SIMULACE OBRAZOVÉHO KODÉRU NA BÁZI 3D KLT

Lukáš Fritsch

ČVUT v Praze, Fakulta elektrotechnická, katedra radioelektroniky

Abstrakt

Obrazové kompresní algoritmy založené na Karhunen- Loeveho transformaci (KLT) konstruující matici dat vstupujících do kodéru na základě jednoho snímku nemohou z principu redukovat časovou redundanci mezi snímky v sekvenci, která byla pořízena např. bezpečnostní kamerou. Ovšem redukce časové redundance může (souvisí s charakterem snímané scény) významným způsobem přispět ke snížení výstupního bitového toku. V tomto příspěvku je prezentována základní verze obrazového kodéru na bázi KLT, jehož vstupem jsou data zkonstruovaná z několika po sobě jdoucích snímků, a proto obsahují informaci o časovém vývoji scény. Kritickým místem kodéru je blok pro výběr vektorů, které budou tvořit jádro KLT. Vhodné kritérium pro jejich stanovení může vyplynout z aplikace, ve které má být kodér nasazen. Např. v případě snímků zachycených bezpečnostní kamerou může být kritériem schopnost rozpoznat pohyb ve scéně, schopnost rozpoznat osobu podle jejího obličeje nebo schopnost určit počet zájmových objektů ve scéně. V příspěvku jsou diskutovány i další přístupy, které mohou snížit výpočetní náročnost základní verze obrazového kodéru.

Vývoj popisovaného obrazového kodéru je realizován v programovém prostředí MATLAB, jehož nástroje práci velmi usnadňují.

1 Úvod

Karhunen- Loeveho transformace (KLT) tvoří významný matematický nástroj využitelný v širokém spektru aplikací. Velice významné je využití této transformace pro zpracování obrazové informace. Typickou úlohou je komprese obrazu, kdy lze dosáhnout relativně vysokých stupňů redukce obrazových dat. Nekorelovanost transformovaných dat (hlavních komponent) předurčuje použití KLT v oblasti rozpoznávání obrazu (lidské tváře, otisky prstů,…) ke vzoru uloženém v trénovací sadě. Nevýhodou KLT je její vysoká výpočetní náročnost (paměťová i časová). Jádro transformace se totiž odvíjí od vstupních dat na základě numericky vypočtených (Jacobiova metoda, Householderova metoda, QL algoritmus aj.) vlastních čísel a vlastních vektorů příslušné kovarianční matice. Obě posledně zmiňované aplikace jsou zpracovány v [1], [2]. Pramen [1] podrobně popisuje problematiku rozpoznávání, pramen [2] se zabývá kompresí statických snímků.

V tomto příspěvku je popsána úloha komprese sekvence snímků založená na KLT. Cílem je zanést do vstupních dat časový vývoj obrazové scény a tím umožnit redukci časové redundance. V textu je nejprve popsáno matematické pozadí úlohy komprese snímků, [1], [3], [4]. Následuje podrobnější výklad prezentovaného obrazového kodéru. Poslední část příspěvku se zabývá experimentálními výsledky, které přinesly simulace kodéru v Matlabu- je diskutována kvalita komprimovaných snímků, možnost snížení výpočetní náročnosti a vliv komprimovaných snímků na úspěšnost v procesu rozpoznávání.

2 Princip redukce dimenze vstupních dat

Cílem KLT je snížit dimenzi prostoru vstupních dat na základě nalezení tzv. hlavních komponent (odtud též jiný název pro KLT- PCA (Principal Component Analysis)). Hlavní komponenty jsou tvořeny výběrovým rozptylem a vektorem koeficientů. Požadavkem je, aby vektory koeficientů hlavních komponent byly vzájemně nekorelované a dále, aby hlavní komponenty co nejlépe popisovaly rozptyl vstupních dat. Výběrový rozptyl je roven vlastnímu číslu a vektor koeficientů je roven vlastnímu vektoru, který přísluší danému vlastnímu číslu, kovarianční matice zkonstruované ze vstupních dat. První hlavní komponenta přitom popisuje největší podíl rozptylu, pro další hlavní komponenty tento podíl postupně klesá. Jak bylo napsáno výše, vlastní vektory jsou vzájemně nekorelované a tvoří transformační matici (jádro KLT), pomocí níž můžeme transformovat vstupní data do hlavních komponent (někdy též do eigenspace). Důležitý je zde pro nás právě krok stanovení jádra transformace. Zájmem je použít pouze ty hlavní komponenty, jejichž vlastní čísla jsou větší než daný práh a tím dostatečně pokrýt popis rozptylu vstupních dat. Tento krok se odvíjí od

aplikace, kde má být KLT nasazena. Jedním kritériem pro výběr relevantních hlavních komponent může být tzv. Kaiserovo pravidlo, které doporučuje neuvažovat ty hlavní komponenty, u nichž je vlastní číslo menší než průměrná hodnota ze všech vlastních čísel.

