Identifikace obličeje osoby snímané kamerou

Person face identification using digital camera

Bc. Petr Morávek

Diplomová práce
2010

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2009/2010

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Petr MORÁVEK**
Studijní program:	**N 3902 Inženýrská informatika**
Studijní obor:	**Informační technologie**
Téma práce:	**Identifikace obličeje osoby snímané kamerou**

Zásady pro vypracování:

1. **Vytvořte SW k identifikaci osoby podle statického obrazu snímaného kamerou připojenou k PC.**
2. **K identifikaci využijte emulaci neuronové sítě s architekturou Neocognitron.**
3. **Vytvořte databázi známých tváří, v nichž bude program hledat snímanou tvář.**
4. **Vyhodnoťte přesnost rozpoznávání.**
5. **Srovnejte výsledky s jinými používanými technikami.**
6. **Zpracujte programovou dokumentaci.**

Rozsah práce:

Rozsah příloh:

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

1. **FUKUSHIMA, K., MIYAKE S., ITO, T.: Neocognitron: a neural network model for a mechanism of visual pattern recognition. IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-13(Nb. 3):pp.826—834, September/October 1983.**
2. **FUKUSHIMA K.: Visual pattern recognition with neural networks, 1992, ISBN 978-3-540-56346-4.**
3. **ZELINKA, I.: Umělá inteligence I, VUT Brno, 1998, ISBN 80-214-1163-5.**
4. **BOSE, N.K., LIANG, P.: Neural Network Fundamentals with Graphs, Algorithms, and Applications, McGraw-Hill Series in Electrical and Computer Engineering, 1996, ISBN 0-07-006618-3.**
5. **TAKIMOTO, H., MITSUKURA, Y., AKAMATSU, N., KHOSLA, R.: Face Identification Method Using the Face Shape, 2003, ISBN 978-3-540-40803-1.**
6. **FUKUSHIMA K.: Neocognitron: a self-organizing neural network model for a mechanism of patterns recognition unaffected by shift in position, Biological Cybernetics, 32:193-202,1980**
7. **BISHOP, Ch. M.: Pattern Recognition and Machine Learning, Springer, 2006, ISBN 0-387-31073-8.**

Vedoucí diplomové práce: **Ing. Zuzana Oplatková, Ph.D.**
Ústav informatiky a umělé intelligence

Datum zadání diplomové práce: **19. února 2010**

Termín odevzdání diplomové práce: **8. června 2010**

Ve Zlíně dne 19. února 2010

prof. Ing. Vladimír Vašek, CSc.
děkan

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky
L.S.
①

prof. Ing. Vladimír Vašek, CSc.
ředitel ústavu

ABSTRAKT

Cílem mojí diplomové práce je navrhnout a implementovat SW řešení identifikace osob sledovaných statickou kamerou. K rozpoznávání osob bude použit princip neuronové sítě s architekturou neocognitron optimalizované k rozpoznávání charakteristik obličejových rysů. Program by měl být schopen pracovat na běžném domácím počítači. Schopnost rozpoznat osobu musí být prokazatelně patrná. Součástí návrhu bude i programátorská a uživatelská dokumentace.

Teoretická část práce bude obsahuje obecný princip neuronových sítí a popis struktury a principu architektury neocognitron.

Klíčová slova: Neuronová síť, identifikace, rozpoznání, neocognitron.

ABSTRACT

The target of my disertation is to design and implement software solutions for the identification of persons reporting a static camera. The recognition of the principle used neural network architecture Neocognitron optimized to recognize the characteristics of facial features. The program should be able to work on normal home computer. Capablity to recognize a person must be demonstrably evident. Disertation must contain the programmer and the user documentation. The theoretical part will contains the general principles of neural networks and a description of the structure and principles of architecture Neocognitron.

Keywords: Neural network, identification, recognition, neocognitron

PODĚKOVÁNÍ

Děkuji vedoucí diplomové práce Ing. Zuzaně Oplatkové, Ph.D. za konzultace, vstřícnou pomoc a věcné připomínky k zdokonalení diplomové práce. Dále bych chtěl poděkovat všem členům mé rodiny, kteří se spolupodíleli na testování programu jako pokusné osoby.

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně …….

podpis diplomanta

OBSAH

ÚVOD

Rozpoznávání obličeje přitahuje značnou pozornost. Někdy jde o porovnávání dvou statických obrazů, jindy o potřebu ověřit identitu jednotlivce nacházejícího se ve skupině lidí.

Doposud měly obličejové rozpoznávací systémy jen omezený úspěch, dnes však vychází najevo, že rozpoznávání obličejů se bude řadit mezi primární technologie pro zajištění systémů vysokých rizik.

Automatizované rozpoznávání lidských obličejů je obtížný komplexní úkol z důvodů proměnlivosti základních fyzikálních veličin obrazu, jakosti a fotometrie, geometrie polohy obličeje – úhlu natočení a přiblížení (zoomu), morfologie změn – emoční výrazy obličeje a stárnutí, a „přestrojení“ (čepice, brýle, vousy). Odtud vyplývá nutnost vytvoření normalizovaného modelu lidského obličeje tak, aby scénář rozpoznávání nebyl ovlivněn těmito reálnými rušivými vlivy. Modelování lidského obličeje se opírá o různé techniky numerického modelování – Fourierův popis, kruhové harmonické expanze, autoregresivní modely a momentové invariance. [1]

V této práci chci navrhnout program a postup identifikace osoby v obraze pomocí neuronové sítě s architekturou neocognitron. K identifikaci bude využit softwarový model neuronové sítě vytvořený v programovacím jazyku C#.

Na téma rozpoznávání obličejů bylo napsáno mnoho prací a projektů s dobrými výsledky. I přesto se mi tato oblast jeví velmi zajímavá a z mého pohledu i tak trochu neprobádaná. Proto jsem si zvolil toto téma.

Je-li neuronová síť implementována na jednom počítači, je typicky pomalejší než tradiční algoritmické řešení. Paralelní povaha neuronových sítí nicméně umožňuje, aby byla vytvořena pomocí mnoha procesorů. Výhodou je velká rychlost při velmi nízkých nákladech na vývoj. Paralelní architektura také umožňuje neuronovým sítím efektivně zpracovávat rozsáhlé množiny dat. Když neuronové sítě zacházejí s rozsáhlým, nepřetržitým proudem informací, jako je rozpoznávání řeči nebo data ze strojových čidel, mohou pracovat podstatně rychleji než jejich lineární protějšky.

Umělé neuronové sítě se ukázaly jako užitečné v rozmanitých aplikacích skutečného světa, které pracují s komplexními a často neúplnými daty. První aplikace neuronových sítí byly v

oblasti rozpoznávání vizuálních předloh a rozpoznávání řeči. Kromě toho využívají neuronové sítě také současné programy pro převod textu na hlas. I mnohé programy pro analýzu písma například ty v populárních organizérech PDA využívají ke své činnosti právě neuronovou síť.

I. TEORETICKÁ ČÁST

1 ARCHITEKTURA NEURONOVÉ SÍTĚ NEOCOGNITRON

V této kapitole si kladu za cíl popsat obecně základní vlastnosti neuronových sítí, jejich členění, možnosti jejich použití a princip učení a vybavování. Samostatně pak popíši architekturu a princip neuronové sítě s architekturou neocognitron.

