da je transformacija $T : \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2}) \to \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2})$ *translacijsko invariantna*, če za vse $k_1, k_2 \in \mathbb{Z}$ in $z \in \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2})$ velja:

$$T(R_{k_1,k_2} z) = R_{k_1,k_2} T(z).$$

Trditev 1.1.12 *Če je transformacija $T : \ell^2(Z_{N_1} \times Z_{N_2}) \to \ell^2(Z_{N_1} \times Z_{N_2})$ translacijsko invariantna, tedaj za vsak Fourierov bazni element F_{m_1,m_2} velja, da je lastni vektor transformacije T.*

Dokaz. Naj bo F_{m_1,m_2} Fourierov bazni element v prostoru $\ell^2(Z_{N_1} \times Z_{N_2})$. Ker je F baza prostora $\ell^2(Z_{N_1} \times Z_{N_2})$, obstajajo taka kompleksna števila $\{a_{i,j}\}_{i,j=0}^{i=N_1-1,j=N_2-1}$, da za vsak par $(n_1, n_2) \in \mathbb{N} \times \mathbb{N}$ velja:

$$T(F_{m_1,m_2})(n_1, n_2) = \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} a_{k_1,k_2} F_{k_1,k_2}(n_1, n_2) \tag{1.5}$$

$$= \frac{1}{N_1 N_2} \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} a_{k_1,k_2} e^{\frac{2\pi\imath k_1 n_1}{N_1}} e^{\frac{2\pi\imath k_2 n_2}{N_2}} . \tag{1.6}$$

Po definiciji preslikave R_{k_1,k_2} velja:

$$\begin{aligned}(R_{1,0} F_{m_1,m_2})(n_1, n_2) &= F_{m_1,m_2}(n_1 - 1, n_2) = \frac{1}{N_1 N_2} e^{\frac{2\pi\imath m_1 (n_1-1)}{N_1}} e^{\frac{2\pi\imath m_2 n_2}{N_2}} \\ &= \frac{1}{N_1 N_2} e^{-\frac{2\pi\imath m_1}{N_1}} e^{\frac{2\pi\imath m_1 n_1}{N_1}} e^{\frac{2\pi\imath m_2 n_2}{N_2}} \\ &= e^{-\frac{2\pi\imath m_1}{N_1}} F_{m_1,m_2}(n_1, n_2) \, .\end{aligned}$$

Ker je T linearna preslikava in je člen $e^{-\frac{2\pi\imath m_1}{N_1}}$ neodvisen od spremenljivk n_1 in n_2, z nadaljno uporabo enačbe (1.5), velja:

$$\begin{aligned}T(R_{1,0} F_{m_1,m_2})(n_1, n_2) &= e^{-\frac{2\pi\imath m_1}{N_1}} T(F_{m_1,m_2})(n_1, n_2) \\ &= e^{-\frac{2\pi\imath m_1}{N_1}} \frac{1}{N_1 N_2} \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} a_{k_1,k_2} e^{\frac{2\pi\imath k_1 n_1}{N_1}} e^{\frac{2\pi\imath k_2 n_2}{N_2}} \\ &= \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} e^{-\frac{2\pi\imath m_1}{N_1}} a_{k_1,k_2} F_{k_1,k_2}(n_1, n_2) \, .\end{aligned}$$

Dobili smo razvoj $T(R_{1,0}F_{m_1,m_2})(n_1,n_2)$ po Fourierovi bazi. Sedaj bomo s pomočjo (1.5) razvili po Fourierovi bazi še $(R_{1,0}T(F_{m_1,m_2}))(n_1,n_2)$:

$$\begin{aligned} R_{1,0}(T(F_{m_1,m_2}))(n_1,n_2) &= T(F_{m_1,m_2})(n_1-1,n_2) \\ &= \frac{1}{N_1N_2}\sum_{k_1=0}^{N_1-1}\sum_{k_2=0}^{N_2-1} a_{k_1,k_2}e^{\frac{2\pi\imath k_1(n_1-1)}{N_1}}e^{\frac{2\pi\imath k_2 n_2}{N_2}} \\ &= \sum_{k_1=0}^{N_1-1}\sum_{k_2=0}^{N_2-1} a_{k_1,k_2}e^{-\frac{2\pi\imath k_1}{N_1}}F_{k_1,k_2}(n_1,n_2)\ . \end{aligned}$$

Ker smo predpostavili, da je T translacijsko invariantna transformacija, za vsak par $(n_1,n_2)\in\mathbb{Z}\times\mathbb{Z}$ velja, da je $T(R_{1,0}(F_{m_1,m_2}))(n_1,n_2) = R_{1,0}(T(F_{m_1,m_2}))(n_1,n_2)$. Zaradi enoličnosti razvoja po bazi sledi, da morajo biti koeficienti v zgornjih dveh razvojih enaki. Za vsak par $(k_1,k_2)\in\mathbb{Z}_{N_1}\times\mathbb{Z}_{N_2}$ velja:

$$a_{k_1,k_2}e^{-\frac{2\pi i m_1}{N_1}} = a_{k_1,k_2}e^{-\frac{2\pi\imath k_1}{N_1}}.$$

Za $k_1\neq m_1$, velja $e^{-\frac{2\pi\imath m_1}{N_1}}\neq e^{-\frac{2\pi\imath k_1}{N_1}}$, ker je $0\leq k_1,m_1\leq N_1-1$. Če želimo, da velja enakost v zgornji enačbi, mora biti za $k_1\neq m_1$ koeficient $a_{k_1,k_2}=0$.

Podobno bi dobili, da mora biti za $k_2\neq m_2$ koeficient $a_{k_1,k_2}=0$, če bi v zgornjih izračunih namesto $R_{1,0}$ vzeli $R_{0,1}$. Dobili smo, da mora biti za $(k_1,k_2)\neq(m_1,m_2)$ koeficient $a_{k_1,k_2}=0$. V enačbi (1.5) zato odpadejo vsi členi, razen tistega pri $(k_1,k_2)=(m_1,m_2)$. Dobimo, da je $T(F_{m_1,m_2})(n_1,n_2) = a_{m_1,m_2}F_{m_1,m_2}(n_1,n_2)$ oz.

$$T(F_{m_1,m_2}) = a_{m_1,m_2}F_{m_1,m_2},$$

kar dokazuje, da je F_{m_1,m_2} lastni vektor transformacije T z lastno vrednostjo a_{m_1,m_2}. S tem je dokazana celotna trditev, saj smo to dokazali za splošen F_{m_1,m_2}. ∎

Posledica 1.1.13 *Translacijsko invariantne linearne transformacije T so diagonalizabilne v Fourierovi bazi.*

Zgornja posledica enostavno sledi iz ravnokar dokazane trditve. V nadaljevanju bomo definirali operator $*$, na podlagi katerega bomo znali hitro izračunati FFT kasneje ob uporabi te posledice.

Definicija 1.1.14 Za vse $z,w\in\ell^2(\mathbb{Z}_{N_1}\times\mathbb{Z}_{N_2})$ in $(m_1,m_2)\in\mathbb{Z}_{N_1}\times\mathbb{Z}_{N_2}$ definiramo operator $z*w\in\ell^2(\mathbb{Z}_{N_1}\times\mathbb{Z}_{N_2})$ s formulo

$$z*w(m_1,m_2) \coloneqq \sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1} z(m_1-n_1,m_2-n_2)w(n_1,n_2)\ .$$

Primer 1.1.15 Naj bosta $z=\left(\begin{smallmatrix}1&1\\0&\imath\end{smallmatrix}\right)$ in $w=\left(\begin{smallmatrix}\imath&0\\1&\imath\end{smallmatrix}\right)$ elementa prostora $\ell^2(\mathbb{Z}_2\times\mathbb{Z}_2)$. Z

upoštevanjem periodičnosti izračunamo

$$\begin{aligned} z * w(0,0) &= \sum_{n_1=0}^{1} \sum_{n_2=0}^{1} z(-n_1,-n_2)w(n_1,n_2) \\ &= z(0,0)w(0,0) + z(0,-1)w(0,1) + z(-1,0)w(1,0) + z(-1,-1)w(1,1) \\ &= z(0,0)w(0,0) + z(0,1)w(0,1) + z(1,0)w(1,0) + z(1,1)w(1,1) \\ &= 1 \cdot \imath + 1 \cdot 0 + 0 \cdot 1 + \imath \cdot \imath = -1 + \imath. \end{aligned}$$

Podobno izračunamo preostale tri vrednosti:

$$z * w(0,1) = \sum_{n_1=0}^{1} \sum_{n_2=0}^{1} z(-n_1,1-n_2)w(n_1,n_2) = 1 \cdot \imath + 1 \cdot 0 + \imath \cdot 1 + 1 \cdot \imath = 3\imath,$$

$$z * w(1,0) = \sum_{n_1=0}^{1} \sum_{n_2=0}^{1} z(1-n_1,-n_2)w(n_1,n_2) = 0 \cdot \imath + \imath \cdot 0 + 1 \cdot 1 + 1 \cdot \imath = 1 + \imath,$$

$$z * w(1,1) = \sum_{n_1=0}^{1} \sum_{n_2=0}^{1} z(1-n_1,1-n_2)w(n_1,n_2) = \imath \cdot \imath + 0 \cdot 0 + 1 \cdot 1 + 1 \cdot \imath = \imath.$$

Torej je $z * w = \left(\begin{smallmatrix} -1+\imath & 3\imath \\ 1+\imath & \imath \end{smallmatrix}\right)$. ◊

Izrek 1.1.16 *Veljajo naslednje trditve:*

1. *Za vse* $(m_1, m_2) \in \mathbb{Z} \times \mathbb{Z}$ *velja*

$$\mathcal{F}(z * w)(m_1, m_2) = \mathcal{F}(z)(m_1, m_2)\mathcal{F}(w)(m_1, m_2).$$

2. *Za* $b \in \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2})$ *definiramo* $T_b : \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2}) \to \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2})$ *z*

$$T_b(z) = b * z.$$

   Vsaki linearni transformaciji te oblike pravimo, da je konvolucijski operator. *Velja, da je* T_b *translacijsko invariantna preslikava.*

3. *Za* $m \in \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2})$ *definiramo* $T_{(m)} : \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2}) \to \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2})$ *kot*

$$T_{(m)}(z) = \mathcal{F}^{-1}(m\mathcal{F}(z)),$$

   kjer je za vsak (n_1, n_2) $(m\mathcal{F}(z))(n_1, n_2) = m(n_1, n_2) \cdot \mathcal{F}(z)(n_1, n_2)$. *Vsaki linearni transformaciji takega tipa pravimo, da je* Fourierov multiplikativni operator. *Velja, da je vsak konvolucijski operator* T_b, *pri izbiri* $m = \mathcal{F}(b)$, *enak Fourierovemu multiplikativnemu operatorju* $T_{(m)}$.

Dokaz. (1) Z uporabo definicij Fourierove transformacije in konvolucije naredimo naslednji izračun:

$$
\begin{aligned}
&\mathcal{F}(z*w)(m_1,m_2) = \\
&= \sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1} z*w(n_1,n_2) e^{\frac{-2\pi\imath m_1 n_1}{N_1}} e^{\frac{-2\pi\imath m_2 n_2}{N_2}} \\
&= \sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1}\sum_{k_1=0}^{N_1-1}\sum_{k_2=0}^{N_2-1} z(n_1-k_1,n_2-k_2) w(k_1,k_2) e^{\frac{-2\pi\imath m_1 n_1}{N_1}} e^{\frac{-2\pi\imath m_2 n_2}{N_2}} \\
&= \sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1}\sum_{k_1=0}^{N_1-1}\sum_{k_2=0}^{N_2-1} z(n_1-k_1,n_2-k_2) w(k_1,k_2) e^{\frac{-2\pi\imath m_1 (n_1-k_1)}{N_1}} e^{\frac{-2\pi\imath m_2 (n_2-k_2)}{N_2}} e^{\frac{-2\pi\imath m_1 k_1}{N_1}} e^{\frac{-2\pi\imath m_2 k_2}{N_2}} \\
&= \sum_{k_1=0}^{N_1-1}\sum_{k_2=0}^{N_2-1} w(k_1,k_2) e^{\frac{-2\pi\imath m_1 k_1}{N_1}} e^{\frac{-2\pi\imath m_2 k_2}{N_2}} \sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1} z(n_1-k_1,n_2-k_2) e^{\frac{-2\pi\imath m_1 (n_1-k_1)}{N_1}} e^{\frac{-2\pi\imath m_2 (n_2-k_2)}{N_2}} .
\end{aligned}
$$

Zadnjo dvojno vsoto z uvedbo novih spremenljivk $l_1 = n_1 - k_1$ in $l_2 = n_2 - k_2$ prevedemo v:

$$
\begin{aligned}
&\sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1} z(n_1-k_1,n_2-k_2) e^{\frac{-2\pi\imath m_1 (n_1-k_1)}{N_1}} e^{\frac{-2\pi\imath m_2 (n_2-k_2)}{N_2}} = \\
&\qquad = \sum_{l_1=-k_1}^{N_1-1-k_1}\sum_{l_2=-k_2}^{N_2-1-l_2} z(l_1,l_2) e^{\frac{-2\pi\imath m_1 l_1}{N_1}} e^{\frac{-2\pi\imath m_2 l_2}{N_2}} \\
&\qquad = \sum_{l_1=0}^{N_1-1}\sum_{l_2=0}^{N_2-1} z(n_1-k_1,n_2-k_2) e^{\frac{-2\pi\imath m_1 l_1}{N_1}} e^{\frac{-2\pi\imath m_2 l_2}{N_2}} .
\end{aligned}
$$

V zadnjem enačaju smo uporabili, da sta izraza v vsoti periodična s periodo N_1 oz. N_2. Z uvedbo novih spremenljivk dobimo:

$$
\begin{aligned}
&\mathcal{F}(z*w)(m_1,m_2) = \\
&\qquad = \sum_{k_1=0}^{N_1-1}\sum_{k_2=0}^{N_2-1} w(k_1,k_2) e^{\frac{-2\pi\imath m_1 k_1}{N_1}} e^{\frac{-2\pi\imath m_2 k_2}{N_2}} \sum_{l_1=0}^{N_1-1}\sum_{l_2=0}^{N_2-1} z(l_1,l_2) e^{\frac{-2\pi\imath m_1 l_1}{N_1}} e^{\frac{-2\pi\imath m_2 l_2}{N_2}} \\
&\qquad = \mathcal{F}(z)(m_1,m_2)\mathcal{F}(w)(m_1,m_2),
\end{aligned}
$$

kar dokazuje prvo točko izreka, saj zgornje velja za poljubna m_1 in m_2.