3 Popis funkce prezentovaného obrazového kodéru a jeho modifikace

Jak bylo uvedeno výše, tento kodér vytváří množinu vstupních obrazových dat na základě několika po sobě jdoucích snímků z dané sekvence, aby bylo možné zachytit v těchto datech časový vývoj scény. Popišme si zjednodušeně základní verzi kodéru, z níž později vyplynou její modifikace. Uvažujme *N* snímků, každý o rozměrech *HxW*. Matici $\mathbf{A}$ vstupních dat, z nichž v dalších krocích sestrojíme kovarianční matici $\mathbf{C}$, sestavíme tak, že do jejích řádků postupně vkládáme jasové úrovně (uvažujeme snímky ve stupních šedé) vždy na stejných pixelových pozicích od každého z *N* snímků. Takže první řádek matice $\mathbf{A}$ obsahuje úrovně pro pixelovou pozici (1,1) ze všech *N* snímků, druhý řádek pak sdružuje úrovně pro pixelovou pozici (1,2), atd. Zřejmě má tato matice *HxW* řádků a *N* sloupců, kde *N* představuje dimenzi vstupních dat. Při této interpretaci vstupních dat si můžeme představit každý řádek matice $\mathbf{A}$ jako samostatný snímek, označme jej p_i, který nám podává informace o časovém vývoji daného pixelu v původním snímku. Naším úkolem je tedy nalézt množinu hlavních komponent, které nám popíší dostatečně věrohodně (odvíjí se od aplikace, kde má být kodér nasazen) rozptyl původních dat. Máme- li sestavenou matici $\mathbf{A}$, můžeme přistoupit k dalšímu kroku, kdy od každého řádku matice $\mathbf{A}$ odečteme průměrný řádek $\overline{p}$ (první hodnota v $\overline{p}$ je rovna průměrné hodnotě všech dat v první dimenzi, druhá pak průměrné hodnotě všech dat ve druhé dimenzi atd.)

$$\mathrm{A}_i = p_i - \overline{p}, i = 1 \ldots HxW \tag{1}$$

kde i představuje *i*-tý řádek matice $\mathbf{A}$.

Následuje výpočet kovarianční matice $\mathbf{C}$

$$\mathbf{C} = \mathbf{A}^{\mathbf{T}} \cdot \mathbf{A}. \tag{2}$$

Další krok spočívá v numerickém výpočtu vlastních čísel a k nim příslušejících vlastních vektorů kovarianční matice $\mathbf{C}$, neboť tato dvojice reprezentuje hlavní komponenty ve smyslu uvedeném v předchozím odstavci. Vlastní vektory tvoří transformační matici $\mathbf{K}$ KLT, k jejíž konstrukci použijeme pouze relevantní hlavní komponenty (vlastní vektory jsou v této matici uspořádány v řádcích). Směrodatným ukazatelem pro posouzení relevantnosti jsou velikosti vlastních čísel. Jak ovšem bylo napsáno výše, nastavení prahové hodnoty pro vlastní čísla není snadnou záležitostí. Vychází se z toho, pro jakou aplikaci bude kodér použit a lze tedy říci, že rozhodující pro stanovení prahu jsou praktické zkušenosti získané simulacemi apod. Míra úspěšnosti použité metody je závislá na statistických vlastnostech zpracovávaných dat.

Máme-li stanovenou transformační matici KLT, můžeme přistoupit k transformaci matice vstupních dat $\mathbf{A}$ do eigenspace

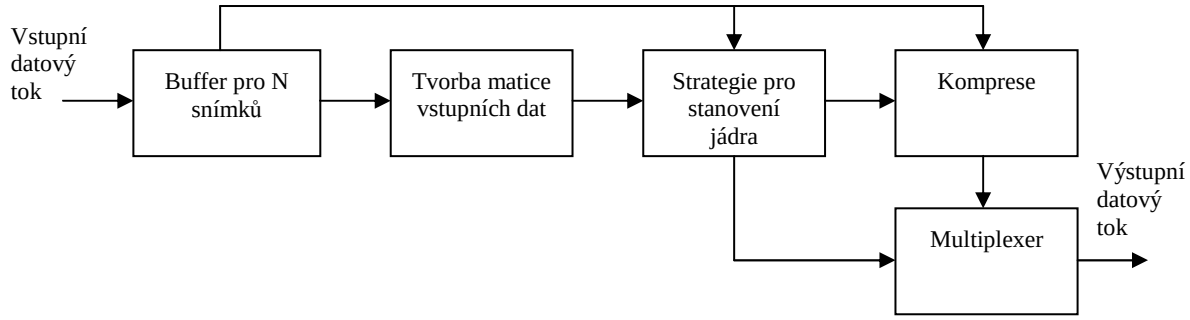
$$\mathbf{T} = \mathbf{K} \cdot \mathbf{A}^{\mathbf{T}}. \tag{3}$$