1.1 Neuronová síť

Neuronová síť je jedním z výpočetních modelů používaných v umělé inteligenci. Jejím vzorem je chování odpovídajících biologických struktur. Skládá se z umělých (nebo také formálních) neuronů, jejichž předobrazem je biologický neuron. Neurony jsou vzájemně propojeny a navzájem si předávají signály a transformují je pomocí určitých přenosových funkcí. Neuron má libovolný počet vstupů, ale pouze jeden výstup.

Formální neuron je v podstatě jednoduchá jednotka, která ohodnotí-vynásobí všechny vstupy jejich vahami (váhy se mění během učení – odtud plyne adaptace sítě) a takto získané hodnoty sečte. Výslednou hodnotu dosadí do přenosové funkce neuronu (obecně vzato je to funkce, která určuje, jaká odezva bude na vstupní podnět) a výstup této funkce je i výstupem z neuronu, který slouží jako vstup do neuronů dalších.[2]

1.1.1 Dělení neuronových sítí

Technické (umělé) neuronové sítě můžeme dělit podle několika základních kriterií. Zde uvedu jen základní členění. Podrobněji k tomuto tématu např. v [2].

Podle počtu vrstev:

- Jednovrstvou
- Vícevrstvou

Podle typu algoritmu učení:

- S učitelem
- Bez učitele

Podle stylu učení

- Deterministické
- Stochastické

1.1.2 Životní fáze neuronové sítě

Neuronová síť prochází postupně třemi fázemi.

1. Konstrukce sítě: V této fázi se určí počet a typ neuronů a spojů mezi nimi.
2. Učení: V této fázi se nastavují váhy u spojů mezi neurony.
3. Vybavování: V této fázi síť pracuje, provádí výpočty svých neuronů a poskytuje výstup na základě svých vstupů.

1.1.3 Princip funkce neuronových sítí typu perceptron

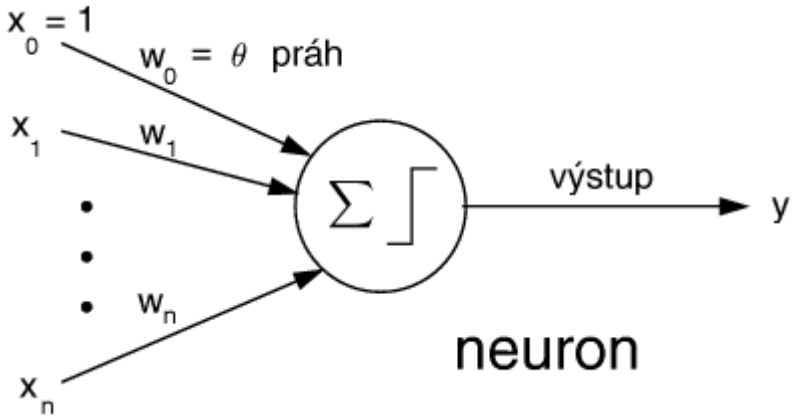
Existuje celá řada typů neuronových sítí, lišících se zejména způsobem spojení mezi jednotlivými neurony, způsobem učení a způsobem výpočtu. Nejčastěji používaným modelem neuronu je perceptron. Neuronová síť neocognitron jako výpočetní jednotku používá také perceptron. Zde popíši základní princip činnosti perceptronu a perceptronové sítě.

Perceptron sestává z jediného výkonného prvku modelovaného obvykle McCullochovým a Pittsovým modelem neuronu, který má nastavitelné váhové koeficienty a nastavitelný práh. Někteří autoři označují stejným názvem i celou síť takových prvků. Algoritmus vhodný k nastavení parametrů perceptronu publikoval poprvé F. Rosenblatt v roce 1958 a později v roce 1962. Rosenblatt dokázal následující větu:

„ Máme-li v n-rozměrném prostoru lineárně separabilní třídy objektů, pak lze v konečném počtu kroků učení (iterací optimalizačního algoritmu) nalézt vektor vah W perceptronu, který oddělí jednotlivé třídy bez ohledu na počáteční hodnotu těchto vah. “ [3]

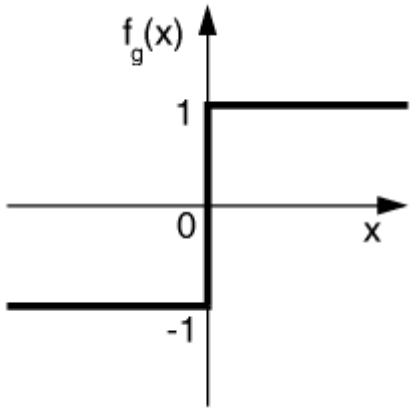
Princip Perceptronu

Princip funkce reálných biologických neuronů je velmi podobný úloze o klasifikaci. Skutečný neuron živých organismů se skládá z těla, dendridů, které plní funkci vstupů do neuronu a axomu, který je výstupem z neuronu. Model neuronu používaný pro umělé neuronové sítě bývá navržen tak, aby byl co nejjednodušší, přitom musí ale co nejlépe plnit funkci biologického neuronu.[2],[4]



Obr. 1. Základní model neuronu. Převzato z [2]

Nejčastěji používaný model neuronu je na obrázku (Obr. 1) Vstupy x1, x2, ..., xn jsou jednotlivé elementy vstupního vzoru X. Koeficienty w1, w2, ..., wn jsou jednotlivé složky váhového vektoru W (představují vnitřní parametry neuronu).



Obr. 2. Funkce signum. Převzato z [2]

Dalším vnitřním parametrem neuronu je prahová hodnota θ. Hodnota výsledné funkce může nabývat nějaké reálné hodnoty. Často potřebujeme znát pouze znaménko výstupu. Proto zavádíme prahovou funkci fh, která se také nazývá Heavisideova funkce. Tvar této funkce je na obrázku (Obr. 2).

Výsledný výstup neuronu bude tedy popsán vztahem:

$$y = f_h\left(\sum_{i=1}^{N} w_i x_i - \theta\right) \tag{1}$$

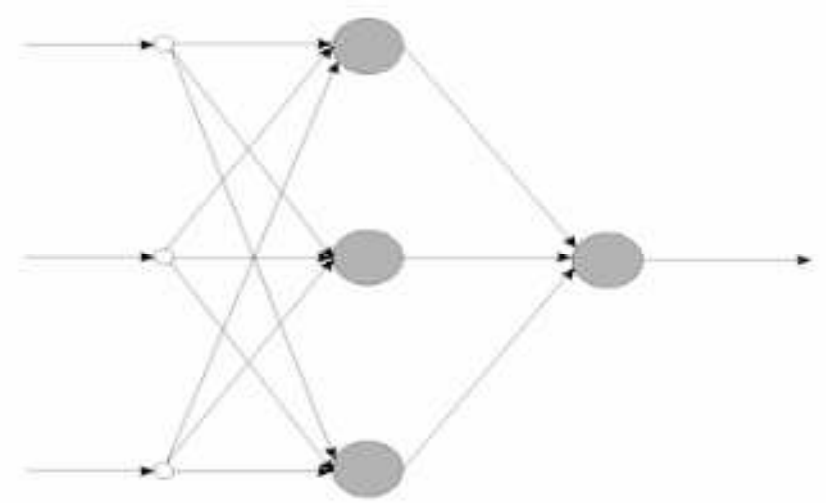
Převzato z [2]

Někdy se zahrnuje práh θ do součtu. Potom je považován za novou váhu s hodnotou $-\theta$ připojenou ke vstupu s konstantní hodnotou. Výstupní hodnota bude potom určena vztahem:

$$y = f_h\left(\sum_{i=0}^{N} w_i x_i\right) \tag{2}$$

Převzato z [2]

Preceptron s jedním výkonným prvkem ale umožňuje nanejvýš klasifikaci do dvou tříd, které musí být lineárně separabilní. Zvětšíme-li však počet výkonných prvků pracujících v perceptronu a zvětšíme-li i počet jeho vrstev, je možno jím klasifikovat do více tříd. Tyto třídy již nemusí být lineárně separabilní, musí však být separabilní. Více např. v [2],[3],[4].