(2) Naj bo $z \in \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2})$ in $(k_1,k_2) \in \mathbb{Z}\times\mathbb{Z}$. Tedaj za vsak par $(k_1,k_2) \in \mathbb{Z}_{N_1}\times\mathbb{Z}_{N_2}$ velja naslednji izračun:

$$
\begin{aligned}
T_b(R_{k_1,k_2}z)(m_1,m_2) &= b*(R_{k_1,k_2}z)(m_1,m_2) \\
&= \sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1} b(m_1-n_1,m_2-n_2)(R_{k_1,k_2}z)(n_1,n_2) \\
&= \sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1} b(m_1-n_1,m_2-n_2) z(n_1-k_1,n_2-k_2) .
\end{aligned}
$$

Z uvedbo novih spremenljivk $l_1 = n_1 - k_1$ in $l_2 = n_2 - k_2$ v zadnji vsoti in upoštevanjem periodičnosti, za vsak par $(m_1, m_2) \in \mathbb{Z} \times \mathbb{Z}$ velja

$$\begin{aligned} T_b(R_{k_1,k_2}z)(m_1,m_2) = & \\ &= \sum_{l_1=-k_1}^{N_1-1-k_1} \sum_{l_2=-k_2}^{N_2-1-k_2} b(m_1-k_1-l_1, m_2-k_2-l_2)z(l_1,l_2) \\ &= \sum_{l_1=0}^{N_1-1} \sum_{l_2=0}^{N_2-1} b(m_1-k_1-l_1, m_2-k_2-l_2)z(l_1,l_2) \\ &= (b*z)(m_1-k_1, m_2-k_2) \\ &= R_{k_1,k_2}(b*z)(m_1,m_2) \\ &= R_{k_1,k_2}T_b(z)(m_1,m_2). \end{aligned}$$

S tem smo pokazali, da je $T_b(R_{k_1,k_2}z) = R_{k_1,k_2}T_b(z)$, tj. T_b je res translacijski operator.

(3) Z uporabo prve točke tega izreka in definicije inverzne Fourierove transformacije za vsak $z \in \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2})$ izračunamo:

$$T_b(z) = b * z = \mathcal{F}^{-1}(b*z) = \mathcal{F}^{-1}(\mathcal{F}(b)\mathcal{F}(z)) = \mathcal{F}^{-1}(m\mathcal{F}(z)) = T_{(m)}(z),$$

kar dokazuje tretjo točko izreka. ∎

Od tod sledi naslednji izrek, v katerem smo dokazali, ostale implikacije pa sledijo enostavno.

Izrek 1.1.17 *Naj bo $T : \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2}) \to \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2})$ linearna transformacija. Naslednje trditve so ekvivalentne:*

1. *T je translacijsko invariantni operator;*
2. *T je Fourierov multiplikativni operator;*
3. *T je konvolucijski operator.*

1.1.4 Hitra Fourierova transformacija

V tem podrazdelku bomo razvili algoritem za dvodimenzionalno hitro diskretno Fourierovo transformacijo. Naj bo $z \in \ell^2(\mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2})$. Za izračun $\mathcal{F}(z)(m_1, m_2)$ potrebujemo N_1N_2 kompleksnih množenj.[2] Za celotni izračun $\mathcal{F}(z)$ torej potrebujemo $N_1^2N_2^2$ kompleksnih množenj.

Če razpišemo izračun za $\mathcal{F}(z)$, opazimo, da se nekateri členi v njej večkrat ponovijo. V naslednji trditvi bomo izkoristili to ugotovitev in s tem zmanjšali potrebno število kompleksnih množenj za izračun rezultata.

[2] Šteli bomo le kompleksna množenja. Operacij seštevanja ne bomo šteli. Število realnih množenj je potem trikrat večje, saj za množenje dveh kompleksnih števil potrebujemo tri realna množenja: $(a+b\imath)(c+d\imath) = (a-b)d + (c-d)a + [(a-b)d + (c+d)b]\imath$.

Trditev 1.1.18 *Naj bo*

$$y(n_1, m_2) \coloneqq \sum_{n_2=0}^{N_2-1} z(n_1, n_2) e^{-\frac{2\pi\imath m_2 n_2}{N_2}}.$$

Tedaj lahko Fourierovo transformacijo zapišemo kot

$$\mathcal{F}(z)(m_1, m_2) = \sum_{n_1=0}^{N_1-1} y(n_1, m_2) e^{-\frac{2\pi\imath m_1 n_1}{N_1}}.$$

Za izračun 2DFT potrebujemo $N_1N_2(N_1 + N_2)$ kompleksnih množenj.

Dokaz. Za izračun $y(n_1, m_2)$ potrebujemo N_2 kompleksnih množenj. Skupno, za vse možne pare (n_1, m_2), torej potrebujemo $(N_1N_2) \cdot N_2$ kompleksnih množenj. Pri izračunu $\mathcal{F}(z)(m_1, m_2)$ upoštevamo, da je vrednost $y(n_1, m_2)$ že znana. Torej za njen izračun potrebujemo N_1 kompleksknih množenj. Za vse možne kombinacije (m_1, m_2) skupaj potrebujemo $(N_1N_2) \cdot N_1$ kompleksnih množenj.

Celotni izračun 2DFT torej zahteva $N_1N_2(N_1 + N_2)$ kompleksnih množenj. ∎

Zgornje število kompleksnih množenj lahko še zmanjšamo. V zgornji trditvi smo izračun 2DFT prevedli na zaporedni izračun dveh enodimenzionalnih DFT. Za vsakega izmed teh izračunov torej potrebujemo največ toliko kompleksnih množenj kot jih zahteva izračun 1DFT. Izkaže se, da lahko v nekaterih primerih izračun 1DFT zelo pohitrimo, o tem se bomo prepričali v naslednjem podrazdelku.

Enodimenzionalna hitra diskretna Fourierova transformacija

Enodimenzionalno hitro diskretno Fourierovo transformacijo $\mathcal{F}$ na vektorju $z \in \ell^2(\mathbb{Z}_N)$ izračunamo po formuli:

$$\mathcal{F}(z)(m) = \sum_{n=0}^{N-1} z(n) e^{-\frac{2\pi\imath mn}{N}}.$$

S pomočjo indukcije bomo pokazali, da lahko časovno zahtevnost izračuna 1DFT izboljšamo, če izkoristimo lastnosti Fourierove transformacije. Kasneje bomo pokazali kako lahko pohitrimo izračun za splošen N, zaenkrat si poglejmo to na primeru, ko je N sodo število, torej $N = 2M$ za $M \in \mathbb{N}$.

Lema 1.1.19 *Naj bo $M \in \mathbb{N}$, $N = 2M$ in $z \in \ell^2(\mathbb{Z}_N)$. Definirajmo $u, v \in \ell^2(\mathbb{Z}_M)$ s formulama $u(k) = z(2k)$ in $v(k) = z(2k + 1)$ za $k \in \{0, 1, \ldots, M - 1\}$. Za $m \in \{0, 1, \ldots, M - 1\}$ je*

$$\mathcal{F}(z)(m) = \mathcal{F}(u)(m) + e^{-\frac{2\pi\imath m}{N}} \mathcal{F}(v)(m)\,. \tag{1.7}$$

Za $m \in \{M, M + 1, \ldots, N - 1\}$ definiramo $l = m - M$, $l \in \{0, 1, \ldots, M - 1\}$. Velja

$$\mathcal{F}(z)(m) = \mathcal{F}(z)(l + M) = \mathcal{F}(u)(l) + e^{-\frac{2\pi\imath l}{N}} \mathcal{F}(v)(l)\,. \tag{1.8}$$

Dokaz. Za $m \in \{0, 1, \ldots, N-1\}$ po definiciji velja

$$\mathcal{F}(z)(m) = \sum_{n=0}^{N-1} z(n) e^{-\frac{2\pi\imath mn}{N}} .$$

To vsoto lahko razbijemo na dve vsoti, pri čemer v prvi delni vsoti vzamemo sode indekse $n = 2k$ in v drugi delni vsoti lihe indekse $n = 2k+1$ za $k \in \{0, 1, \ldots, M-1\}$:

$$\begin{aligned}
\mathcal{F}(z)(m) &= \sum_{k=0}^{M-1} z(2k) e^{-\frac{2\pi\imath 2km}{N}} + \sum_{k=0}^{M-1} z(2k+1) e^{-\frac{2\pi\imath (2k+1)m}{N}} \\
&= \sum_{k=0}^{M-1} u(k) e^{-\frac{2\pi\imath km}{N/2}} + e^{-\frac{2\pi\imath m}{N}} \sum_{k=0}^{M-1} v(k) e^{-\frac{2\pi\imath km}{N/2}} \\
&= \sum_{k=0}^{M-1} u(k) e^{-\frac{2\pi\imath km}{M}} + e^{-\frac{2\pi\imath m}{N}} \sum_{k=0}^{M-1} v(k) e^{-\frac{2\pi\imath km}{M}} .
\end{aligned}$$

Za $m \in \{0, 1, \ldots, M-1\}$ je zadnji izraz enak $\mathcal{F}(u)(m) + e^{-\frac{2\pi\imath m}{N}} \mathcal{F}(v)(m)$, torej dobimo enakost (1.7). Za $m \in \{M, M+1, \ldots, N-1\}$ pišemo $m = l + M$ in vstavimo to v zadnji izraz. Z upoštevanjem dejstva, da je $e^{-\frac{2\pi\imath kl}{M}}$ periodična funkcija s periodo M, dobimo

$$\begin{aligned}
\mathcal{F}(z)(m) &= \sum_{k=0}^{M-1} u(k) e^{-\frac{2\pi\imath k(l+M)}{M}} + e^{-\frac{2\pi\imath (l+M)}{N}} \sum_{k=0}^{M-1} v(k) e^{-\frac{2\pi\imath k(l+M)}{M}} \\
&= \sum_{k=0}^{M-1} u(k) e^{-\frac{2\pi\imath kl}{M}} + e^{-\frac{2\pi\imath l}{N}} \sum_{k=0}^{M-1} v(k) e^{-\frac{2\pi\imath kl}{M}} .
\end{aligned}$$

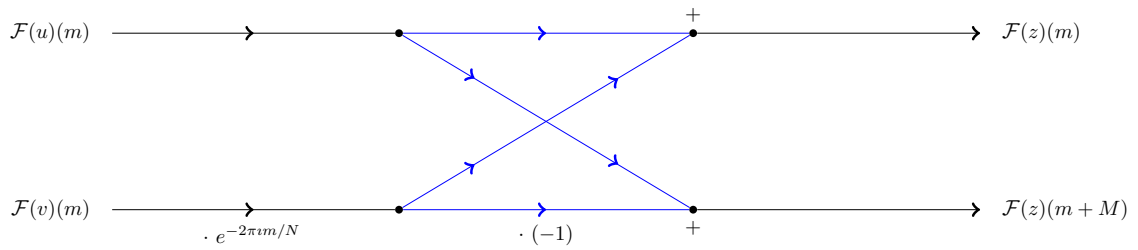
S tem smo pokazali še veljavnost enakosti (1.8). ∎

Za izračun (1.7) in (1.8) potrebujemo iste vrednosti, kar pomeni, da lahko te vrednosti predhodno izračunamo in jih nato uporabimo v omenjenih izračunih. Najprej moramo torej izračunati vrednosti $\mathcal{F}(u)(l)$ in $\mathcal{F}(v)(l)$ za $l \in \{0, 1, \ldots, M-1\}$. Za vsakega izmed izračunov potrebujemo M^2 kompleksnih množenj, saj sta u in v vektorja dolžine $M = \frac{N}{2}$. Za izračun $e^{-\frac{2\pi\imath l}{N}} \mathcal{F}(v)(l)$ potrebujemo nato še dodatnih M kompleksnih množenj. Skupno število za izračun (1.7) in (1.8) je zato enako

$$M^2 + (M^2 + M) = 2M^2 + M = 2\left(\frac{N}{2}\right)^2 + \frac{N}{2} = \frac{N^2 + N}{2} .$$

Če uporabimo postopek iz zgornje leme, za izračun DFT potrebujemo polovico manj kompleksnih množenj kot bi jih v primeru, ko bi računali DFT direktno po formuli. Postopek iz leme je najbolj osnovni primer hitre Fourierove transformacije. Zanimiv je grafični prikaz postopka izračuna iz leme, saj ima diagram (glej sliko 1.1.4), ki prikazuje postopek, obliko *metulja*.

Originalni problem izračuna Fourierove transformacije na vektorju dolžine N znamo z nkeja računske manipulacije torej prevesti na podproblema velikosti $N/2$. Za podprobleme sode velikosti bi lahko zgornji postopek ponovili. V primeru, ko je N potenca števila 2, tj. $N = 2^n$ za $n \in \mathbb{N}$, bomo lahko problem razdelili vse do podproblemov velikosti 1. Zaradi lažjega zapisa, bomo za število potrebnih kompleksnih množenj pri izračunu DFT na vektorju dolžine N, uvedli oznako $E(N)$.



Slika 1.1: Metulj - osnovni primer izračuna DFT. Pri štetju operacij, ki jih potrebujemo pri programski opremi (hardware), velkokrat štejejo število operacij v enotah metulja.