Získáme matici, kde data mají menší dimenzi než v matici $\mathbf{A}$ (závisí na počtu použitých hlavních komponent v transformační matici). Jednotlivé dimenze jsou v matici $\mathbf{T}$ nyní uspořádány v řádcích. Po tomto kroku dochází k nevratné ztrátě dat (po zpětné transformaci nezískáme data shodná s daty původními). Použijeme-li samozřejmě všechny hlavní komponenty ke konstrukci transformační matice, získáme po zpětné transformaci původní data (až na případné odchylky způsobené konečnou přesností při počítání s reálnými čísly). Zpětnou transformaci lze popsat vztahem

$$\mathbf{A}^{\mathbf{T}} = \mathbf{K}^{\mathbf{T}} \cdot \mathbf{T}. \tag{4}$$

Na závěr přičteme k rekonstruovaným datům vypočteným podle vztahu (4) v *N* dimenzích průměrný vektor $\overline{p}$ délky *N*, který jsme vypočetli na začátku této úlohy.

Na obr. 1 je znázorněno zjednodušené blokové schéma výše popisované základní verze obrazového kodéru.



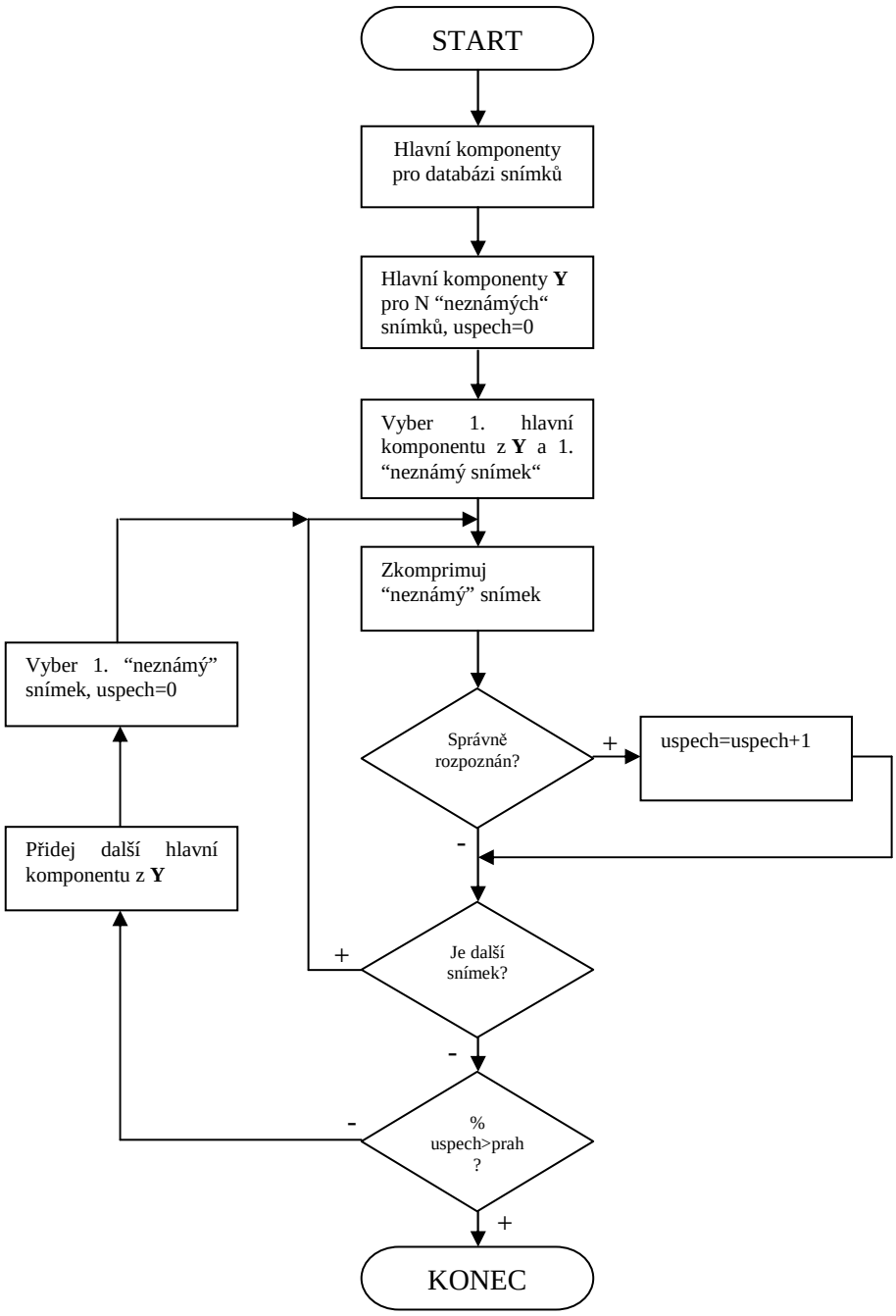
Obr. 1: Zjednodušené blokové schéma základní verze kodéru