Obr. 3. Rosenblattova perceptronova síť. Převzato z [2]

Neurony v první vrstvě, jak je znázorněno na obrázku (Obr. 3) neprovádí žádné výpočty. Jedná se pouze o receptory, které předávají svou aktivitu skryté vrstvě. Jednotlivé neurony skryté vrstvy a vrstvy výstupní se chovají stejně jako již popsaný perceptron.

1.2 Architektura neocognitron

Neocognitron je hierarchická vícevrstvá neuronová síť navržená profesorem Fukushimou v roce 1980. Fukushima se k jejímu vytvoření inspiroval modelem Hubela a Wiesela z roku 1959. Ti objevili a popsali ve vizuálním kortexu dva typy neuronových buněk, které nazvali *simple cell a complex cell.* Sestavili také kaskádový model propojení a funkce těchto buněk.

Z hlediska členění popsaného v kap. 1.1.1 se jedná o vícevrstvou deterministickou síť ve variantách učení s učitelem i bez učitele. Jednotlivé buňky se chovají stejně jako perceptron, jen s tím rozdílem, že přenosová funkce není skoková. Přenosová funkce neocognitronu bude popsána později.

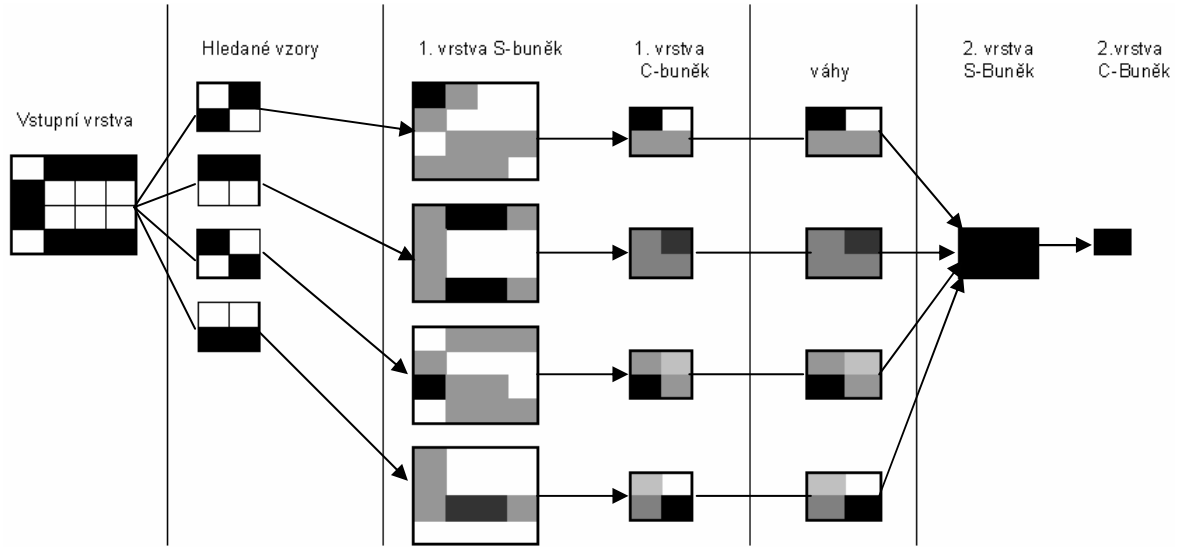
V současné době existuje již mnoho různých variant této neuronové sítě. Původní dvě základní verze navržené Fukushimou se od sebe odlišují především použitým principem učení. Jedná se o neocognitron využívající :

- učení bez učitele
- učení s učitelem

První verze sítě neocognitron byla založena na principu učení bez učitele. Tato verze je také často označována pojmem samoorganizujícíse neocognitron. Hlavní výhodou neuronové sítě neocognitron je schopnost správně rozpoznávat nejen naučené vzory, ale i vzory, které z nich vzniknou částečným posunutím, otočením nebo jinou deformací. [6],[7]

1.3 Základní princip fungování sítě neocognitron

Princip sítě je založen na postupném rozpoznávání vzoru od nejmenších částí, jejich postupným skládáním do větších celků a nakonec celkovým sestavením. Pouze slovní popis postupu by nebyl dostatečně vypovídající, proto předkládám na demonstraci dva obrázky.



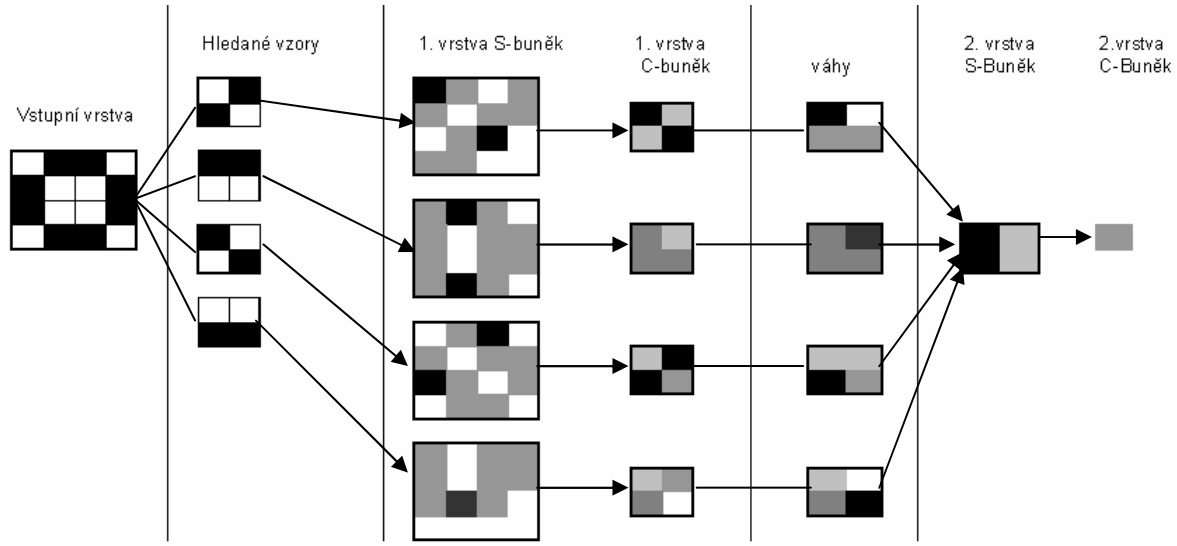
Obr. 4. Postupné rozpoznání vzoru sítí.

Na prvním obrázku (Obr. 4) je vidět z leva vstupní vrstva s obrazem písmene „c“, který v tomto případě přesně odpovídá naučenému vzoru. Hledané vzory jsou čtyřbodové fragmenty obrazu ve druhém sloupci.

Ve třetím sloupci je naznačena aktivita S-buněk. Je dána tím, do jaké míry hledaný vzor odpovídá předloženému, a to v každém bodě předloženého vzoru. Černá barva znamená, že v tomto bodě odpovídá přesně a bílá barva znamená nulovou shodu. Dále je zde vidět aktivita C-buněk první vrstvy. V podstatě se jedná o vážený průměr aktivity S-buněk vždy pro příslušnou část oblasti. Každá C-buňka v 1. vrstvě připojuje oblast 2x2 S-buňky. Aktivita těchto ploch zde přesně odpovídá vahám v následujícím sloupci, a proto je zde již ve 2. S-vrstvě maximální aktivita všech buněk.