Lema 1.1.20 *Naj bo $N = 2^n$ za $n \in \mathbb{N}$. Tedaj velja*

$$E(N) \leq \frac{1}{2} N \log_2 N.$$

Dokaz. Dokaz naredimo z indukcijo na $n \in \mathbb{N}$. Baza indukcije je v primeru $n = 1$, ko je vektor z dolžine 2, $z = (a, b)$. Z direktno uporabo enodimenzionalne Fourierove transformacije lahko izračunamo $\mathcal{F}(z) = (a+b, a-b)$, za kar ne potrebujemo kompleksnih množenj. Dobimo, da je $E(2) \leq \frac{1}{2} 2 \log_2 2 = 1$, kar pomeni, da v primeru $n = 1$ lema velja. Predpostavimo sedaj, da lema velja za $n-1$ in dokažimo, da velja za n. Iz postopka v dokazu prejšnje leme je razvidno, da je za izračun vektorja z dolžine $N = 2M$ potrebnih največ $2E(M)+M$ kompleksnih množenj. Z uporabo te opazke in indukcijske predpostavke sledi izračun

$$E(2^n) \leq 2E(2^{n-1}) + 2^{n-1} \leq 2\frac{1}{2} 2^{n-1} \log_2 2^{n-1} + 2^{n-1} = n2^{n-1} = \frac{1}{2} n2^n = \frac{1}{2} N \log_2 N,$$

ki pokaže, da lema velja za vsak $n \in \mathbb{N}$. ∎

Sedaj pa si oglejmo primer, ko N ni sodo število, ampak gre za sestavljeno število $N = pq$. V tem primeru lahko namreč nekoliko posplošimo zgornjo lemo.

Lema 1.1.21 *Naj bosta $p, q \in \mathbb{N}$, $N = pq$ in $z \in \ell^2(\mathbb{Z}_N)$. Za $k \in \{0, 1, \ldots, q-1\}$ definiramo zaporedje vektorjev $w_0, w_1, \ldots, w_{p-1} \in \ell^2(\mathbb{Z}_q)$ s formulo*

$$w_l(k) = z(kp + l).$$

Zaporedje vektorjev $v_0, v_1, \ldots, v_{q-1} \in \ell^2(\mathbb{Z}_p)$ definiramo za $l \in \{0, 1, \ldots, p-1\}$ s formulo

$$v_b(l) = e^{-\frac{2\pi\imath bl}{N}} \mathcal{F}(w)(b).$$

Tedaj za $a \in \{0, 1, \ldots, p-1\}$ in $b \in \{0, 1, \ldots, q-1\}$ velja

$$\mathcal{F}(z)(aq + b) = \mathcal{F}(v_b)(a). \tag{1.9}$$

Dokaz. Osnovni izrek o deljenju naravnih števil nam pove, da lahko vsako število $m \in \{0,1,\ldots,N-1\}$ zapišemo v obliki $m = aq + b$ za $a \in \{0,1,\ldots,p-1\}$ in $b \in \{0,1,\ldots,q-1\}$. Torej znamo po lemi s formulo (1.9) izračunati $\mathcal{F}(z)(m)$ za vsak $m \in \{0,1,\ldots,q-1\}$.

Po osnovnem izreku o deljenju lahko zapišemo vsak $n \in \{0,1,\ldots,N-1\}$ kot $n = kp+l$ za $k \in \{0,1,\ldots,p-1\}$ in $l \in \{0,1,\ldots,q-1\}$. Z uporabo tega zapisa števila n in novo definiranimi zaporedji iz predpostavk leme izračunamo:

$$\begin{aligned}\mathcal{F}(z)(aq+b) &= \sum_{n=0}^{N-1} z(n)e^{-\frac{2\pi\imath(aq+b)n}{N}} = \sum_{l=0}^{p-1}\sum_{k=0}^{q-1} z(kp+l)e^{-\frac{2\pi\imath(aq+b)(kp+l)}{pq}} \\ &= \sum_{l=0}^{p-1} e^{-2\pi\imath al} p e^{-\frac{2\pi\imath bl}{N}} \sum_{k=0}^{q-1} w_l(k)e^{-\frac{2\pi\imath bk}{q}} = \sum_{l=0}^{p-1} e^{-\frac{2\pi\imath al}{p}} e^{-\frac{2\pi\imath bl}{N}} \mathcal{F}(w_l)(b) \\ &= \sum_{l=0}^{p-1} e^{-\frac{2\pi\imath al}{p}} v_b(l) = \mathcal{F}(v_b)(a).\end{aligned}$$

■

Sedaj pa si poglejmo koliko kompleksnih množenj je potrebno v primeru, ko je $N = pq$ sestavljeno število. Ideja je podobna kot v zgornjem dokazu. Najprej moramo izračunati vektorje $\mathcal{F}(w_l)$ za $l \in \{0,1,\ldots,p-1\}$. Za vsak posamezen $\mathcal{F}(w_l)$ potrebujemo $E(q)$ kompleksnih množenj, skupno torej $pE(q)$. Dobljene vektorje $\mathcal{F}(w_l)(b)$ dolžine q moramo nato pomnožiti z $e^{-\frac{2\pi\imath bl}{N}}$, kar zahteva dodatnih pq kompleksnih množenj. Ko tako dobimo vektorje $v_b(a)$, moramo izračunati $\mathcal{F}(v_b)(a)$ za $b \in \{0,1,\ldots,q-1\}$, kar zahteva dodatnih $qE(p)$ kompleksnih množenj. Skupno je torej za izračun FFT vektorja dolžine pq potrebnih

$$E(pq) \leq pE(q) + qE(p) + pq \tag{1.10}$$

kompleksnih množenj. Kaj pa se zgodi v primeru, ko je N praštevilo? Tedaj na vektorju ne moremo uporabiti FFT. Vendar pa še ni vse izgubljeno, vektor lahko namreč na koncu podaljšamo z ničlami do željene velikosti, ki omogoča izvedbo FFT. Zaradi dodanih ničel namreč ne povzročimo velike škode, izračun pa postane opazno hitrejši.

Od tu dalje bomo sedaj predpostavili, da je N potenca števila 2, tj. $N = 2^n$. Tedaj lahko $m \in \{0,1\ldots N-1\}$ zapišemo v obliki polinoma $m = m_0 + m_1 \cdot 2 + m_2 \cdot 2^2 + \ldots + m_{n-1} \cdot 2^{n-1}$, kjer so $m_0, m_1, \ldots, m_{n-1} \in \{0,1\}$. Za $z \in \ell^2(\mathbb{Z}_N)$ bomo pisali $z(m) = z(m_{n-1}, m_n, \ldots, m_1, m_0)$. Za $k = k_0 + k_1 \cdot 2 + k_2 \cdot 2^2 + \ldots + k_{n-1} \cdot 2^{n-1}$, $k_0, k_1, \ldots, k_{n-1} \in \{0,1\}$ izračunamo

$$\begin{aligned}&\mathcal{F}(z)(k) = \sum_{m=0}^{N-1} z(m)e^{-\frac{2\pi\imath km}{N}} \\ &= \sum_{m_0=0}^{1}\sum_{m_1=0}^{1}\cdots\sum_{m_{n-1}=0}^{1} z(m_{n-1}, m_{n-2}, \ldots, m_1, m_0)\cdot \\ &\cdot \exp\left(\frac{-2\pi\imath(k_0 + k_1 2 + k_2 2^2 + \ldots + k_{n-1}2^{n-1})(m_0 + m_1 2 + m_2 2^2 + \ldots + m_{n-1}2^{n-1})}{2^n}\right).\end{aligned}$$

Eksponent razbijemo na produkte glede na vsoto $m_0 + m_1 2 + m_2 2^2 + \ldots + m_{n-1}2^{n-1}$, poenostavimo dobljeni izraz in ga vstavimo v formulo za $\mathcal{F}(z)(k)$. Dobimo:

$$\mathcal{F}(z)(k) = \sum_{m_0=0}^{1}\sum_{m_1=0}^{1}\cdots\sum_{m_{n-1}=0}^{1} z(m_{n-1}, m_{n-2}, \ldots, m_1, m_0)\cdot\exp\left(\frac{-2\pi\imath k_0 2^{n-1} m_{n-1}}{2^n}\right)$$
$$\cdot\exp\left(\frac{-2\pi\imath(k_0 + k_1 2)2^{n-2} m_{n-2}}{2^n}\right)\cdots\exp\left(\frac{-2\pi\imath(k_0 + k_1 2 + \cdots + k_{n-1}2^{n-1})m_0}{2^n}\right).$$

Notranja vsota je odvisna od spremenljivk $m_0, m_1, \ldots, m_{n-2}$ in k_0, nič pa ni odvisna od $k_1, k_2, \ldots, k_{n-1}$. Lahko definiramo

$$y_1(k_0, m_{n-2}, m_{n-3}, \ldots, m_0) =$$
$$= \sum_{m_{n-1}=0}^{1} z(m_{n-1}, m_{n-2}, \ldots, m_1, m_0)\cdot\exp\left(-\frac{2\pi\imath k_0 2^{n-1} m_{n-1}}{2^n}\right)$$
$$= z(0, m_{n-2}, \ldots, m_1, m_0)\cdot 1 + z(1, m_{n-2}, \ldots, m_1, m_0)\cdot\exp\left(-\frac{2\pi\imath k_0 2^{n-1}}{2^n}\right).$$

Izračun y_1 zahteva le eno kompleksno množenje za vsako izmed 2^n možnih vrednosti parametrov $k_0, m_{n-2}, m_{n-1}, \ldots, m_1, m_0$. Za izračun vseh možnih vrednosti izraza y_1 potrebujemo torej 2^n kompleksnih množenj.

Na naslednjem koraku lahko sedaj izračunamo y_2 s formulo

$$y_2(k_0, k_1, m_{n-3}, \ldots, m_0) =$$
$$= \sum_{m_{n-1}=0}^{1} y_1(k_0, m_{n-2}, \ldots, m_1, m_0)\cdot\exp\left(-\frac{2\pi\imath(k_0 + k_1 2)2^{n-2} m_{n-2}}{2^n}\right).$$

Ta izračun prav tako potrebuje 2^n kompleksnih množenj. Naprej nadaljujemo podobno. Poglejmo si shemo, ki prikazuje kako izračunamo z:

$$\begin{aligned} z(m_{n-1}, m_{n-2}, \ldots, m_1, m_0) &\to y_1(k_0, m_{n-2}, \ldots, m_1, m_0)\\ y_1(k_0, m_{n-2}, m_{n-3}, \ldots, m_0) &\to y_2(k_0, k_1, m_{n-3}, \ldots, m_0)\\ &\vdots\\ y_{n-1}(k_0, k_1, k_2, \ldots, k_{n-2}, m_0) &\to y_n(k_0, k_1, k_2, \ldots, k_{n-2}, k_{n-1})\ . \end{aligned}$$

Kot smo rekli, na vsakem koraku potrebujemo 2^n kompleksnih množenj, imamo n korakov, skupno torej $n\cdot 2^n = N\log_2 N$ kompleksnih množenj. Poleg tega opazimo tudi, da po izračunu y_j vektorja y_{j-1} ne potrebujemo več. Izračune vektorjev lahko torej delamo na mestu, kar pomeni, da lahko na vsakem koraku prejšnje podatke zamenjamo z novimi trenutnimi podatki.

Prav tako lahko z $\frac{N}{2}\log_2 N$ kompleksnimi množenji izračunamo IDFT po formuli $\mathcal{F}^{-1}(w)(n) = \frac{1}{N}\mathcal{F}(w)(N-n)$.

S pomočjo DFT in IDFT lahko sedaj pospešimo tudi računanje konvolucije. Vemo namreč, da velja

$$z * w = \mathcal{F}^{-1}(\mathcal{F}(z)\mathcal{F}(w)).$$

Skupno porabimo za izračun konvolucije $N + \frac{3N}{2}\log_2 N$ kompleksnih množenj.

Dvodimenzionalna hitra diskretna Fourierova transformacija

V trditvi 1.1.18 smo definirali $y(n_1, m_2)$. Direkten izračun vseh vrednosti $y(n_1, m_2)$ zahteva $N_1 N_2^2$ kompleksnih množenj, za tem pa direkten izračun vseh vrednosti za $\mathcal{F}(z)(m_1, m_2)$ zahteva $N_1^2 N_2$ kompleksnih množenj. Skupaj torej potrebujemo $N_1 N_2 (N_1 + N_2)$ množenj.

Trditev 1.1.22 *Naj bosta N_1 in N_2 števili potence 2. Z uporabo računanja od zgoraj namesto direktnega računanja, se da $\mathcal{F}(z)$ izračunati z uporabo največ $\frac{1}{2} N_1 N_2 \log_2(N_1 N_2)$ kompleksnih množenj.*

1.2 Transformacije slik

1.2.1 Signali

Signali so funkcije ene ali več spremenljivk, ki nam posredujejo določene informacije o nekem fizikalnem pojavu, npr. podatke o zvočni frekvenci glasbila. Podatke, ki bi jih želeli dobiti iz vhodnega signala, največkrat ne dobimo direktno, temveč moramo vhodni signal pred njegovo analizo pretvoriti v nek drug sistem oz. zapis. Matematično povedano, na funkciji (matematični predstavitvi signala) moramo izvesti neko transformacijo. Predstavitev signala s funkcijo in transformacije, ki jih izvedemo na tej funkciji, so odvisne od lastnosti, ki jim preučevani signal ustreza (periodičnost, končnost, zveznost ...). Slika je signal, ki ga uvrščamo v skupini diskretnih in digitalnih signalov. Če imamo sivinsko sliko, potem je osnovna predstavitev signala $n \times m$ matrika, ki ima vrednosti elementov z intervala $[0, 1]$. Pri tem sta n višina in m širina slike, vrednosti elementov pa nastavimo na vrednosti pripadajočih slikovnih točk. Sliko bomo označili z I, njene slikovne točke pa z $I(x, y)$.