Zabývejme se nyní podrobněji možnostmi řešení bloku *Strategie pro stanovení jádra* uvedeného na obr. 1. V našem případě jde o aplikaci, kdy chceme komprimovaný snímek podrobit případnému procesu rozpoznávání lidské tváře na základě výběru nejpodobnější tváře z databáze. Rozpoznávání je založeno na KLT, kdy využíváme nulové korelace mezi hlavními komponentami stanovenými pro snímky uložené v databázi. Proces rozpoznávání je podrobněji popsán v [1]. Požadavkem může být určitá procentuální úspěšnost rozpoznávání reprezentativních tváří, u kterých předem známe nejpodobnější snímky z databáze. V prvním kroku se uvažuje např. pouze první hlavní komponenta reprezentativních tváří ("neznámých" tváří), komprimovaný snímek vstoupí do procesu rozpoznávání a zjistí se, zda byl nalezen očekávaný nejpodobnější snímek. Toto se provede pro všechny reprezentativní tváře a vyčíslí se procentuální úspěšnost rozpoznání. Na základě tohoto výsledku se rozhodne, zda se přidá nejbližší méně významná hlavní komponenta a celý cyklus rozpoznávání se zopakuje nebo se pro proces komprese použije takový počet hlavních komponent, který splňuje stanovený požadavek na procentuální úspěšnost rozpoznávání. Jde vlastně o trénovací fázi obrazového kodéru, kdy si kodér zjistí optimální volbu hlavních komponent pro kompresi sady vstupních snímků. Popsaný proces natrénování je uveden na obr. 2 na následující straně.

Zabývejme se nyní možnostmi, které mohou vést ke snížení výpočetní náročnosti výše popisované základní softwarové verze kodéru, kde jsou zřejmé poměrně vysoké paměťové nároky na matice pro správu obrazových dat. Jednou z jednodušších možností může být verze, kdy každý člen vstupní matice neodpovídá vždy právě jednomu pixelu, ale je tvořen hodnotou určenou váhováním pixelů z dané oblasti obrazu. V našem případě byly simulovány možnosti, kdy oblast tvořilo $m.n$ pixelů ($m=2, 4, 8$ a $n=2, 4, 6$), přičemž hodnota reprezentující tuto oblast byla stanovena na základě průměrné, minimální, maximální a náhodně vybrané úrovně z oblasti. Na takto ohodnocenou oblast pak v dalších výpočtech pohlížíme, jako kdyby šlo o jeden pixel. Zřejmě se tedy sníží počet řádků matice vstupních dat zkonstruované podle výše uvedených postupů z hodnoty HxW na hodnotu $HxW/(m.n)$. Jistou nevýhodou je ovšem zhoršení kvality komprimovaných snímků, v obrazu je totiž zřejmá bloková struktura způsobená náhradou $m.n$ pixelů pixelem jedním. Pro snížení viditelnosti blokové struktury v komprimovaném obrazu byla provedena filtrace obrazu v prostorové oblasti filtrem typu dolní propust.

Další možnost vedoucí ke snížení výpočetní náročnosti je založena na rozdělení původní sekvence o N vstupních snímcích do dvou sekvencí o $N/2$ snímcích. Transformační matice $\mathbf{K}$ se vypočítá pouze pro první z takto vytvořených sekvencí. Druhá sekvence použije ke kompresi transformační matici $\mathbf{K}$ vypočtenou pro první sekvenci.

Následující odstavec pojednává o experimentálních výsledcích získaných simulacemi výše diskutovaných verzí obrazového kodéru. Simulace byly realizovány v programovém prostředí Matlab. Záležitosti týkající se problematiky rozpoznávání pro potřeby natrénování obrazového kodéru byly simulovány v aplikaci podrobněji prezentované v [1].



Obr. 2 Vývojový diagram trénovací fáze obrazového kodéru pro zjištění optimálního počtu hlavních komponent v procesu komprese