V posledním sloupci je vidět vrstva s pouze jednou C-buňkou, která má také maximální aktivitu a indikuje shodu předloženého vzoru s naučeným.

Na druhém obrázku (Obr. 5) je ještě zobrazen výsledek stejně naučené sítě při předložení písmenka „o“ místo „c“. Mimo jiné je možno v tomto modelu sledovat ve 2. S-vrstvě, ve kterých místech a jak moc se lišil vzor od originálu.

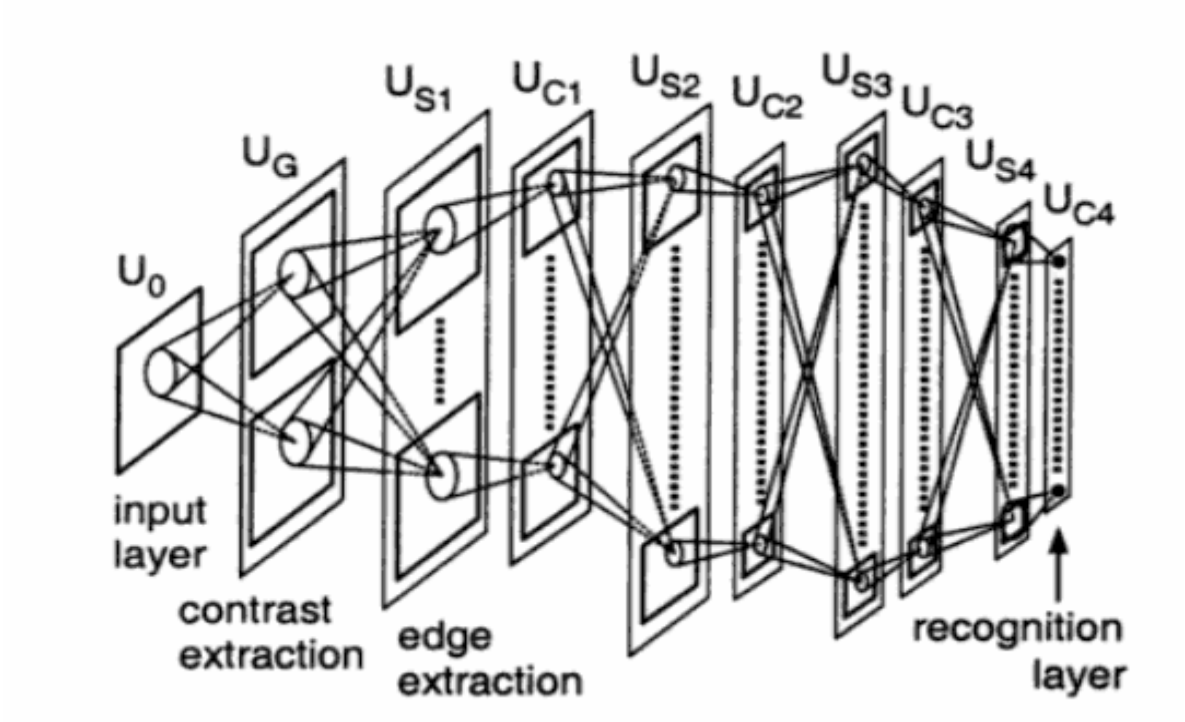


Obr. 5. Vyhodnocení znaku „o“ na vstupu sítě naučené na znak „c“.

1.4 Struktura sítě

Síť neocognitron, jak již bylo řečeno, je složena z několika vrstev složených střídavě z buněk S-cell a C-cell, dále ze vstupní vrstvy a pomocných vrstev buněk V-cell.

Původní struktura navržená Fukushimou neobsahovala ještě pomocné vrstvy V. O ty byla doplněna později. Schéma původní struktury je vidět na obrázku (Obr. 6.). V této variantě je mezi vstupní vrstvou a vrstvou S1 vložena ještě jedna vrstva nazvaná „contrast extraction“. Tato vrstva se používá pro vstupy, které nejsou jen černobílé a obsahují škálu odstínů. Vrstva má za úkol vytvořit kontrastnější obraz, který je vhodnější pro rozpoznávání.



Obr. 6. Struktura neocognitronu navržená profesorem Fukushimou. Převzato z [5]

1.4.1 S-vrstvy

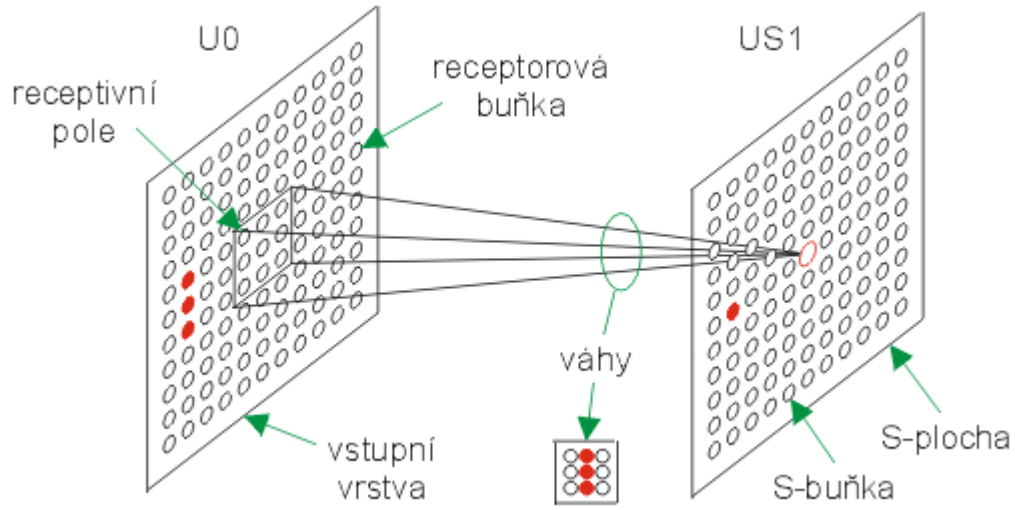
Každá S-vrstva je složená z několika ploch, kde každá plocha rozměrově odpovídá rozměrům vstupní plochy nebo připojené C-plochy vyšší úrovně. Počet ploch ve vrstvě S přesně odpovídá počtu vzorů, které se v této vrstvě vyhodnocují. V příkladu na obrázcích (Obr. 4) a (Obr. 5) měla 1. S-vrstva 4 plochy a 2. S-Vrstva jen jednu.

Každou plochu si můžeme představit jako dvourozměrné pole buněk. Tuto představu uvádím pro zjednodušení, protože takto byly plochy použity v příkladu v minulé kapitole a je optimální pro vyhodnocování dvourozměrných obrazových vzorů. Nicméně počet rozměrů pole buněk není nijak omezen.

Každá buňka S-plochy je propojena pomocí vah do připojovací oblasti C-ploch z C-vrstvy vyšší úrovně nebo ke vstupní vrstvě. Pomocí vah se přenáší aktivita buněk v připojovací oblasti na S-buňku. Váhy spojující vstupní vrstvu nebo C-vrstvu vyšší úrovně s S-buňkou se nazývají a-váhy. Každá sada a-vah v podstatě určuje jeden rozpoznávaný vzor sítě.