Transformacijo slike I bomo označili s T in jo definirali kot:

$$T : I(x, y) \to J(u, v).$$

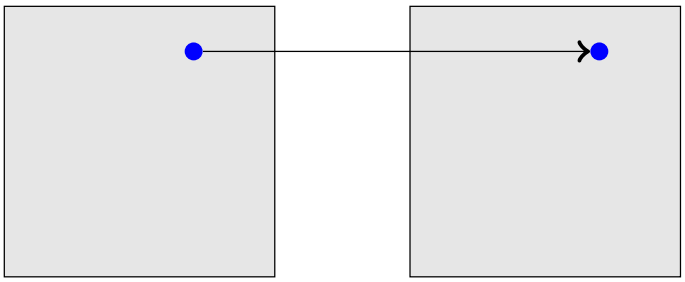
Obravnavali bomo tri skupine transformacij:

1. točkovne transformacije (izhodna vrednost transformacije je odvisna le od ene vrednosti; ni nujno, da uporabimo pri izračunu vrednosti;
2. lokalne transformacije (izhodna vrednost transformacije T je odvisna od vseh vrednosti v soseski pripadajoče slikovne točke vhodne slike);
3. globalne transformacije (izhodna vrednost transformacije T v točki (x, y) je odvisna od vseh vrednosti vhodne slike).

V nadaljevanju bomo spoznali posamezne skupine teh transformacij in njihovo praktično vrednost.

1.2.2 Točkovne transformacije

Točkovne transformacije pri izračunu vrednosti $T(x, y)$ uporabijo le podatek o vrednosti slikovne točke (x, y) (glej sliko 1.2). Za izračun točkovne transformacije T na $n \times m$ veliki sliki I porabimo $\mathcal{O}(k \cdot nm)$ časa, pri čemer je $O(k)$ časovna zahtevnost izračuna vrednosti $T(x, y)$ za eno slikovno točko. Ker za izračun potrebujemo le vrednost dotične slikovne



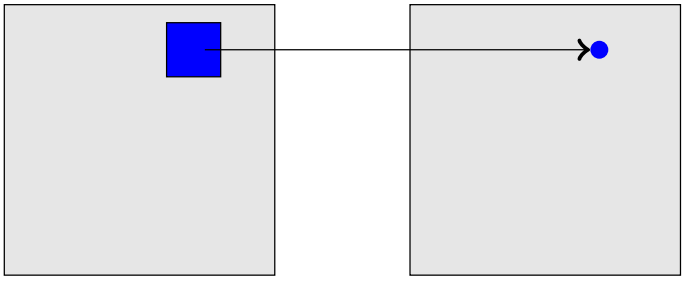
Slika 1.2: Točkovna transformacija vhodne slike.

točke, lahko izvedemo računanje na mestu, torej pri računanju ne potrebujemo dodatnega pomnilnika.

Enostavni primeri točkovnih transformacij so: rotacija slike, zrcaljenje, skrčitev, ... Za nas pomembne točkovne transformacije so tiste, ki spreminjajo barve itd.

1.2.3 Lokalne transformacije

Pri lokalnih transformacijah za izračun $T(x, y)$ v slikovni točki (x, y) uporabimo vrednosti slikovnih točk v njeni soseki (glej sliko 1.3).



Slika 1.3: Lokalna transformacija na sliki.

Na področju obdelave slik so tovrstne transformacije uveljavljen postopek, ki mu pravimo *filtriranje*. V tem kontekstu je $T(x, y)$ utežena vsota vrednosti slikovnih točk iz pravokotne okolice slikovne točke (x, y). Uteži shranimo v *filtrirno masko* h, ki je $(2b + 1) \times (2a + 1)$ matrika z realnimi vrednostmi. Običajno so filtrirne maske kvadratne in manjših velikosti. Vrednosti $T(x, y)$, ki jih dobimo pri filtriranju slike I s filtrom h, izračunamo s pomočjo konvolucije:

$$T(x,y) = \frac{(I * h)(x,y)}{\sum_{s=-a}^{a}\sum_{t=-b}^{b} h(s,t)} = \frac{\sum_{s=-a}^{a}\sum_{t=-b}^{b} h(s,t)I(x-s,y-t)}{\sum_{s=-a}^{a}\sum_{t=-b}^{b} h(s,t)}\,. \tag{1.11}$$

Opomba 1.2.1 V literaturi velikokrat poudarijo, da gre za filtriranje v slikovni domeni, saj poznamo tudi filtriranje v Fourierovi domeni, ki pa ga bomo mi spoznali malce kasneje.

Opomba 1.2.2 Kot vidimo v enačbi (1.11), smo rezultat, ki ga dobimo pri konvoluciji slike in filtrirne maske, normirali. Namesto tega bi lahko za filtrirno masko zahtevali, da

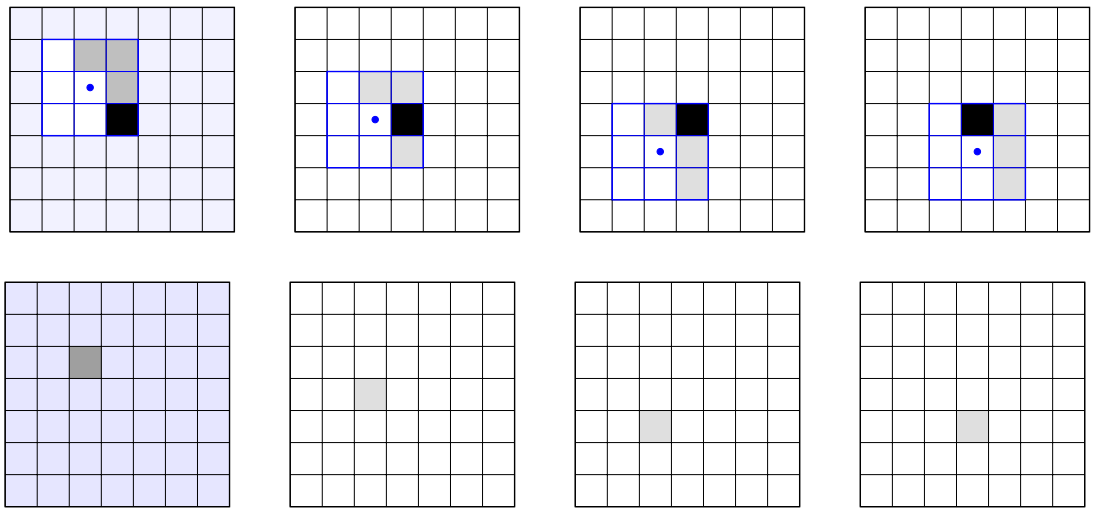
je normirana (tj. ima vsoto vrednosti 1). Filtrirne maske, ki jih bomo predstavili v našem delu, bomo zapisali v normirani obliki.

Postopek filtriranja obsega večji del področja pri obdelavi slik. Z uporabo filtrirnih mask lahko poiščemo, zgladimo ali izostrimo robove na sliki, zameglimo celotno sliko ... V nadaljevanju bomo predstavili primere filtrirnih mask, ki jih bomo kasneje tudi uporabljali v algoritmih za likovno upodabljanje slik.

Tehnični pogled na filtriranje

Za boljšo predstavo bomo sedaj filtriranje predstavili še grafično. Filtrirno masko h (glej **??**) najprej rotiramo za 180°, da dobimo filtrrino masko h'. Slednjo nato postavimo na vhodno sliko I tako, da center maske sovpada s slikovno točko na sliki I, v kateri želimo izračunati vrednost $T(x, y)$. Vrednost $T(x, y)$ sedaj izračunamo tako, da pomnožimo istoležne vrednosti v matriki h' in na sliki I (glej sliko **??**). Za razliko od točkovne transformacije, pri kateri smo novo vrednost lahko kar shranili na mesto (x, y) v matriki I, moramo naračunane vrednosti shraniti v novo matriko J.

V primeru, ko želimo izračunati npr. vrednost $T(x, y)$ za slikovno točko v kotu slike, del filtrirne maske ne pokriva slike I. Da se izognemu temu problemu, sliko I, pred filtriranjem s filtrirno masko velikosti $(2a + 1) \times (2a + 1)$, obrobimo naokoli z a celicami. Njihove vrednosti lahko nastavimo na več načinov (za primer glej sliko **??**). Pri tem opazimo, da lahko sedaj (ob privzetku, da filtrirane vrednosti računamo po vrsti) namesto, da za novo naračunane vrednosti ustvarimo novo matriko J, pri računanju vrednosti $T(x, y)$ novo vrednost shranimo na mesto $(x - a, y - 1)$, saj te vrednosti ne bomo več potrebovali pri izračunu ostalih vrednosti. S tem v primeru velikih slik naredimo velik prihranek pri pomnilniku.



Slika 1.4: Filtriranje v slikovni domeni.

Primer 1.2.3

Omenimo naj še, da so filtrirne maske, za razliko od maske v zgornjem primeru, običajno simetrične. Pri računanju filtriranih vrednosti lahko zato preskočimo rotacijo filtrirne

maske, saj dobimo nazaj isto filtrirno masko.

Zgoraj predstavljeni postopek je osnovni način filtriranja. V naslednjem podrazdelku, kjer bomo izpeljali filter za zameglitev slike, pa bomo spoznali še drug pristop k filtriranju.

Zamegljenje slike

Impulzna slika $\Delta_{p,q}$ je slika, ki ima vrednosti vseh slikovnih točk, razen ene z vrednostjo 1, enake 0. Definirana je kot:

$$\Delta_{p,q}(x-p, y-q) = \begin{cases} 1 & \text{če } (x,y) = (p,q), \\ 0 & \text{sicer.} \end{cases}$$

Konvolucijo slike I z w uteženo impulzno sliko $\Delta_{p,q}$ izračunamo kot:

$$[I * w\Delta_{p,q}](x,y) = w \cdot I(x-p, y-q) \ .$$

Postopek za izračun filtrirane slike:

1. Za vsako celico filtrirne maske h naredimo eno prazno matriko velikosti $(h+2a) \times (w+2a)$.

2. V vsako izmed teh praznih matrik kopiramo vhodno sliko I tako, da

Na sliki **??**

Primer 1.2.4

Filter za zaznavo robov na sliki

Kaj je rob ?

Ne maram ljudi.

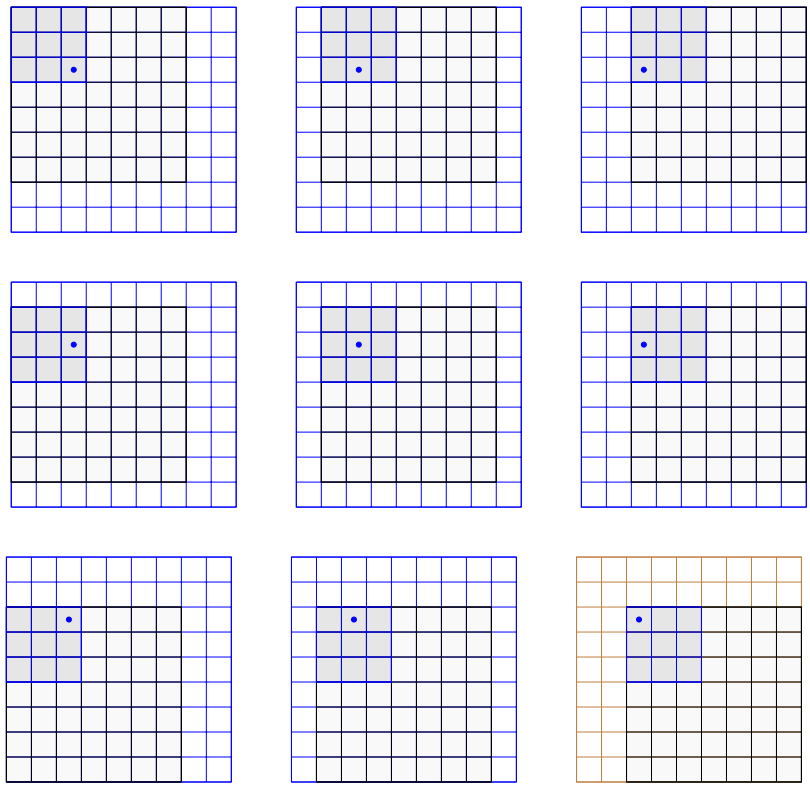
1.2.4 Globalne transformacije

Pri globalnih transformacijah za izračun vrednosti $T(x,y)$ upoštevamo vrednosti vseh slikovnih točk na sliki (glej sliko 1.8). Najbolj pomembna transformacija te vrste je za nas Fourierova transformacija. V tem podrazdelku si bomo pogledali kakšna je praktična vrednost Fourierove transformacije pri obdelavi slik.

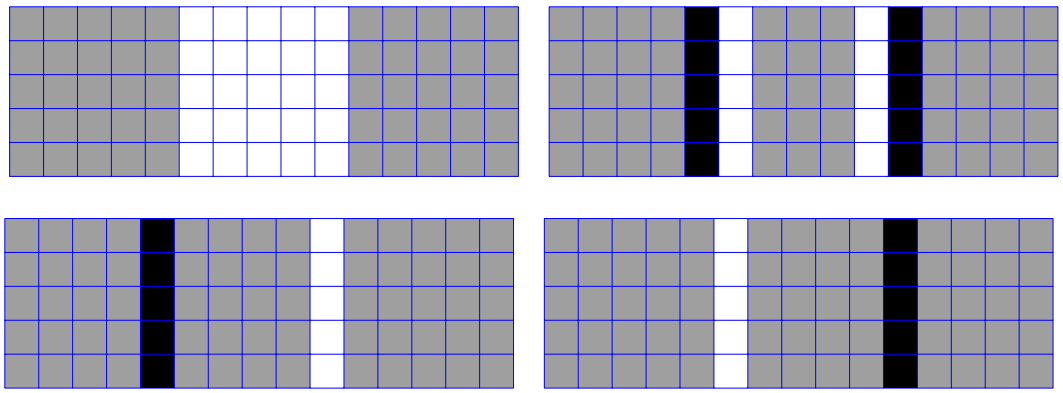
Kot smo videli v prvem razdelku tega poglavja lahko sliko I, namesto v običajni evklidski bazi, zapišemo v Fourierovi bazi:

$$I(x,y) = \sum_{v=0}^{N_1-1} \sum_{u=0}^{N_2-1} \mathcal{F}(I)(u,v) F_{u,v}(x,y) \ .$$

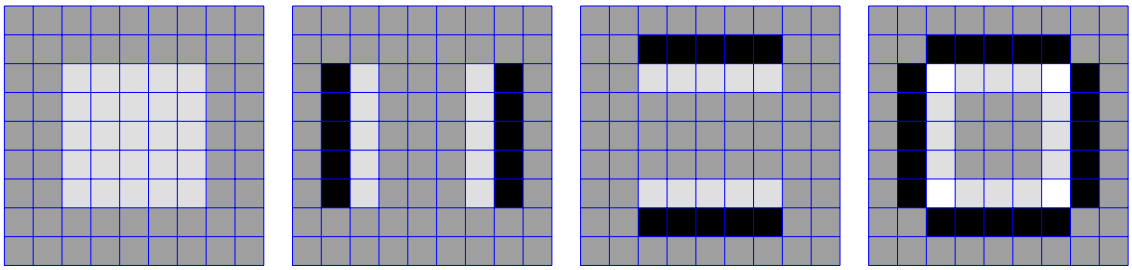
Za vizualno predstavitev slike bomo na Fourierovo transformacijo pogledali z nekoliko bolj fizikalnega vidika. Za začetek si poglejmo dvodimenzionalne sinusoide.