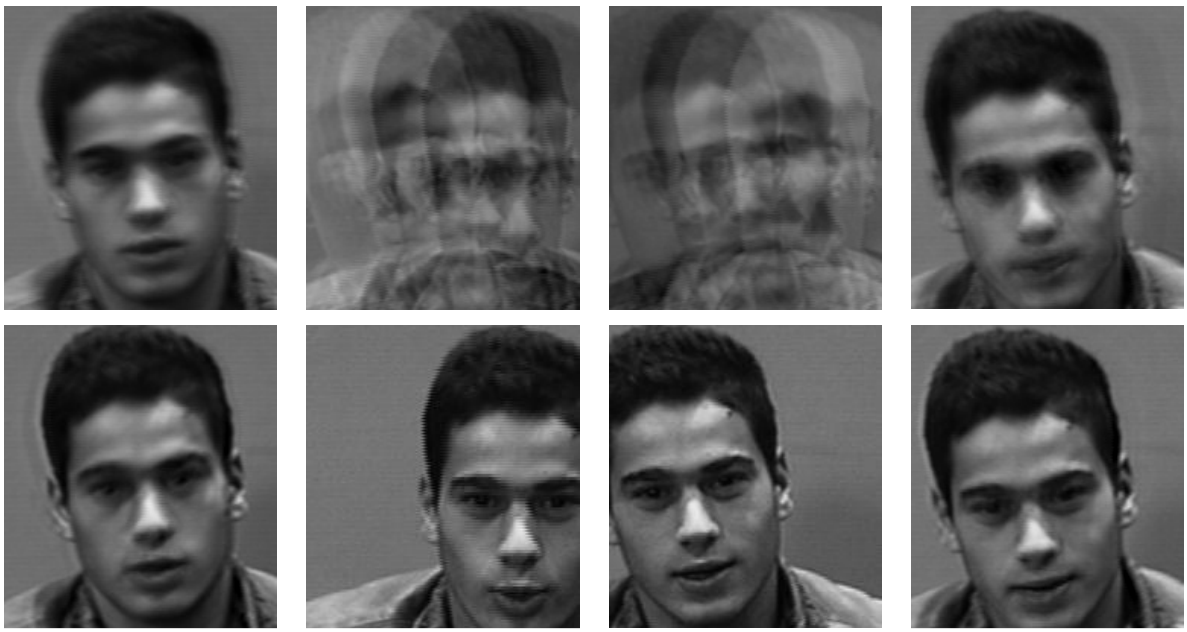
4 Experimentální výsledky

Nejprve uvedeme výsledky simulací základní verze obrazového kodéru uvedené na obr. 1. Vstupní data jsou tvořena obrazovou sekvencí zachycující mluvčího. Každý ze snímků má rozměry 200x180 pixelů, je ve stupních šedé a počet snímků v sekvenci je N=20. Na obr. 3 je pro ilustraci uvedeno několik originálních, nekomprimovaných snímků z této sekvence. Použitá obrazová data pocházejí ze zdroje [5]. Pozn.: je-li v dalším textu uvedeno, že se ke kompresi používá n hlavních komponent, vždy se tím myslí n nejvýznamnějších hlavních komponent.

Pořadí 10. Pořadí 13. Pořadí 15. Pořadí 20.

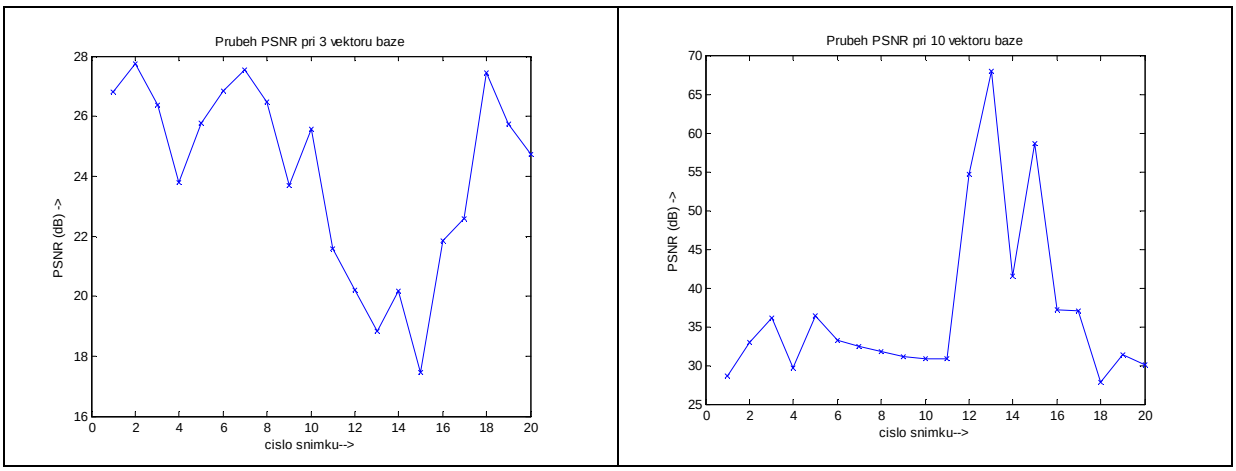
Obr. 3 Ukázka vybraných originálních snímků vstupujících do obrazového kodéru. Umístění snímku určuje pořadí daného snímku v sekvenci.

První experiment používá k odhadu počtu hlavních komponent jednoduché Kaiserovo pravidlo, které doporučuje použít pouze takové vlastní vektory, jejichž příslušná vlastní čísla jsou větší než průměrná hodnota vlastního čísla. Komprimované snímky, které korespondují se snímky na obr. 3, jsou uvedeny na obr. 4. Pro porovnání jsou na obr. 4 (spodní sada snímků) ještě znázorněny tytéž snímky komprimované při použití 10 hlavních komponent.