Síť neocognitron má oproti jiným neuronovým sítím jednu zvláštnost. Každá S-plocha má váhy pro všechny svoje buňky společné. To znamená, že síť může mít a zpravidla také mívá, víc buněk než fyzických vah. Ve skutečnosti každá buňka váhy používá, takže je pro

výpočet jedno jestli, má pro všechny buňky jednu společnou sadu vah nebo má každá buňka vlastní sadu vah, ale s váhami shodnými s ostatními buňkami. Je jasné, že v živých neuronech asi není možné sdílet váhy mezi jednotlivými buňkami. V technických řešeních je zase naopak zbytečné udržovat mnohonásobné kopie stejných hodnot.



Obr. 7. Připojení připojovací oblasti a buněk S-plochy. Převzato z [6]

Na obrázku (Obr. 7) je vidět připojení buněk S-plochy se vstupní vrstvou. Buňky jsou připojeny 9-ti váhami, které reprezentují určitý hledaný vzor. Aktivní buňka ve druhém sloupci S-plochy ukazuje místo výskytu vzoru v připojovací oblasti.

Nejlepší způsob popisu aktivity buněk v S-vrstvě je matematický popis. Pro zjednodušení popisu si představíme jednotlivé plochy sítě jako jednorozměrné. To znamená, že poloha buňky v ploše a poloha váhy v poli vah budou určeny pouze pořadovým číslem. Kompletní přesné označení buňky bude dáno pořadovým číslem vrstvy, pořadovým číslem plochy ve vrstvě a pořadovým číslem buňky v ploše. Hodnota váhy bude určena pořadovým číslem vrstvy, pořadovým číslem plochy ve vrstvě a pořadovým číslem v připojovací oblasti.

Výpočet hodnoty buňky S je vyjádřen vztahem:

$$u_S(l,n,\kappa) = r_l \cdot \varphi \cdot \left[\frac{1 + \sum_{k=1}^{K_{C(l-1)}} \sum_{v \in A_l} a_l(v,k,\kappa) \cdot u_{C(l-1)}(n+v,\kappa)}{1 + \frac{r_l}{1+r_l} \cdot b_l(k) \cdot u_{Vl}(n)} - 1 \right] \tag{3}$$

kde

u_S	je	buňka v S-vrstvě
u_V	je	buňka v V-vrstvě
u_C	je	buňka v C-vrstvě nebo vstupní vrstvě
l	je	pořadové číslo vrstvy
n	je	pořadové číslo buňky v ploše
κ	je	pořadové číslo plochy ve vrstvě
r	je	Selektivita (nastavitelný parametr vrstvy)
v	je	pořadí buňky v připojovací oblasti
b	je	hodnota b-váhy (mezi vrstvou V a S)
a	je	hodnota a-váhy (mezi připojovací oblastí a vrstvou S)
φ	je	nelineární funkce $\varphi[x] = \begin{cases} x \; pro \; x \geq 0 \\ 0 \; pro \; x < 0 \end{cases}$

Jak je vidět ve vztahu (3), pro výpočet hodnoty aktivity buňky v S-Ploše musíme mít k dispozici ještě hodnoty aktivity buněk ve vrstvě V ($u_{vl}(n)$ je aktivita buňky s indexem n ve vrstvě V, ploše s indexem l). Podrobněji například [7],[8],[9].

1.4.2 V-vrstvy

Vrstva V slouží k potlačení aktivity buněk v připojovací oblasti. Hodnota buňky z vrstvy V se nachází ve jmenovateli zlomku (3), proto bude vyšší aktivita buňky vrstvy V snižovat aktivitu odpovídajících buněk vrstvy S. Působí tedy jako inhibitor. Její význam se dá velmi jednoduše popsat na příkladu, kdy se síť naučí nějaký vzor. Potom je plocha S schopná detekovat vzor, ve kterém aktivní buňky pokryjí naučený vzor. Pokud budou ve vstupní oblasti kromě buněk odpovídajících vzoru aktivní i další buňky, byla by výsledná aktivita buňky v S-ploše stejná jako v případě, kdy vzor přesně odpovídá naučenému znaku. Například úplně celá černá(aktivní) plocha by se také shodovala se vzorem.

Tento nedostatek řeší Vrstva V. Každá vrstva V má pouze jednu plochu. Každá buňka V-plochy obsahuje průměrnou aktivitu buněk v připojovací oblasti. K navazující vrstvě S se vrstva V chová inhibičně. Tedy snižuje aktivitu odpovídajícím S-buňkám. Čím víc buněk je aktivních v připojovací oblasti V-buněk, tím vyšší bude jejich aktivita a tím více budou snižovat hodnotu aktivity příslušných buněk v S-Ploše.

Matematický popis výpočtu hodnoty buňky ve vrstvě V:

$$u_{Vl}(n) = \sqrt{\sum_{\kappa=1}^{K_{C(l-1)}} \sum_{v \in Al} c_l(v) \cdot u_{Cl-1}^2(n+v, \kappa)} \tag{4}$$

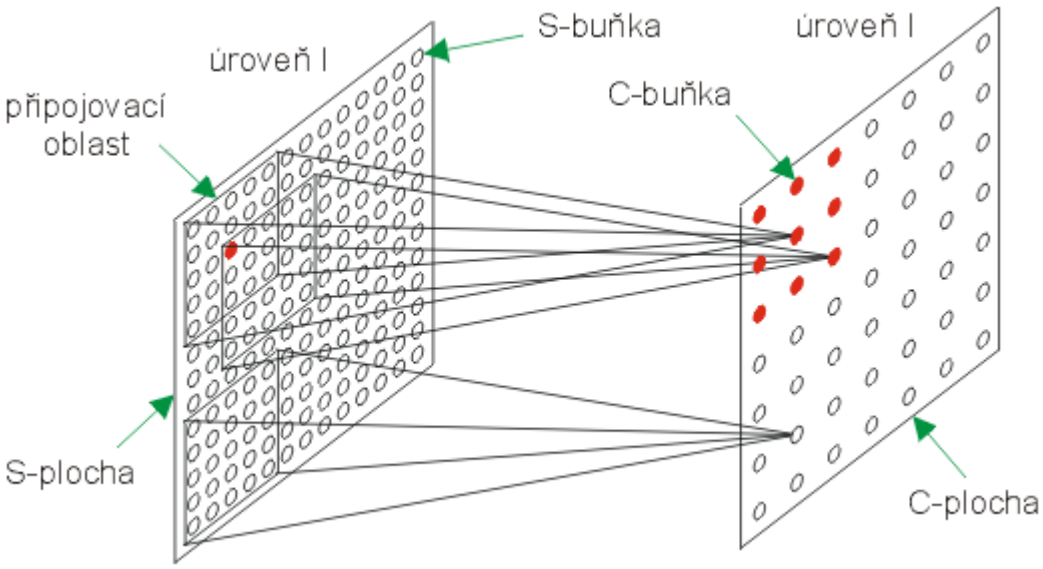
kde

u_V	je	buňka v V-vrstvě
u_C	je	buňka v C-vrstvě nebo vstupní vrstvě
l	je	pořadové číslo vrstvy
n	je	pořadové číslo buňky v ploše
κ	je	pořadové číslo plochy ve vrstvě
v	je	pořadí buňky v připojovací oblasti
c	je	hodnota c-váhy (mezi vrstvou C a V nebo vstupní vrstvou a V)

Ze vztahů (3) a (4) je zřejmé, že by bylo možno oba výpočty spojit do jednoho a vrstvu V úplně vypustit. Na funkci sítě by se nic nezměnilo. Výhoda použití vrstvy V není matematická, ale algoritmická. Pro každou buňku vrstvy S se použije pouze jednou spočítaná hodnota. Pokud by se vrstva V nepoužila, musel by se výpočet opakovat tolikrát, kolik je ploch ve vrstvách S. Podrobněji např. v [7],[8],[9].