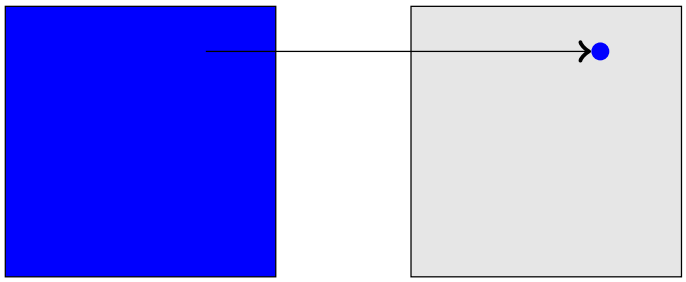
Slika 1.5: Filtriranje v slikovni domeni.



Slika 1.6: Robovi.



Slika 1.7: Robovi v kvadratu.



Slika 1.8: Globalna transformacija.

Dvodimenzionalne sinusoide

Zaradi poenostavitve zapisa naj velja $N_1 = N_2 = N$. Tedaj bomo zapisali $F_{u,v}(x,y)$ v polarni obliki:

$$F_{u,v}(x,y) = e^{\pm\imath\frac{2\pi\omega}{N}\cdot(u\sin\theta+v\cos\theta)},$$

kjer je $y = \omega\sin\theta$, $x = \omega\cos\theta$ in $\omega = \sqrt{x^2+y^2}$. Če pišemo še $\lambda = \frac{N}{\omega}$, dobimo

$$F_{u,v}(x,y) = \cos[\frac{2\pi}{\lambda}(u\sin\theta + v\cos\theta)] \pm \imath\sin[\frac{2\pi}{\lambda}(u\sin\theta + v\cos\theta)].$$

Tako realni del $F_{u,v}(x,y)$

$$\cos[\frac{2\pi}{\lambda}(u\sin\theta + v\cos\theta)]$$

kot imaginarni del

$$\pm\sin\left[\frac{2\pi}{\lambda}(u\sin\theta + v\cos\theta)\right]$$

sta sinusoidi z amplitudo 1, periodo λ in smerjo θ. Tedaj je $\frac{2\pi\omega}{N}$ radialna frekvenca, $\frac{\omega}{N}$ je frekvenca in $\lambda = \frac{N}{\omega}$ je dolžina vala sinusoide, merjena v slikovnih točkah.

Dvodimenzionalno sinusoido bomo zapisali s formulo

$$\frac{A}{2}\cdot\left(\cos\left[\frac{2\pi}{\lambda}\cdot(u\cdot\sin\Phi + v\cdot\cos\Phi) + \phi\right] + 1\right),$$

kjer je A amplituda sinusoide in ϕ je fazni zamik.

Definicija 1.2.5 Naj bosta Re in Im realni in kompleksni del vrednosti $\mathcal{F}(I)(u,v)$. Definiramo pojme:

1. *Fourierov spekter*:

$$|\mathcal{F}(I)(u,v)| = \sqrt{Re^2(u,v) + Im^2(u,v)};$$

2. *fazni zamik*:

$$\Phi(u,v) = \arctan\frac{Im(u,v)}{Re(u,v)};$$

3. *močnostni spekter*:

$$P(u,v) = Re^2(u,v) + Im^2(u,v).$$

Fourierovo transformacijo lahko z novo definiranimi pojmi zapišemo v polarni obliki:

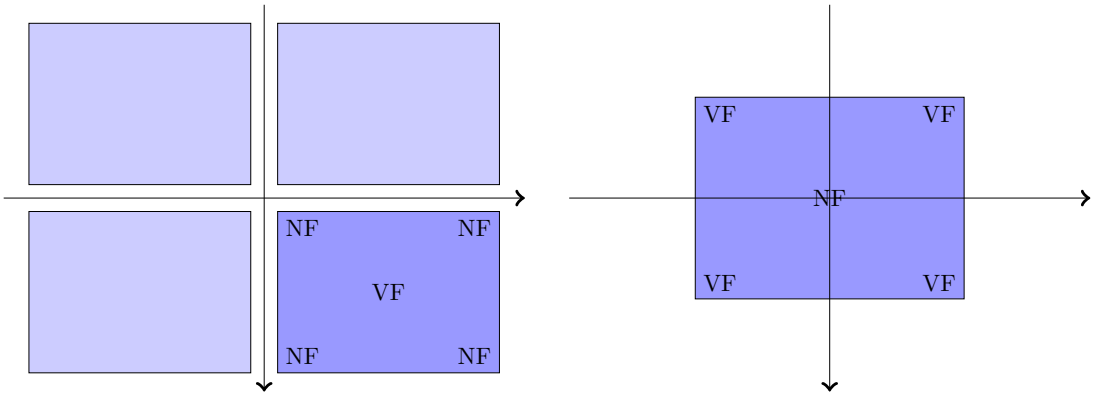
$$\mathcal{F}(z)(u,v) = |\mathcal{F}(z)(u,v)|\, e^{-\imath\cdot\Phi(u,v)}.$$

Poveš, da če je pikica v frekvenčni domeni je to sinus v navadni evklidski ravnini.

Primer 1.2.6 Primer slike in njenega Fouirerovega spektra. In primer ene slike, ki bo neka vsota, nekaj nekaj baznih elemntov.

Primer 1.2.7 Poglejmo si primer, ko imamo isti spekter in različni fazni zamik; ter isti fazni zamik, a različni spekter. Fazni zamik spreminjamo tako, da množimo z matriko R. Dobimo razne slike, ena izmed je lepa, ostale so kar nekaj ...

O spektru in faznem zamiku bomo govorili v nadaljevanju, saj sta to informaciji, ki nam povesta ogromno o sliki. Sedaj pa si najprej poglejmo še o translacijah in rotacijah Fourierove transformacije ...



Slika 1.9: Originalna DFT.

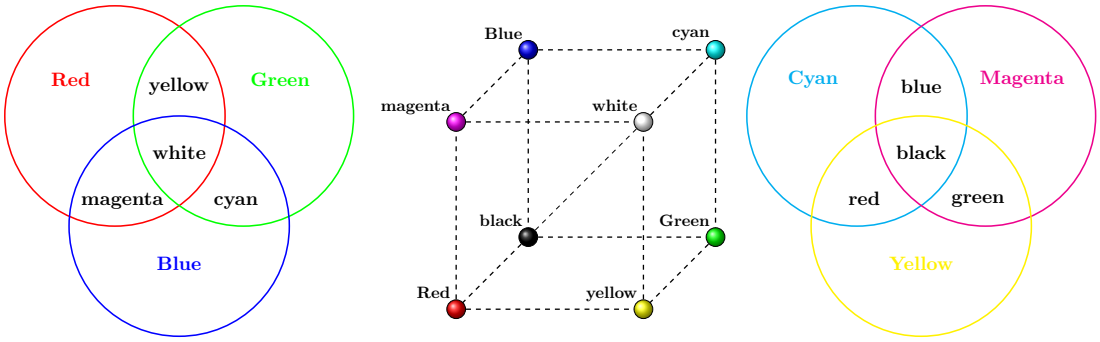
1.3 Barvni modeli

Poznamo različne barvne modele:

- RGB barvni model;
- CMY barvni model;
- HSV barvni model;
- HSL barvni model;
- YUV barvni model.

1.3.1 RGB in CMY barvna modela

RGB barvni prostor predstavimo z enotsko kocko. Oglišča kocke so enaka:



(a) (b) (c)

Slika 1.10: RGB kocka.

1.3.2 HSV in HSL barvna modela

To je pa polarna predstavitev RGB barvnega modela.

1.3.3 Kubelka–Monk barvni model

1.4 Odstranjevanje šuma na sliki

Odstranjevanje (digitalnega) šuma predstavlja pomembno področje pri obdelavi slik, saj ta običajno povzroča motnje in nepravilnosti pri delovanju algoritmov, ki jih uporabljamo za obdelavo slik. Pri tem naj opozorimo, da s pojmom šum mnogi poimenujejo tudi fine strukture in teksture, ki so prisotne na sliki. Slednje namreč tudi predstavljajo težave, vendar pa se moramo njim izogniti pri samem algoritmu za obdelavo slik.

V tem razdelku bomo predstavili merilo uspešnosti za odpravljanje digitalnega šuma na sliki, ki je predstavljen v članku [**?**]. Nadalje bomo primerjali različne metode, ki so predstavljene v tem članku, in nato metodo nelokalnega odstranjevanja šuma dopolnili z njeno pohitritvijo, opisano v članku [**?**]. V zadnjem delu bomo nato predstavili še implementacije posameznih metod in primerjali eksperimentalno dobljene podatke o učinkovitosti posameznih metod.

1.4.1 Metoda šuma

Naj bo I vhodna slika, na kateri je prisoten šum. Zaradi poenostavitve sintakse bomo slikovne točke, namesto z (x, y), označili z $i = (x_i, y_i)$. Vrednosti slikovnih točk na sliki I bomo označili z $v(i)$. Slednjo vrednost lahko zapišemo kot vsoto $v(i) = u(i) + n(i)$, pri čemer $u(i)$ predstavlja vrednost slikovne točke na sliki brez prisotnega šuma in $n(i)$ predstavlja vrednost prisotnega šuma. Z v bomo označili množico vrednosti $v = \{v(i) \mid i \in I\}$, ki jo poimenujemo *realna slika*. *Originalna slika* je množica vrednosti $u = \{u(i) \mid i \in I\}$. Množica $n = \{n(i) \mid i \in I\}$ pa je pristoni *šum* na sliki. Za modeliranje prisotnosti digitalnega šuma na sliki je najboljša izbira *Gaussov beli šum*, tj. Gaussove vrednosti s srednjo vrednostjo 0 in standardnim odklonom σ^2.

Definicija 1.4.1 *Operator šuma* D_h je tak operator, za katerega lahko naredimo dekom-

pozicijo realne slike v:

$$v = D_h v + n(D_h, v),$$

kjer je parameter h odvisen od standardnega odklona šuma.

Od operatorja šuma pričakujemo, da bo množica (slika) $D_h v$ gladkejša od originalne slike. Za dobljeni šum $n(D_h v)$ pa želimo, da čimbolje aproksimira beli šum. Nočemo pa, da bi poleg digitalnega šuma na sliki operator šuma odstranil tudi morebitno prisotno teksturo in pokvaril videz drobnih detajlov na sliki. Ker večina metod deluje na principu povprečenja sosednjih vrednosti slikovnih točk, pa zgornje zahteve povečini pri metodah ne držijo v celoti. Za merilo uspešnosti posameznih metod oz. operatorja šuma bomo uvedli t. i. *metodo šuma.*

Definicija 1.4.2 Naj bo u originalna slika in D_h operator šuma, ki je odvisen od parametra h. *Metoda šuma* je definirana kot razlika

$$u - D_h u.$$

Metoda šuma torej predstavlja razliko med originalno sliko u (na kateri ni nujno prisoten šum) in $D_h u$ (originalna slika, na kateri smo uporabili operator šuma). V primeru, ko bi originalna slika vsebovala neko teksturo, vzorce ali določeno strukturo, se te v metodi šuma ne smejo pojaviti, saj bi to kazalo na to, da smo pokvarili videz originalne slike. Metoda šuma mora izgledati kot aproksimacija šuma, celo v primerih, ko je na originalni sliki šum prisoten v zelo majhnih količinah.

1.4.2 Lokalni algoritmi za odstranjevanje šuma na sliki

Predstavili bomo tri metode za odstranjevanje šuma, ki pri računanju operatorja šuma izkoriščajo le lokalne podatke.

Gaussovo filtriranje

Gaussovo filtriranje sodi v skupino izotropnega linearnega filtriranja slik, kjer naredimo konvolucijo slike z linearnim simetričnim filtrom. Gaussov filter definiramo s funkcijo

$$G_h(x) := \frac{1}{4\pi h^2} e^{-\frac{|x|^2}{4h^2}} .$$

V tem primeru ima G_h standardni odklon h in velja naslednji izrek.

Izrek 1.4.3 *Metoda šuma za Gaussovo filtriranje je za dovolj majhen h enaka*

$$u - G_h * u = -h^2 \Delta u + o(h^2) .$$

Zgornji izrek dokažemo s pomočjo toplotne enačbe, saj izkoristimo dejstvo, da je fundamentalna rešitev toplotne enačbe porazdeljena po Gaussu (dokaz lahko bralec poišče v [**?**]).

Izrek nam pove, da bo v harmoničnih delih slike metoda šuma enaka 0, medtem ko bo v bližini linij, robov in delov tekstur imela visoke vrednosti. Na zveznih območjih slike bo metoda šuma torej optimalna, saj bo v celoti odstranila prisotni šum in pri tem ohranila videz slike. Deli slike v okolici linij, robov in prisotne teksture na sliki bodo po odstranitvi šuma postali zamegljeni in s tem pokvarili videz slike.