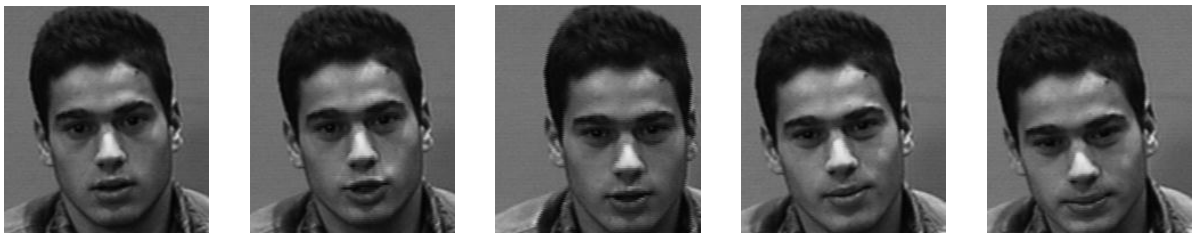
Obr. 4 Komprimované snímky z obr. 3 při použití 3 nejvýznamnějších hlavních komponent, jejichž počet byl určen Kaiserovým pravidlem (první čtveřice snímků shora). Čtveřice dole je komprimována při uvažování 10 nejvýznamnějších komponent.

Z průběhů PSNR na obr. 5 na následující straně vidíme, že pro 3 hlavní komponenty jsou hodnoty PSNR pro snímky v pořadí 13. a 15. nízké. Na obr. 4 se můžeme přesvědčit, že tyto snímky jsou pro tento počet komponent v podstatě nepoužitelné. Uvažujeme-li ale 10 hlavních komponent, situace se znatelně změní. Snímky, které měly nízkou hodnotu PSNR, nyní dosahují hodnot PSNR vyšších než snímky ostatní (u nich k výraznému nárůstu PSNR nedošlo). Tento stav je zřejmě dán tím, že u snímků 13 a 15 (a některých dalších, které zde pro omezený prostor neuvádíme) je poloha hlavy mluvčího ve výrazně jiných pozicích než u ostatních snímků. Snímky s relativně ustálenou pozicí hlavy mluvčího potřebují malý počet komponent (zde stačí 3) pro hrubší rozlišení polohy, ovšem vyžadují vyšší počet pro rozpoznání drobných detailů ve snímku (poloha očí, úst apod.).



Obr. 5 Průběhy PSNR pro všechny komprimované snímky v sekvenci pro 3 a 10 použitých hlavních komponent

Další experiment se zabývá aplikací vývojového diagramu na obr. 2 pro potřeby odhadu počtu hlavních komponent pro kompresi snímků z hlediska možného nasazení procesu rozpoznávání. Snahou je, aby se natrénováním kodéru dosáhlo stanovené teoretické úspěšnosti rozpoznání komprimovaného snímku. V našem případě použijeme jako trénovací sadu množinu 20 vstupních snímků. Z těchto vstupních snímků vybereme 5 reprezentantů, které postupně komprimujeme s použitím daného počtu hlavních komponent a vyčíslujeme procento úspěšnosti rozpoznání. Prohlásíme, že proces rozpoznávání je úspěšný, pokud je správně rozpoznáno alespoň 4 z 5 reprezentantů. Odpovídající počet hlavních komponent pak použijeme pro kompresi zbývajících 15 snímků. Proces rozpoznávání byl prováděn v aplikaci prezentované v [1]. Obr. 6 uvádí reprezentanty, podle nichž natrénování kodéru probíhá.



Obr. 6 Snímky použité pro natrénování kodéru podle algoritmu popsaného vývojovým diagramem na obr. 2

Úspěšného rozpoznání (4 z 5 rozpoznáno správně) bylo dosaženo u snímků komprimovaných použitím 7 hlavních komponent. Pro ilustraci si na obr. 7 uveďme komprimované reprezentanty pro tento počet komponent.

Obr. 7 Komprimované snímky z obr. 6 po natrénování obrazového kodéru. Použito 7 hlavních komponent.

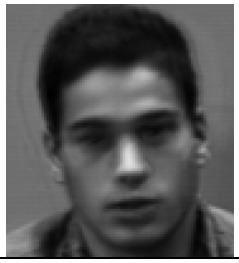
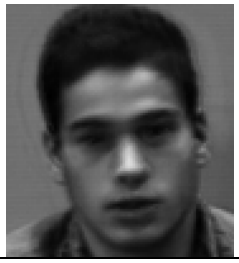
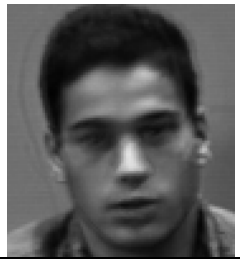
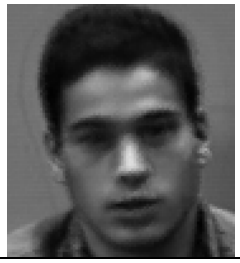
Tab. 1 uvádí pro ilustraci výsledky rozpoznávání pro 6, 7 a 8 hlavních komponent.