1.4.3 C-Vrstvy

Ve vrstvě S síť zjistila přítomnost určitého vzoru v nějakém místě vstupu. Vrstva C z těchto dat vytvoří informaci o tom, že daný vzor je blízko. Udělá to tak, že aktivuje buňky připojené k oblasti v níž se daný bod nachází. Situaci dobře vystihuje obrázek (Obr.8), kde je znázorněno propojení ploch z vrstvy S do C.



Obr. 8. Připojení buněk C-plochy na příslušnou S-plochu. Převzato z [6]

Každé ploše ve vrstvě S odpovídá jedna plocha ve vrstvě C. Vrstva C poskytuje v podstatě jakýsi rozmazaný zmenšený obraz vrstvy S.

K čemu je to dobré ?

Významy tohoto rozmazání jsou dva. První spočívá v tom, že síť je schopná akceptovat i drobné vzájemné posunutí vzorů. Druhý význam je v tom, že v každé C-vrstvě se síť zmenšuje až do konečného výsledku. Rozmazání nám nezpůsobí žádnou ztrátu ve zpracování, protože vrstva S již není interpretací nebo úpravou původního obrazu, ale jen jakousi mapou pravděpodobnosti výskytu nějakého vzoru.

Matematický popis výpočtu hodnoty v C-buňce

$$u_{Cl}(n,k) = \psi\left[\sum_{\kappa=1}^{K_{Sl}} j_l(\kappa,k) \cdot \sum_{v \in Dl} d_l(v) \cdot u_{Sl}(n+v,\kappa)\right] \tag{5}$$

kde

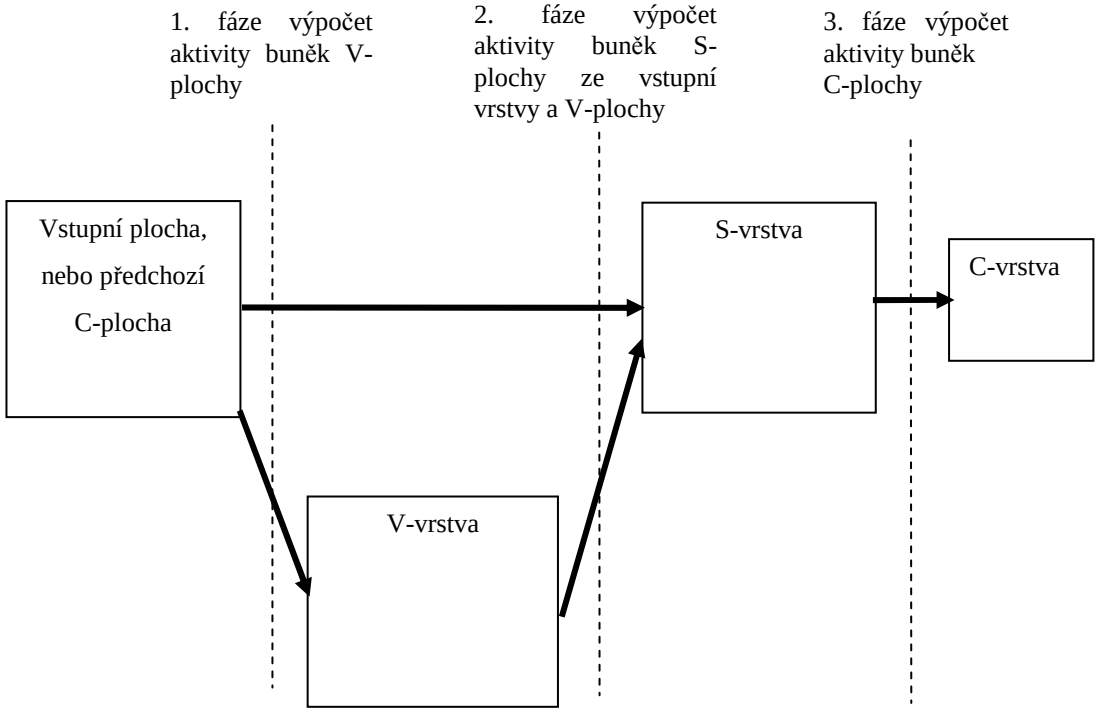
u_S	Buňka v S-vrstvě
u_C	Buňka v C-vrstvě nebo vstupní vrstvě
l	Pořadové číslo vrstvy
n	Pořadové číslo buňky v ploše
κ	Pořadové číslo plochy ve vrstvě

v	Pořadí buňky v připojovací oblasti
d	Hodnota d-váhy (mezi vrstvou S a C)
ψ	$\psi[x] = \frac{\varphi[x]}{1+\varphi[x]}$
φ	$\varphi[x] = \begin{cases} x \; pro \; x \geq 0 \\ 0 \; pro \; x < 0 \end{cases}$

Hodnota vah označená ve vztahu (5) písmenem d je většinou nastavena tak, aby se maximálně zesílil střed oblasti a méně okraje. Podrobněji např. v [7],[8],[9].

1.4.4 Spojení vrstev S,V,C

Výše uvedené popisy jsou přesným určením hodnot aktivit buněk v jednotlivých vrstvách. Pro názornější pohled na závislosti vrstev, uvádím ještě graficky jejich spojení v diagramu na obrázku (Obr. 9).

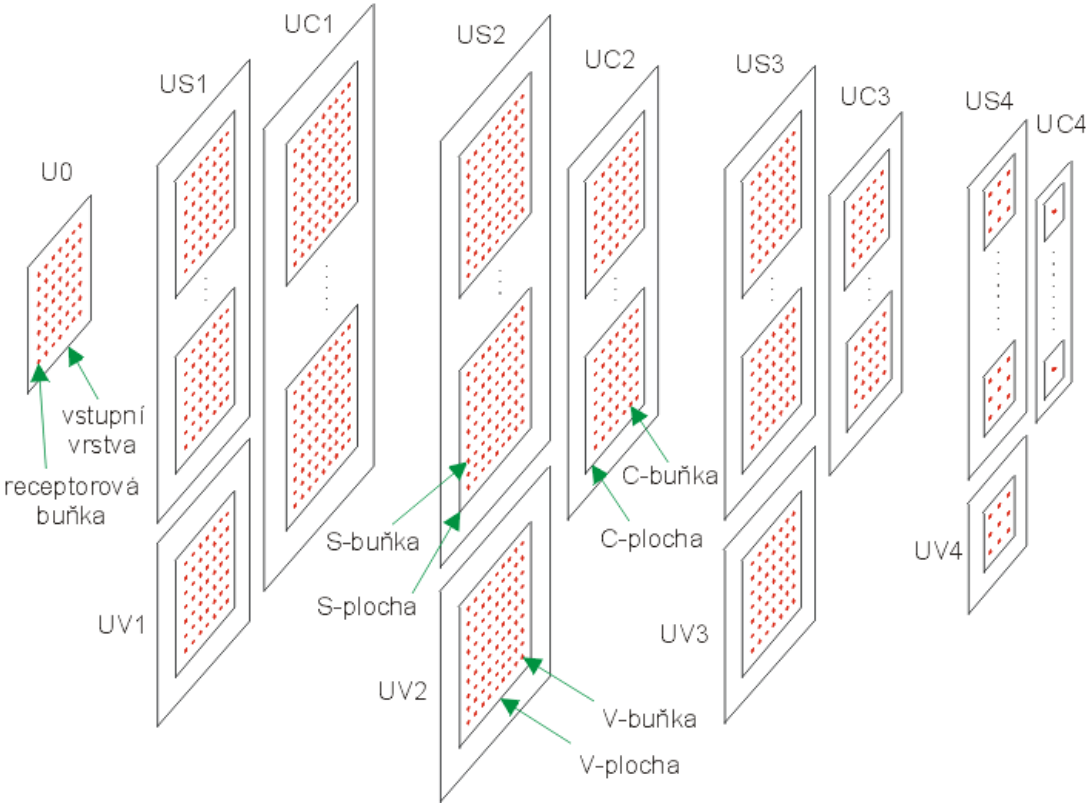


Obr. 9. Diagram postupu výpočtu jednotlivých typů vrstev sítě.