Anizotropno filtriranje

Ideje za to vrsto filtriranja so bile predstavljene v [**?**]. Pri anizotropnem filtriranju (kratica AF) poskušamo dejstvo, da pri Gaussovem filtriranju dobimo zamegljene dele slike, popraviti s tem, da v slikovni točki i naredimo konvolucijo s filtrom G_h le v smeri, ki je ortogonalna na $G(i)$. Pri tem je $G = \sqrt{(\partial_x u)^2 + (\partial_y u)^2}$, $\partial_x u$ in $\partial_y u$ sta gradientni sliki v x in y smeri. Konvolucijo naredimo torej le v "smeri linije, ki poteka skozi slikovno točko i."

Operator šuma za AF je za x, $G(x) \neq 0$, definiran s funkcijo

$$AF_h u(x) = \int G_h(t) \cdot u\left(x + t\frac{G(x)^\perp}{|G(x)|}\right) dt \, .$$

Ob predpostavki, da je originalna slika u dvakrat zvezno odvedljiva v x, lahko s pomočjo Taylorjeve vrste drugega reda pokažemo veljavnost naslednjega izreka (dokaz bomo tukaj izpustili).

Izrek 1.4.4 *Metoda šuma za anizotropni filter AF je na območjih, kjer velja $Du(x) \neq 0$, enaka*

$$u(x) - AF_h u(x) = -\frac{1}{2}h^2 \, |G| \cdot \kappa(u)(x) + o(h^2) \, .$$

V metodi šuma se pojavi ukrivljenost krivulje $\kappa(u)(x)$, saj je uspešnost te metode odvisna od ukrivljenosti krivulje, ki poteka skozi točko. V bližini slikovnih točk, kjer je linija, ki poteka skozi, lokalno ravna, je metoda šuma enaka 0. Posledično se ravne linije in robovi pri tej vrsti filtriranja šuma zelo lepo ohranijo. V okolici slikovnih točk, kjer imajo linije veliko ukrivljenost in so vrednosti gradienta velike, pa metoda šuma doseže velike vrednosti. Posledično na zveznih delih slike in v okolici linij, ki imajo veliko ukrivljenost, dobimo zamegljeno sliko.

Sosedsko filtriranje

Sosedsko filtriranje je skupno ime za skupino filtriranj, pri katerih novo vrednost slikovne točke i določimo na podlagi povprečja vrednosti tistih slikovnih točk iz njene soseske, ki imajo podobno vrednost.

Leta 1985 je Yaroslavsky v [**?**] predstavil filter te vrste. Operator šuma je za $x \in \Omega$ definiral s funkcijo

$$YNF_{h,\rho} u(x) \coloneqq \frac{1}{C(x)} \int_{B_\rho(x)} u(y) e^{-\frac{|u(y)-u(x)|^2}{h^2}} dy \, ,$$

kjer je normalizacijska konstanta definirana kot $C(x) = \int_{B_\rho(x)} e^{-\frac{|u(y)-u(x)|^2}{h^2}} dy$. Za sosesko slikovne točke x je izbral fiksno okolico $B_\rho(x)$, katere velikost je odvisna od parametra ρ. S filtrirnim parametrom h pa nadzira katere vrednosti slikovnih točk bomo vzeli v povprečje. Ker pri računanju povprečja uporabimo le slikovne točke s podobno vrednostjo, se na ta način izognemu zamegljenju linij in robov.

Deset let kasneje je bil predstavljen bolj prepoznavni filter t. i. SUSAN [**?**], pri katerem ne fiksiramo območja po katerem integriramo, temveč integrand pomnožimo z izrazom $e^{-\frac{|y-x|^2}{\rho^2}}$. V tem izrazu, podobno kot pri YNF, s parametrom ρ določimo soseko slikovnih

točk, ki jih bomo upoštevali pri računanju povprečja. Operator šuma je definiran s funkcijo

$$SNF_{h,\rho}u(x) \coloneqq \frac{1}{C(x)} \int_{\Omega} u(y) e^{-\frac{|y-x|^2}{\rho^2}} e^{-\frac{|u(y)-u(x)|^2}{h^2}} dy \ ,$$

kjer je normalizacijska konstanta definirana kot $C(x) = \int_{\Omega} e^{-\frac{|y-x|^2}{\rho^2}} e^{-\frac{|u(y)-u(x)|^2}{h^2}}$.

V praksi se operatorja šuma $YNF_{h,\rho}$ in $SNF_{h,\rho}$ obnašata podobno. Uporabljeni pristop za izbiro slikovnih točk, ki jih bomo upoštevali pri povprečju, v okolici linij in robov poskrbi, da pri računanju ne bomo hkrati vzeli v povprečje vrednosti, ki nastopijo na robu in v njegovi bližini. Na ta način se izognemu pojavu zamegljenosti robov po filtriranju. Vendar pa sosedski filter odpove v primeru, ko je v slikovnih točkah, ki jih primerjamo med seboj, prisoten šum. Tedaj namreč dobimo lažno območje slikovnih točk s podobno vrednostjo. Slednje vodi do umetnega pojava nepravilnosti na filtrirani sliki. Problem nastane zato, ker v integrandu primerjamo le vrednosti dveh slikovnih točk, nič pa ne upoštevamo geometrijske sestave njunih okolic. Na ta način ne zaznamo, da je v teh dveh slikovnih točkah prisoten šum, ki daje lažno prepričanje, da sta vrednosti podobni. Natančno opredeljen izrek, ki podpira trditve v tem odstavku, lahko bralec skupaj z dokazom najde v [**?**, strani 10–13].

1.4.3 Nelokalni algoritem za odstranjevanje šuma na sliki

Naj bo dana realna slika $v = \{v(i) \mid i \in I\}$. Za razliko od prejšnih metod, kjer smo vrednost za slikovno točko i naračunali le na podlagi vrednosti, ki se nahajajo v njeni bližini, bomo sedaj pri računanju uporabili vrednosti vseh slikovnih točk. S tem bomo sicer zelo povečali računsko zahtevnost algoritma, vendar bomo slednjo nato v praksi zmanjšali s premišljeno uporabo nekaterih podatkovnih struktur in omejitvijo območja, ki ga bomo uporabili pri računanju vrednosti v slikovni točki i.

Predstavitev postopka

Najprej bomo definirali vrednost $NL(v)(i)$, ki jo izračunamo kot uteženo povprečje vseh slikovnih točk:

$$NL(v)(i) \coloneqq \sum_{j \in I} w(i,j) v(j) \ ,$$

kjer je $\{w(i,j)\}_j$ množica uteži, ki zadošča pogojema $0 \leq w(i,j) \leq 1$ in $\sum_j w(i,j) = 1$.

Namesto, da bi primerjali le vrednosti slikovnih točk i in j, bomo utež $w(i,j)$ definirali kot merilo podobnosti okolic slikovnih točk i in j. Okolico slikovne točke k označimo z $\mathcal{N}_k$. Utež $w(i,j)$ bo odvisna od podobnosti vrednosti $v(\mathcal{N}_i)$ in $v(\mathcal{N}_j)$. Izračunali jo bomo z evklidsko razdaljo $\|v(\mathcal{N}_i) - v(\mathcal{N}_j)\|^2_{2,a}$, kjer je $a > 0$ standardni odklon Gaussovega jedra. Uporabo evklidske razdalje za izračun podobnosti utemeljimo z izračunom matematičnega upanja razdalje $\|v(\mathcal{N}_i) - v(\mathcal{N}_j)\|^2_{2,a}$, ki da rezultat

$$E[\|v(\mathcal{N}_i) - v(\mathcal{N}_j)\|^2_{2,a}] = \|u(\mathcal{N}_i) - u(\mathcal{N}_j)\|^2_{2,a} + 2\sigma^2.$$

Pri računanju podobnosti dveh območij s prisotnim šumom, bomo torej dobili enako stopnjo podobnosti, kot bi jo dobili v primeru, ko šum ni prisoten. S tem smo se izognili problemu, ki smo ga imeli pri sosedskem filtriranju.

Utež $w(i,j)$ izračunamo s formulo

$$w(i,j) = \frac{1}{Z(i)} e^{-\frac{\left\| v(\mathcal{N}_i) - v(\mathcal{N}_j) \right\|_{2,a}^2}{h^2}},$$

kjer je normalizacijska konstanta $Z(i)$ enaka

$$Z(i) = \sum_j e^{-\frac{\left\| v(\mathcal{N}_i) - v(\mathcal{N}_j) \right\|_{2,a}^2}{h^2}}.$$

Utež $w(i,j)$ bo torej poskrbela, da bomo pri izračunu nove vrednosti za slikovno točko i, povprečili le vrednosti tistih slikovnih točk, ki imajo podobno geometrijsko strukturo v okolici.

Primer 1.4.5

Pravilnost postopka

Če upoštevamo stacionarnost, potem algoritem NL-povprečenja konvergira k vrednosti pogojnega matematičnega upanja za piksel i. V tem primeru stacionarnost pomeni, da se z večanjem velikosti slike ohranijo podobna območja z detajli na sliki.

Naj bo V naključno polje in v njegova realizacija. Z Z označimo zaporedje $Z_i = \{X_i, Y_i\}$, kjer je $Y_i = V(i)$ realna vrednost in je $X_i = V(\mathcal{N}_i \setminus \{i\})$ vektor iz $\mathbb{R}^p$. NL-povprečenje vzamemo kot pogoj pri pogojnemu matematičnemu upanju, izračun je tak

$$r(i) = E[Y_i \mid X_i = v(\mathcal{N}_i \setminus \{i\})].$$

Izrek 1.4.6 *Naj bo $Z = \{V(i), V(\mathcal{N} \setminus \{i\})\}$ za $i = 1, 2 \ldots$ popolnoma stacionaren in mešan proces. Z NL_n označimo algoritem NL-povprečenja, ki ga uporabimo na zaporedju $Z_n = \{V(i), V(\mathcal{N}_i \setminus \{i\})\}_{i=1}^n$. Potem velja*

$$|N_n(j) - r(j)| \to 0$$

skoraj za vsak $j \in \{1, \ldots, n\}$.

Zgornji izrek nam pove, da algoritem NL-povprečenja popravi sliko s šumov in ne poskuša ločiti narazen šuma in originalne slike.

V tem primeru predpostavimo, da imamo aditiven bel šum. Naslednji rezultati nam povedo, da je pogojno matematično upanje funkcija $V(\mathcal{N}_i \setminus \{i\})$, ki minimizira povprečno kvadratno napako za pravo sliko (originalno).

Izrek 1.4.7 *Naj bodo V, U in N naključna polja, za katera velja, da je $V = U + N$, kjer je N neodvisno od signala prisoten bel šum. Tedaj veljata naslednji trditvi:*

1. *Za vsak $i \in I$ in $x \in \mathbb{R}^p$ velja $E[V(i) \mid X_i = x] = E[U(i) \mid X_i = x]$.*
2. *matematično upanje $E[U(i) \mid V(\mathcal{N}_i \setminus \{i\})]$ je funkcija v odvisnosti od $V(\mathcal{N}_i \setminus \{i\})$, ki minimizira kvadrat napake*

$$\min_g E[U(i) - g(V(\mathcal{N}_i \setminus \{i\}))]^2.$$

Časovna zahtevnost

Kljub temu, da nelokalni algoritem za odstranjevanje šuma, da zelo dobre rezultate in je robusten, pa je njegova uporaba zaradi visoke časovne zahtevnosti dokaj omejena.

Recimo, da je velikost slike n^2 in velikost okolice $\mathcal{N}_k$ enaka M^2 (M je konstanta). Časovna zahtevnost za izračun uteži $w(i,j)$ je enaka M^2. Izračun vrednosti $NL(v)(i)$ zahteva izračun n^2 uteži, kar pomeni, da je časovna zahtevnost izračuna ene vrednosti $NL(v)(i)$ enaka $\mathcal{O}(M^2 \cdot n^2)$. Skupna časovna zahtevnost je torej $\mathcal{O}(M^2 \cdot n^4)$, saj moramo izračunati vrednost $NL(v)(i)$ za vsako izmed n^2 slikovnih točk.

Prva izboljšava, ki jo naredimo v praksi za zmanjšanje časovne zahtevnosti, je omejitev območja pri računanju vrednosti $NL(v)(i)$. Namesto, da vsoto naredimo po vseh slikovnih točkah, jo naredimo le na omejeni okolici slikovne točke i. V članku izbrana velikost je 21×21. Skupaj z izbiro $M = 7$ dobimo izboljšano časovno zahtevnost $\mathcal{O}(49 \cdot 441 \cdot n^2)$. Pridobili smo kvadratno časovno zahtevnost, kar pa še vedno ni dobro za širšo uporabo nelokalnega filtra v praksi. Poleg tega omejitev območja prinese slabšo kakovost filtrirane slike.