Tab. 1 Úspěšnost rozpoznání pro různé počty hlavních komponent

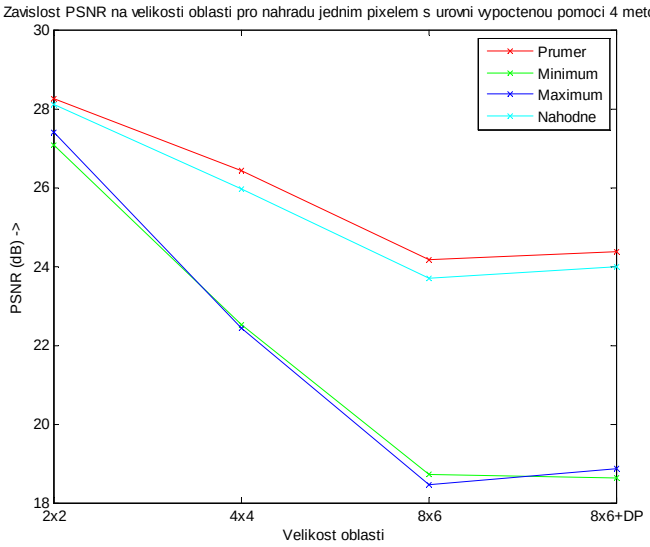
Hlavních komponent	*Snímek 1*	*Snímek 2*	*Snímek 3*	*Snímek 4*	*Snímek 5*
6	Rozpoznán	Rozpoznán	Rozpoznán	Nerozpoznán	Nerozpoznán
7	Rozpoznán	Rozpoznán	Rozpoznán	Rozpoznán	Nerozpoznán
8	Rozpoznán	Rozpoznán	Rozpoznán	Rozpoznán	Rozpoznán

Další část experimentů se zabývala možnostmi snížení výpočetní náročnosti základní verze obrazového kodéru. Jako první si uveďme výsledky simulací pro případ, kdy jsme nahradili danou skupinu pixelů ve snímku jedním pixelem. Úroveň tohoto pixelu jsme stanovili na základě průměrné, minimální, maximální a náhodně vybrané úrovně z dané skupiny pixelů. Skupina pixelů byla tvořena bloky velikosti 2x2, 4x4 a 8x6 (výška x šířka) pixelů. Výsledky simulací jsou uvedeny v tab. 2, kde jsou pro úsporu místa zobrazeny pouze výsledky pro 1 vstupní snímek, jehož originál je na obr. 3, první zleva.

Tab. 2 Výsledky simulací pro zmenšení výpočetní náročnosti na základě náhrady skupiny pixelů jedním pixelem

Velikost oblasti	*Průměrná úroveň pixelu*	*Minimální úroveň pixelu*	*Maximální úroveň pixelu*	*Náhodně vybraná úroveň pixelu*
2x2				
4x4				
8x6				
8x6 a filtrace dolní propustí				

Na obr. 8 je uveden graf závislosti PSNR na metodě pro náhradu skupiny pixelů pro 3 uvažované velikosti oblastí. Pro oblast 8x6 byly komprimované snímky podrobeny ještě procesu filtrace dolní propustí pro snížení viditelnosti blokové struktury. Ve všech případech jde o snímek z obr. 3, první zleva. Bylo použito 7 hlavních komponent (zvoleno podle výsledků předchozí simulace). Vidíme, že se zvětšující se oblastí klesá PSNR. Nejlepší výsledky týkající se velikosti PSNR dává metoda nahrazující úrovně pixelů v oblasti průměrnou úrovní. Aplikace dolní propusti přispěla ke zlepšení subjektivní i objektivní kvality komprimovaného snímku.

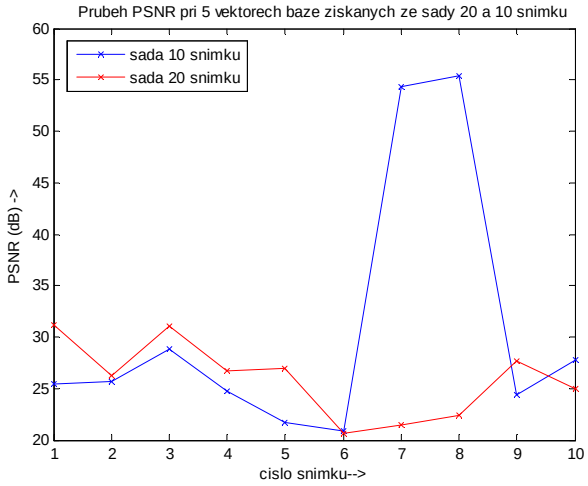


Obr. 8 Graf závislosti PSNR na metodě pro náhradu skupiny pixelů pro 3 uvažované velikosti oblastí, přičemž snímek s oblastí 8x6 pixelů byl navíc po kompresi podroben procesu filtrace dolní propustí. Použit snímek z obr. 3, první zleva.