Obdélníky na obrázku (Obr. 9) znázorňují plochu typu V,S, nebo C a šipky znázorňují výstupy aktivit buněk dané plochy a jejich vstup do plochy následující. Postup je zobrazen ve třech fázích.

1. Výpočet aktivit V-buněk ze vstupní vrstvy.
2. Výpočet aktivit S-buněk ze vstupní vrstvy a z V-vrstvy.
3. Výpočet aktivity C-buněk z S-vrstvy.

Toto zapojení vrstev se následně opakuje v dalších úrovních sítě. Rozdíl mezi úrovněmi je takový, že se neustále zmenšuje velikost a počet jejich ploch. Největší počet buněk a vah má první úroveň sítě. Každá další vrstva má již počet buněk přibližně čtvrtinový.



Obr. 10. Příklad struktury ploch s buňkami v síti neocognitron. Převzato z [6]

1.5 Životní cyklus sítě

Životní cyklus sítě se skládá ze dvou fází. Fáze učení, ve které se síť naučí rozpoznávat příslušný vzor a fáze života, v níž pak síť tento vzor rozpoznává. Postupně popíši obě fáze.

1.5.1 Fáze učení

Síť neocognitron se příslušný vzor naučí rozeznávat podle jednoho jediného vzoru přivedeného na vstupní vrstvu sítě. Její učení je algoritmicky poměrně jednoduché a rychlé. Teoreticky by nemělo zabrat více času, než následné rozpoznávání.

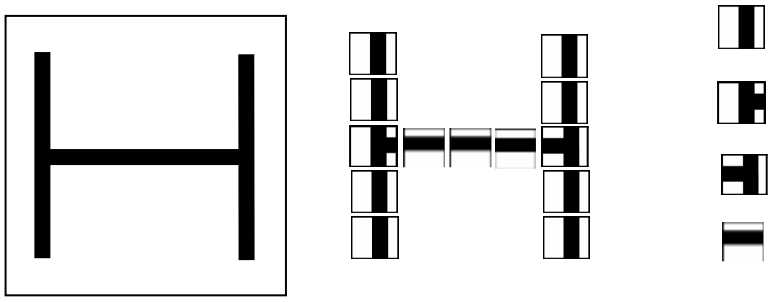
Síť neocognitron má čtyři typy vah označené jako váhy a, b, c, d. Z těchto vah jsou váhy c, d nastaveny napevno při vytvoření sítě a není třeba je přizpůsobovat. Váhy a, b se nastaví během učení a oba typy jsou ve vrstvě S. Více např. v [9],[10],[11].

Výběr oblastí vzoru pro učení první S-vrstvy

Nejprve na vstupní vrstvě nastavíme vzor, který se má síť naučit. Potom vybereme oblasti, ve kterých se nacházejí znaky mající identifikační význam pro daný objekt. Tuto činnost může vykonat osoba obsluhující program nebo mohou být oblasti vybrány nějakým vhodným algoritmem automaticky. Jednou z možností automatického rozdělení je rozdělit obraz na rovnoměrnou síť menších částí a každou vzniklou část použít jako jeden ze vzorů. Tento postup je možné bez problémů aplikovat při učení psaných znaků, ale komplikovanější je jeho použití u vzorů s členitým pozadím, které není součástí učeného vzoru. Z tohoto důvodu je třeba rozmyslet si princip určení s ohledem na následné využití sítě.

Vytvoření a nastavení vah první S-vrstvy

Tato část je poměrně jednoduchá. Ke každé části vzoru je vytvořeno pole vah se stejným rozměrem, jaký má tato část předloženého vzoru. Hodnoty vah v tomto poli jsou nastaveny na stejné hodnoty, jako je aktivita buněk v oblasti vzoru. V podstatě je část vzoru zkopírována do matice vah. Z této fáze vychází výsledná struktura sítě, protože počet ploch v první vrstvě je dán počtem oblastí učeného vzoru. Stejné pole vah se může ve vzoru vyskytnout několikrát. V takovém případě mu samozřejmě stačí pouze jedna S-plocha a jedno pole vah. Bohužel při rozpoznávání obličejů nikdy vzor takovou vlastnost mít nebude. Aby byla tato vlastnost dobře demonstrovatelná, ukážu ji na příkladu znaku „H“ na obrázku (Obr. 11), kde je ukázka rozložení znaku na fragmenty reprezentující jeho jednotlivé části.



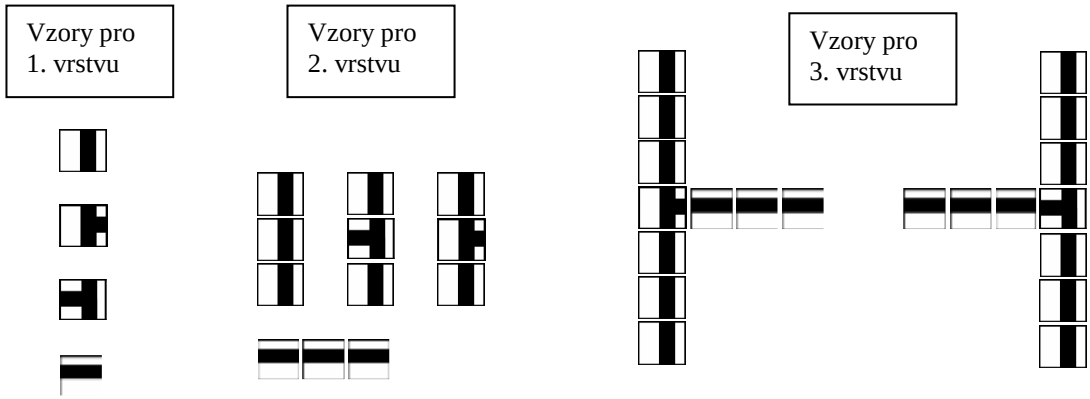
Obr. 11. Příklad charakteristických oblastí písmene H.

Z obrázku (Obr. 11) jsou zřejmé 4 charakteristické složky znaku „H“. Jedná se o svislou čáru, vodorovnou čáru a oblasti pravého a levého průsečíku. V první vrstvě sítě neocognitron by tedy mohly být 4 plochy a 4 pole vah reagující na tyto složky.