Pohitritev algoritma

Wang je skupaj s sodelavci v članku [**?**] predstavil algoritem za nelokalno odstranjevanje šuma, ki v praksi porabi občutno manj časa za izračun kot originalni algoritem, pri čemer pa ohrani kakovost filtriranja. Kot smo videli, največ časa porabimo za izračun evklidske razdalje med dvema okolicama. Za to razdaljo bomo uvedli novo oznako $S(i,j)$:

$$S(i,j) = \|\mathcal{N}_i - \mathcal{N}_j\|^2 \tag{1.12}$$

$$= \sum_{l=0}^{M-1}\sum_{m=0}^{M-1} [v(\mathcal{N}_i)(l,m) - v(\mathcal{N}_j)(l,m)]^2 \, . \tag{1.13}$$

Če sliko I postavimo v koordinatni sistem in jo zrcalimo preko koordinatnega izhodišča (glej sliko **??**), lahko $\mathcal{N}_j(l,m)$ zapišemo kot $\mathcal{N}_j(l - x'_j, m - y'_j)$, kjer je $x'_j = \frac{3M}{2} + x_j$ in $y'_j = \frac{3M}{2} + y_j$. Enačba (1.12) se s tem prepiše v obliko:

$$\begin{aligned} S(i,j) &= \sum_{l=0}^{M-1}\sum_{m=0}^{M-1} [v(\mathcal{N}_i)(l,m) - v(\mathcal{N}_j)(l - x'_j, m - y'_j)]^2 \\ &= N_i^2 + N_j^2 - N_i * N_j \, , \end{aligned}$$

kjer so

$$N_i^2 = \sum_{l=0}^{M-1}\sum_{m=0}^{M-1} (v(\mathcal{N}_i)(l,m))^2,$$

$$N_j^2 = \sum_{l=0}^{M-1}\sum_{m=0}^{M-1} (v(\mathcal{N}_j)(l - x'_j, m - y'_j))^2,$$

$$N_i * N_j = 2 \cdot \sum_{l=0}^{M-1}\sum_{m=0}^{M-1} (v(\mathcal{N}_i)(l,m) \cdot v(\mathcal{N}_j)(l - x'_j, m - y'_j)) \, .$$

Vrednosti N_i^2 in N_j^2 lahko s pomočjo podatkovne strukture *vsote kvadratov* (*angl. Summed Square Image*), ki je predstavljena v 1.4.3, izračunamo brez uporabe množenj.

Vsota kvadratov

Za slikovno točko (x_0, y_0) naj bo

$$SSI[y_0, x_0] \coloneqq \sum_{x \leq x_0, y \leq y_0} v^2(y, x).$$

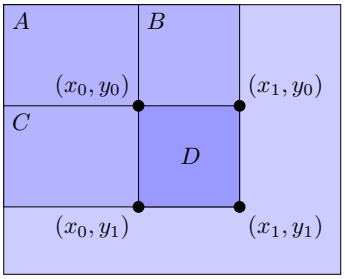
Algoritem 1 Izračun $n \times m$ matrike vsote kvadratov.

```
1: SSI ← Prazna n × m matrika.
2: SSI[0, 0] ← v²(0, 0)
3: for x₀ ∈ {1, ..., m − 1} do
4:     SSI[0, x₀] ← SSI[0, x₀ − 1] + v²(0, x₀)
5: end for
6: for y₀ ∈ {1, ..., n − 1} do
7:     SSI[y₀, 0] ← SSI[y₀ − 1, 0] + v²(y₀, 0)
8: end for
9: for x₀ ∈ {1, ..., m − 1} do
10:    for y₀ ∈ {1, ..., n − 1} do
11:        SSI[y₀, x₀] ← SSI[x₀ − 1, y₀] + SSI[x₀, y₀ − 1] − SSI[x₀ − 1, y₀ − 1] + v²(y₀, x₀)
12:    end for
13: end for
```

Zgornji algoritem ima časovno zahtevnost $O(nm)$, pri čemer je $n \times m$ velikost slike. Vsako slikovno točko po zgornjem algoritmu namreč obiščemo natanko enkrat.

Če nas sedaj zanima kolikšna je vsota kvadratov vrednosti slikovnih točk na pravokotnem imamo dano strukturo SSI, lahko vrednost poljubnega dela slike ali slikovne točke izračunamo v konstantnem času. Naj bo D območje za katerega želimo izračunati vsoto kvadratov vrednosti. Vrednost S_D določimo na naslednji način:

$$\begin{aligned} S_D &= S_{A \cup B \cup C \cup D} + S_A - S_{A \cup C} - S_{A \cup B} \\ &= SSI(x_1, y_1) + SSI(x_0, y_0) - SSI(x_0, y_1) - SSI(x_1, y_0). \end{aligned}$$



1.4.4 Implementacija in primeri

Poglavje 2

Likovno upodabljanje slik

Likovno upodabljanje slik (v nadaljevanju LUS) je področje v računalniški grafiki, ki se ukvarja z vprašanjem kako realistični medij slike pretvoriti v artistično likovno delo. V nasprotju z realističnim upodabljanjem, kjer je poudarek na podrobnostih, pri LUS poskušamo upodobiti imanentne informacije. LUS ima zelo široko uporabo pri upodabljanju slik, risb, tehničnih ilustracij, risanih animacij itd. V nasprotju s fotorealističnimi slikami imajo tako dobljena likovna dela namreč to prednost, da lahko likovni umetnik (ali tehnični risar) da poudarek na določen detajl slike, omeji količino podrobnosti, prilagodi paleto barv ali npr. doda sliki svojo osebno noto.

LUS se je razvilo iz računalniškega področja prepoznavanja (geometrijskih) oblik na sliki in digitalne obdelave slik. Z razvojem filtrov za obdelavo slik, ki so sprva služili bolj kot npr. orodje pri tehnični obdelavi medicinskih slik, se je sčasoma njihova uporaba razširila tudi v umetniške vode. Prišlo je do razvoja računalniškega poustvarjanja določenih likovnih stilov, kjer so izkoristili lastnosti in delovanje filtrov.

V prvem razdelku si bomo najprej pogledali primer uporabe filtrov za dosego nekega likovnega efekta na dani fotografiji. Ugotovili bomo, da uporaba filtrov, ki delujejo lokalno, ne pripelje do kompleksnih artističnih efektov. V preostalih razdelkih bomo zato spoznali različne načine za likovno upodabljanje slik, kjer se bomo poslužili modelov čopiča, svinčnika in voščenke ter med drugim uporabljali tudi optimizacijski pristop za slikanje in ustvarjanje mozaikov. V vsakem razdelku bomo navedli nekaj osnovnih člankov in njihovih povzetkov, na podlagi katerih si lahko bralec ustvari predstavo o bogatem dogajanju na področju LUS.

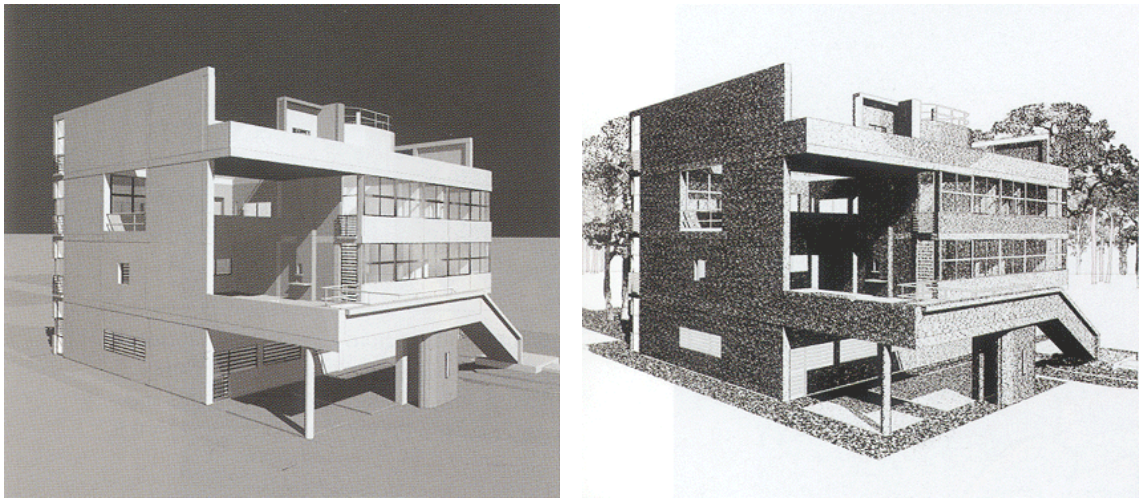
2.1 Artistični filter

Obstaja tudi vrsta komercialnih programov, ki ustvarjajo slike s pomočjo artističnih filtrov, npr. Adobe Photoshop.

2.2 Tehnične ilustracije

Tehnična ilustracija je

Leta 1991 sta Seligmann in Feiner v [26] opisala metodo za avtomatično konstruiranje 3D ilustracij. Ustvarjanje ilustracij poteka znotraj sistema stilističnih pravil in tehničnih zahtev, ki določajo končni videz ilustracije in jih lahko izbere uporabnik programa. Njun



Slika 2.1: Razlika med realističnim upodabljanjem in likovnim upodabljanjem.

sistem Njihov osnovni cilj je high-level of compositing the best model for comunicatting a particular intent. Yessios je opisal prototip sistema za upodabljanje materialov v arhitekturnih načrtih, kot so kamni, les, rože in tla. Namesto mehaničnega videza želijo ilustracijo prikazati z nežnejšimi potezami. Miyata [21] je opisal algoritem za upodabljanje vzorcev kamnitih sten. Appel je z soavtorji v [2] predstavil metodo, s katero narišemo črto, ki ima videz, da gre zadaj za drugo. Kamada in Kawai [14] sta posplošila nuno delo in ppredstavila različne parametre črte, kot so črtkanost, pikčastost, ..., s pomočjo katerih lahko skrite črte vizualno boljše predstavimo. Dooley in Cohen sta kasneje v [**?**, **?**] predstavila še več parametrov linij, kot je npr. debelina črte in podala diskusijo o načinu uporabnikove interakcije, da bi izboljašlai likovni vtis ilustracij, pri čemer bi obravnavali obliko objekta (linije, ki ga določajo) in njegovo senčenje.

Saito in Takahashi sta v [**?**] predstavila koncept *G*-bufferja za ustvarjanje razumljivih upodobitev 3D scen. Njihova metoda v bistvu po tem, ko ustvarijo množico *G*-bufferjev, uporablja tehnike iz procesiranja slik. V [**?**] sta Winkenbach in Salesin opisala tradicionalne pristope za upodabljanje slik na pen-and-ink način in pokazala, da lahko mnoge izmed njih implementiramo kot del avtomatskega likovnega upodabljanja slik. Predstavila sta stroke texture, s pomočjo katerega lahko tako teksturo kot tudi senčenje dosežemo z risanjem linij. Stroke texture dovoljuje tudi likovno upodabljanje slik, ki je odvisno od izbrane resolucije. Opisane tehnike sta demonstrirala s pomočjo kompleksnega arhitekturnega modela, ki vključuje tudi Lloyd's Wright's Robie House. Svoje delo sta v [31] nadgradila, kjer sta uvedla nove algoritme in tehnike za likovno upodabljanje parametrično podanih površin, ki imajo nepravilno obliko. Predstavila sta idejo hatchinga, kjer kontroliramo gostoto risanja linij, da bi dosegli želejeni ton senčenja, obliko in teksturo. Pokazala sta tudi kako lahko ravninski zemljevid, ki je glavna podatkovna struktura pri njunem upodabljanju ilustracij, konstruiramo iz parametrično podanih površin in ga uporabimo za rezanje linij in ustvarjanje linij, ki omejuje objekte. Za ukrivljene objekte sta predstavila tudi senčenje, ki posnema linijo ukrivljenega objekta, ki ga upodabljamo.

Lansdown in Schofield sta v [**?**] sta predstavila sistem Piranesi, ki uporablja tehnike likovnega upodabljanja za ustvarjanje ilustracij iz podanega 3D modela. Piranedsi upora-

blja standarne grafične plinovode, s pomočjo katerih ustvari 2D reliefno sliko. Uporabnik lahko nato posameznim območjim na sliki določi teksturo ali pa je ta določena avtomatsko.

Salisbury je skupaj s sodelavci v [25] predstavil interaktivni sistem za risanje ilustracij na podlagi črnobele vhodne slike, kjer linije na ilustraciji posnemajo obliko objektov na vhodni sliki. Uporabnik, preko novih tehnik za dodajanje vektorskega polja, določi orientacijo za vsako posamezno območje. Računalnik nariše linije, ki temeljijo na uporabnikovih primerih liniji in poskusi doseči iste tone na ilustracji kot so na vhodni sliki, s pomočjo novega algoritma, ki primerja zamegljeno verzijo trenutne ilustracije z vhodno sliko. S poravnanim vektorskim poljem z orientacijami površin objektov na sliki lahko uporabnik ustvari teksturo, ki ima videz, da je pritisnjena na površino, namesto da z njo le potemnimo neketere dele. To ima kot posledico bolj privlačne ilustracije, kot smo jih lahko poprej upodobili iz 2D slik.

Goodwin [27] s sodelavcama je predstavil pristop za določanje debeline črt v računalniško generiranih ilustracijah gladkih površin. Predpostavka, da se poudarjene črte rišejo na tistih območjih slike kjer je temnejše senčenje, jih je vodila do preproste formule za debelino linij, ki omejujejo objekte in tistih linij, ki sugestirajo detajle na objektih. Formula je odvisna od globine, ukrivljenosti in smeri svetlobe. Za upodabljanje linij uporabijo lokalno obliko in relacijo globine.

2.3 Risbe

Risbe so med osnovnimi likovnimii komunikacijami, ki človeku pomagajo v možganih reproducirati vhodno sliko risbe. Risanje s svinčnikom lahko razdelimo v dve skupini glede na vhodne podatke. te lahko namreč dobimo kot 2D informacijo ali pa kot 3D model. Glavna likovna stila, ki ju s svinčnikom poskušamo upodobiti sta skica ()in hatching (z risanjem na istem mestu v enakih ali različnih smereh poskušamo doseči določeni ton slike). Sousa in Buchanan [**?**] sta predstavili modele za grafit, papir za risanje, radirke. S temi modeli predstavi teksturo, ton in Modeli so narejeni na podlagi interakcije med grafitom svinčnika in papirja ter na podlagi lastnosti radiranja in smoofanja S pomočjo parametrov, kot je

Lee s sodelavci so v [**?**] iz podane 3D površine objekta izločili linije, ki objekt omejujejo in jih predelali za simulacijo ročno narisanih linij. Senčenje simulirajo s pomočjo teksture svinčnika in pa usmerjenih linij. Temelji na svetlobi in materijalu danega 3D modela. Pri upodabljanju slik s svinčnikom so algoritmi najpogosteje taki, da dobijo za vhodni podatek 3D model in nato poskušajo dobiti informacijo o linijah za risbo iz modela [**?**, **?**, **?**, **?**]. Rusinkiewicz je v [**?**] naredil pregled likovnega upodabljanja s svinčnikom na podlagi danega 3D modela.