Poslední experiment se zabývá rozdělením vstupní sekvence 20 snímků na dvě sekvence délky 10, přičemž transformační matice $\mathbf{K}$ se stanoví pouze pro jednu z nich. Druhá použije tutéž transformační matici $\mathbf{K}$. Aby byly ve zkrácené sekvenci rovnoměrně zastoupeny snímky z celé původní sekvence, vstupuje do zkrácené sekvence každý 2. snímek. Obr. 8 uvádí pro ilustraci 5 vybraných komprimovaných snímků. První řádek obsahuje snímky komprimované pomocí 5 hlavních komponent stanovených z celé sekvence 20 snímků, ve druhém řádku jsou stejné snímky jako v řádku prvním, ale jsou komprimovány předem vypočtenou transformační maticí z první sady 10 snímků při uvažování 5 hlavních komponent.

Obr. 8 K rozdělení původní sekvence na 2 sekvence stejné délky. První řádek obsahuje snímky komprimované pomocí 5 hlavních komponent stanovených z celé sekvence 20 snímků, ve druhém řádku jsou stejné snímky jako v řádku prvním (jsou součástí druhé sady 10 snímků), ale jsou komprimovány předem vypočtenou transformační maticí z první sady 10 snímků při uvažování 5 hlavních komponent.

Pro doplnění je uveden na obr. 9 průběh PSNR pro snímky z obr. 8.



Obr. 9 Průběh PSNR pro snímky z obr. 8

5 Závěr

V tomto příspěvku byl prezentován obrazový kodér na bázi KLT, který matici vstupních dat konstruuje na základě několika snímků, čímž do procesu komprese zanáší časový vývoj obrazové scény. Kritickým místem prezentovaného obrazového kodéru je blok pro stanovení počtu hlavních komponent. Byla popsána metoda odhadu tohoto počtu na základě stanovení akceptovatelné úspěšnosti rozpoznání vzoru z trénovací sady snímků ke komprimovanému snímku. Další část příspěvku se zabývá možnostmi snížení výpočetní náročnosti (paměťové i časové) základní verze obrazového kodéru. První způsob redukce výpočetní náročnosti je založen na náhradě skupiny pixelů ve vstupním obrazu jedním pixelem při použití různých metod pro stanovení nahrazující úrovně a různých velikostí oblastí pro nahrazení. Druhý způsob redukce výpočetní náročnosti je založen na rozdělení vstupní sekvence na dvě sekvence, přičemž transformační matice je stanovena pouze pro jednu z nich. Druhá používá již vypočtenou transformační matici. Všechny uvedené simulace dávají zajímavé výsledky, které mohou být podkladem pro další práci v této oblasti. Velice vhodná je pro další experimentování především metoda pro odhad relevantních hlavních komponent na základě úspěšného rozpoznání. Rovněž vcelku jednoduchá metoda pro snížení výpočetní náročnosti na základě náhrady skupiny pixelů může být inspirací pro další experimenty vedoucí např. ke zlepšení subjektivní kvality komprimovaných snímků (redukce blokové struktury).

Poděkování

Tento projekt je podporován interním grantem ČVUT CTU CTU0803213 - Algoritmus pro kompresi videa pro kamerové zabezpečovací systémy.

Reference

[1] Fritsch, L. Metoda PCA a její implementace v jazyce C++. Sborník příspěvků 15. Ročníku konference Technical Computing Prague 2007. Praha, 2007. ISBN 978-80-7080-658-6.

[2] Fritsch, L. Low bitrate video coding using Karhunen- Loeve transform. In Proceedings of the 18th International Conference Radioelektronika 2008. Praha: Československá sekce IEEE, 2008. ISBN 978-1-4244-2087-2.

[3] Metoda hlavních komponent [soubor PDF]. URL:< http://www2.zf.jcu.cz/public/departments/kmi/MSMT_05/pca.pdf>.

[4] A tutorial on Principal Component Analysis [soubor PDF]. URL:<http://www.snl.salk.edu/~shlens/pub/notes/pca.pdf>.

[5] Collection of Facial Images: Faces94 [online]. URL:<http://cswww.essex.ac.uk/mv/allfaces/faces94.html>.

Kontakt:
Ing. Lukáš Fritsch
ČVUT v Praze
Fakulta elektrotechnická
katedra radioelektroniky
Technická 2, Praha 6- Dejvice, 166 27
E-mail: fritsl1@fel.cvut.cz