V [**?**] je bil predstavljen model senčenja. Različne hatching stile sta definirala Hertzmann in Zorin v [**?**]. Praun pa je skupaj sodelavci [**?**] razvil algoritem za hatching, ki se izvaja v realnem času. V [**?**] je predstavljen algoritem za risanje potretov s svinčnikom s stilom, ki posnema določeno tradicijo ali pa stil risarjev.

Mao s sodelavci v [**?**] so predstavili postopek za lokalno strukturo orientacije na podlagi vhodne slike in vključili linearno integracijsko konvolucijo (LIC) z namenom simulacije efekta senčenja. Yamamoto pa je s sodelavci v [**?**] je razdelil vhodno sliko na plasti z različnimi toni v različnih obsegih. Li in Huang [**?**] sta uporabila geometrijske parametre, from ...

2.4 Painterly rendering

Cohen [3, 22] se je problema risanja oz. slikanja lotil z vidika umetne inteligence. Ustvaril je sistem Aaron, ki na podlagi množice randomiziranih pravil ustvari originalno kompozicijo v določenem artističnem stilu. Aaron ima na voljo tudi robotsko slikanje.

Haeberli je v [7] predstavil interaktivni sistem za slikanje, ki ustvari impresionistično sliko iz fotografije. Poteze čopiča je upodobil v vrstnem redu. Vsaka poteza je imela parametre (položaj, barva, velikost in oblika), ki so bili večinoma določeni interaktivno, tj. s pomočjo uporabnika. Shiraishi in Yamaguchi [19] določata orientacijo in obliko potez čopiča na podlagi lokalnih značilnosti vhodne slike. Curtis in ostali avtorji so v [4] predstavili semi-avtomatski algoritem za upodabljanje watercolor slik. Njihov algoritem ne bo ustvaril vidnih potez čopiča, zato s tem izgubijo artistični vtis slikanja s čopičem. Litwinowicz [17] je opisal avtomatičen sistem za postavitev potez čopiča na podlagi vektorskega polja izračunanega iz gradienta vhodne slike in uporabil rezanje potez na robovih slike. Izhodna slika algoritma je naslikana v impresionističnem stilu, za katerega so značilne kratke poteze. Sistem je razširil na likovno upodabljanje posnetkov. Za upodabljanje posnetkov je uporabil kratke poteze in premikal narisane poteze na platnu s pomočjo izračunanega optičnega toka, ustvarjenega na podlagi posnetka. Pristop, ki ga je Litwinowicz uporabil za postavljanje potez na platno (tj. postavitev potez na mrežo) je najbolj pogost način, ki ga uporabljajo za upodabljanje slik s čopičem. Razlike med algoritmi so v obliki in orientaciji potez. Treavett in Chen [28] sta predstavila način za likovno upodabljanje s statistično analizo vhodne slike, na podlagi katere določita položaj, orientacijo in velikost upodobljenih potez. Litwinowicz in Treavett s Chenom so sliko pobarvali enoplastno. Pri enoplastnem slikanju je potrebno natančno predviditi položaje in obliko potez, kar pa je težaven problem. Pomankljivost takih algoritmov je torej v tem, da ne moremo morebitnih detajlov in napakic, ki nastanejo pri slikanju ne moremo naknadno popraviti še z eno plastjo potez čopiča. Hertzmann je v [9] predlagal način upodabljanja slik z večplastnim slikanjem, kjer postavlja na platno poteze čopiča različnih debelin in oblik, ki jih določi s pomočjo Bezierovih krivulj. Začetno točko poteze ne postavi na mrežo slike, temveč v okolici mrežnih točk poišče točko z največjo napako. Kontrolne točke Bezierovih krivulj pa poišče na podlagi izračunanega gradienta vhodne slike. Tak način upodabljanja slikanja je tudi najbolj podobno dejanskemu načinu slikanja. Slikar namreč najprej naredi grobo sliko, ki jo nato z manjšimi čopiči popravi na delih, kjer je potrebno več poudarka in detajlov. S pomočjo te metode lahko ustvarimo sliko, ki ima artistični vtis, vendar pa je potratna in detajlov na sliki vedno ne nariše lepo. To metodo je Hertzmann skupaj s Perlinom razširil na upodabljanje posnetkov [1].

Ročno narisane poteze s čopičem nimajo konstantne barve vzdolž poteze, temveč imajo vgrajeno strukturo. Običajna metoda za upodobljanje potez čopiča je s pomočjo antialiazirane črte, ki ima konststno barvo. Litwinowicz je v [17] predstavil idejo dodajanja teksture in svetlobe pri upodabljanju potez. Hertzmann je v [11] predstavil posebno tehniko za dodajanje efekta svetljenja za likovno upodobljene slike s pomočjo višinskega polja slike in bump-mappinga. V [13] so Huang in sodelavci predstavili podoben način kot je bump-mapping, vendar je bistveno enostavnejši za implementacijo. Predstavili so model anizotropičnega čopiča. Upodobaljanje potez poteka tako, da pripravljeno masko čopiča (na podlagi ...)

Hays in Essa sta v [8] predstavila uniform likovno upodabljanje za slike in posnetke. Za razliko od večine prejšnjih algoritmov, so poteze pri tej metodi postavljene na večjem

platnu. Orientacije za poteze so izračunane z RBF interpolacijo. Pri upodabljanju posnetkov, se parametri potez med postopkom spreminjajo, orientacija potez pa je določena z vektorskim poljem na podlagi gradienta posameznih sličic, različne frekvence robov na sliki pa pripomorejo h upodobitvi detajlov na sliki.

Uporaba informacije o 3D geometrijski strukturi slike, ki jo želimo likovno upodobiti, nam dovoljuje veliko več fleksibilnosti pri ustvarjanju posnetkov. Meier iz Walt Disney Feature Animation je v [20] predstavila avtomatičen sistem, ki postavlja poteze čopiča na podlagi opisa posameznih delov slike. V filmih *Tarzan* [5] in *What Dreams May Come* so bile uporabljene razširitve tega algoritma. Tak sistem naravno zagotavlja časovno koherenco in ekonomičnost likovnega upodabljanja posnetka. Vendar pa podatki o geometrijski strukturi niso vedno dosegljivi. Klein [15] je objekte upodobil s pomočjo filtrirane strukture. Uporaba 3D strukture tipično proizvede drugačen srtistični učinek, saj so poteze čopiča vezane na geometrijsko struktuo objektov. Turk in Banks sta v [29] za ilustriranje vektorskega polja z uporabila relaksacijo. Njuno delo in uporabo relaksacije, ki jo je uporabil Haeberli v [7], je Hertzmann razširil v članku [10].

Lee je v [16] predstavil novo metodo za za določanje oprientacije potez čopiča v likovnem upodabljanju posnetkov. Informacijo o premikih, ki jo dobimo iz zaporedja sličič posnetka uporabimo za orientiranje potez čopiča v območjih, kjer je bilo zaznano večje gibanje. V območjih z rahlio zaznanim gibanjem pa določimo orientacijo potez čopiča na podlagi gradienta slike. Določanje orientacije po tem postopku ima prednost pri izražanju gibanja objektov.

Literatura

[1] K. Perlin A. Hertzmann, *Painterly Rendering for Video and Interaction*, NPAR '00 Proceedings of the 1st international symposium on Non-photorealistic animation and rendering, ACM New York, NY, USA, 2000, str. 7–12.

[2] A. Appel, F. J. Rohlf, A. J. Stein, *The haloed line effect for hidden line elimination*, SIGGRAPH '79 Proceedings of the 6th annual conference on Computer graphics and interactive techniques, ACM New York, NY, USA, 1979, str. 151–157.

[3] H. Cohen, *The Further Exploits of Aaron, Painter*, Stanford Humanities Review **4** (1995), 141–158.

[4] C. J. Curtis, S. E. Anderson, J. E. Seims, K. W. Fleischer, D. H. Salesin, *Computer-generated Watercolor*, SIGGRAPH '97 Proceedings of the 24th annual conference on Computer graphics and interactive techniques, ACM Press/Addison-Wesley Publishing Co. New York, NY, USA, 1997, str. 421–430.

[5] E. Daniels, *Deep canvas in Disney's Tarzan*, SIGGRAPH '99 ACM SIGGRAPH 99 Conference abstracts and applications, ACM New York, NY, USA, 1999, str. 200–200.

[6] M. W. Frazier, *An Introduction to Wawelets Through Linear Algebra*, Springer–Verlag New York, New York, USA, 1999.

[7] P. Haeberli, *Paint By Numbers: Abstract Image Representations*, SIGGRAPH '90 Proceedings of the 17th annual conference on Computer graphics and interactive techniques, ACM New York, NY, USA, 1990, str. 207–214.

[8] J. Hays, I. Essa, *Image and Video Based Painterly Animation*, NPAR '04 Proceedings of the 3rd international symposium on Non-photorealistic animation and rendering, ACM New York, NY, USA, 2004, str. 113–120.

[9] A. Hertzmann, *Painterly Rendering with Curved Brush Strokes of Multiple Sizes*, SIGGRAPH '98 Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, ACM New York, NY, USA, 1998, str. 453–460.

[10] A. Hertzmann, *Paint By Relaxation*, Tech. report, New York University, NY, USA, 2000.

[11] A. Hertzmann, *Fast Paint Texture*, NPAR '02 Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering, ACM New York, NY, USA, 2002, str. 91–ff.

[12] A. Hertzmann, *Non-Photorealistic Rendering and the Science of Art*, NPAR '10 Proceedings of the 8th International Symposium on Non-Photorealistic Animation and Rendering, ACM New York, NY, USA, 2012, str. 147–157.

[13] H. Huang, T. N. Fu, C. F. Li, *Painterly Rendering with Content-dependent Natural Paint Strokes*, The Visual Computer: International Journal of Computer Graphics **27** (2011), 861–871.

[14] T. Kamada, S. Kawai, *An enhanced treatment of hidden lines*, ACM Transactions on Graphics (TOG) **6** (1987), 308–323.

[15] A. Klein, M. M. Kazhdan W. Li, W. T. Corrêa, A. Finkelstein, T. A. Funkhouser, *Non-photorealistic Virtual Environments*, SIGGRAPH '00 Proceedings of the 27th annual conference on Computer graphics and interactive techniques, ACM Press/Addison-Wesley Publishing Co. New York, NY, USA, 2000, str. 527–534.

[16] H. Lee, C. H. Lee, K. Yoon, *Motion Based Painterly Rendering*, EGSR'09 Proceedings of the Twentieth Eurographics conference on Rendering, Eurographics Association Aire-la-Ville, Switzerland, Switzerland, 2009, str. 1207–1215.

[17] P. Litwinowicz, *Processing Images and Video For an Impressionist Effect*, SIGGRAPH '97 Proceedings of the 24th annual conference on Computer graphics and interactive techniques, ACM Press/Addison-Wesley Publishing Co. New York, NY, USA, 1997, str. 407–414.

[18] C. Lu, L. Xu, J. Jia, *Combining Sketch and Tone for Pencil Drawing Production*, NPAR '12 Proceedings of the Symposium on Non-Photorealistic Animation and Rendering, Eurographics Association Aire-la-Ville, Switzerland, 2012, str. 65–73.

[19] Y. Yamaguchi M. Shiraishi, *An Algorithm for Automatic Painterly Rendering Based on Local Source Image Approximation*, NPAR '00 Proceedings of the 1st international symposium on Non-photorealistic animation and rendering, ACM New York, NY, USA, 2000, str. 53–58.

[20] B. J. Meier, *Painterly Rendering for Animation*, SIGGRAPH '96 Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, ACM New York, NY, USA, 1996, str. 477–484.

[21] K. Miyata, *A method of generating stone wall patterns*, SIGGRAPH '90 Proceedings of the 17th annual conference on Computer graphics and interactive techniques, ACM New York, NY, USA, 1990, str. 387–394.

[22] M. L. Morbey, *From Canvas to Computer: Harold Cohen's Artificial Intelligence Paradigm for Art Making*, doktorska disertacija, The Ohio State University, Ohio, USA, 1992.

[23] D. Rudolf, D. Mould, E. Neufeld, *Simulating Wax Crayons*, PG '03 Proceedings of the 11th Pacific Conference on Computer Graphics and Applications, IEEE Computer Society Washington, DC, USA, 2003, str. 163–172.

[24] D. Rudolf, D. Mould, E. Neufeld, *A Bidirectional Deposition Model of Wax Crayons*, Computer Graphics Forum **24** (2005), 27–39.

[25] M. P. Salisbury, M. T. Wong, J. F. Hughes, D. H. Salesin, *Orinetable Textures for Image-Based Pen-and-Ink Illustration*, SIGGRAPH '97 Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques, ACM Press/Addison-Wesley Publishing Co. New York, NY, USA, 1997, str. 401–406.

[26] D. D. Seligmann, S. Feiner, *Automated Generation of Intent-based 3D Illustrations*, SIGGRAPH '91 Proceedings of the 18th annual conference on Computer graphics and interactive techniques, ACM New York, NY, USA, 1991, str. 123–132.

[27] A. Hertzmann T. Goodwin, I. Vollick, *Isophote Distance: A Shading Approach to Artistic Stroke Thickness*, NPAR '07 Proceedings of the 5th international symposium on Non-photorealistic animation and rendering, ACM New York, NY, USA, 2007, str. 53–62.

[28] S. M. F. Treavett, M. Chen, *Statistical Techniques for the Automated Synthesis of Non-photorealistic Images*, Proc. 15th Eurographics UK Conference, 1997.

[29] G. Turk, D. Banks, *Image-guided Streamline Placement*, SIGGRAPH '96 Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, ACM New York, NY, USA, 1996, str. 453–460.

[30] G. Winkenbach, D. H. Salesin, *Computer-Generated Pen-and-Ink Illustrations*, SIGGRAPH '94 Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques, ACM New York, NY, USA, 1994, str. 91–100.

[31] G. Winkenbach, D. H. Salesin, *Rendering Parametric Surfaces in Pen and Ink*, SIGGRAPH '96 Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques, ACM New York, NY, USA, 1996, str. 469–476.