Nastavení vah pro výpočet následujících S a C vrstev

Po nastavení a-vah, můžeme přistoupit k výpočtu buněk první S-vrstvy. Nejprve přivedeme na vstupní vrstvu sítě takovou část vzoru, která svou komplexitou odpovídá druhé vrstvě. Jedná se většinou o složení několika vzorků z první vrstvy. Dále necháme spočítat hodnoty buněk první S-vrstvy a první C-vrstvy. V místě odpovídajícím místu se vzorem vstupní oblasti získáme ve vrstvě C odpovídající vzor. Tento použijeme jako vstupní vzor pro následující vrstvy. Akci opakujeme se všemi vzory s vyšší komplexností.

V dalších vrstvách postupujeme stejným způsobem se složenými vzory z předchozích vrstev až nakonec v poslední vrstvě učíme celý vstupní vzor. Příklad vzorů pro první, druhou a třetí vrstvu k naučení písmene „H“ z obrázku (Obr. 11) je vidět na obrázku (Obr. 12). Pro následující čtvrtou vrstvu by se již na vstup předkládalo celé písmeno „H“.



Obr. 12. Příklad sestavení vzorů k učení jednotlivých vrstev.

Z obrázku (Obr. 12) je patrné, že vzory pro 2. vrstvu jsou složeny ze vzorů první vrstvy a dále vzory pro třetí vrstvu jsou zase složeny ze vzorů pro druhou vrstvu.

Po nastavení vah poslední vrstvy je síť připravena k fázi života.

1.5.2 Fáze života

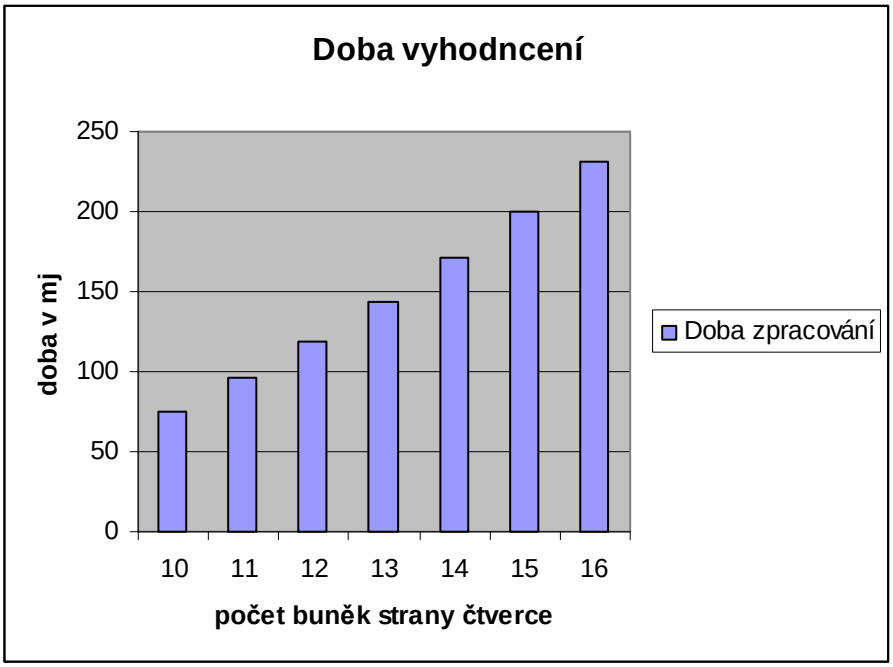
Fáze života je ta doba v existenci neuronové sítě, kdy je síť naučená a rozpoznává vstupní znaky. Rozpoznávání se děje hierarchicky od první vrstvy k poslední. Poslední vrstva dává výslednou hodnotu. Výhodou je, pokud je v poslední vrstvě pouze jedna buňka. V takovém případě nám aktivita této buňky určuje přímo do jaké míry odpovídá vstup naučenému znaku.

Vstupní vrstva

Není nutné, aby vstupní vrstva měla při vybavování stejný rozměr jako v době učení. Síť rozpozná vzor spolehlivě na jakkoli velké vstupní oblasti, potom ale narůstá doba rozpoznávání. Zároveň čím větší je vstupní oblast, tím větší jsou potom všechny následující vrstvy i poslední C-vrstva, z níž získáváme výsledek. Nárůst doby rozpoznání ukážu na příkladu:

Máme dvě vstupní plochy, obsahující hledaný vzor. První plocha má rozměr m bodů šířky a m bodů výšky. Celkový počet bodů ve vstupní vrstvě je m^2. Druhá plocha má n bodů šířky a n bodů výšky. Celkový počet bodů ve vrstvě je n^2 .Vyhodnocení vstupních vzorů probíhá pro každý bod první S-vrstvy, která musí rozměrově odpovídat vstupní vrstvě.

Rozdíl v době zpracování obou sítí tedy bude n^2 - m^2. Graficky je tento rozdíl znázorněn na obrázku (Obr. 13).



Obr. 13. Závislost doby vyhodnocení sítě na velikosti vstupní vrstvy.

V grafu si můžeme všimnout rychlosti nárůstu doby zpracování. Zatím, co pro vstupní vrstvu velikosti 10x10 bodů trvá výpočet 75 časových jednotek, tak pro síť 15x15 bodů je již doba zpracování 200 časových jednotek.

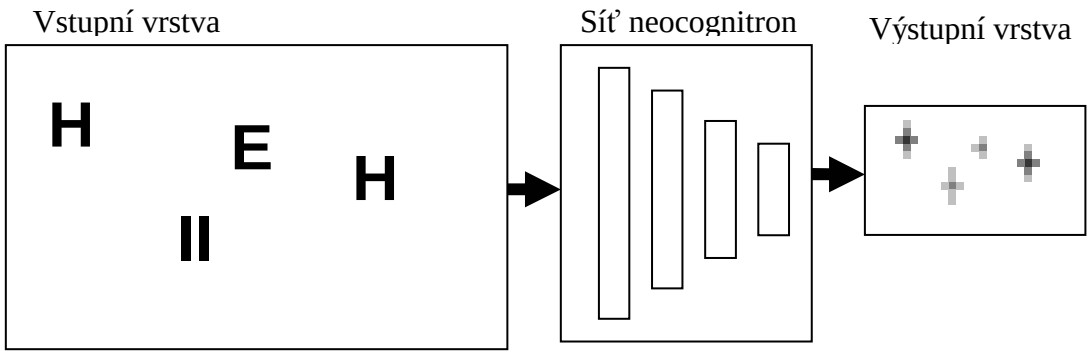
Výše uvedená časová závislost nás nutí k tomu, abychom se snažili dosáhnout takové konstrukce sítě, v níž bude co nejmenší vstupní vrstva.

Výstupní vrstva

Ve výstupní vrstvě (poslední vrstva typu C) se po postupném provedení výpočtů ve všech předchozích vrstvách nachází výsledek rozpoznávání. Pokud výstupní vrstvu tvoří jedna jediná buňka, pak je hodnota její aktivity zároveň mírou rozpoznání daného vzoru.

Pokud výstupní vrstvu tvoří více buněk, pak každá buňka určuje míru rozpoznání naučeného vzoru v určitém místě vstupní vrstvy. Správně rozpoznaných vzorů může být

více. Ukázka možného vstupu a výstupu sítě rozpoznávající vzor písmene „H“ je na obrázku (Obr. 14). Ve vstupní vrstvě se vyskytuje znak „H“ dvakrát a kromě něj i jiné znaky, které se mu částečně podobají. Ve výstupní vrstvě je aktivita buněk znázorněna odstínem šedé barvy. Čím tmavší buňka, tím vyšší aktivita.



Obr. 14. Ukázka možného vstupu a výstupu sítě neocognitron.

Na výstupní vrstvě v obrázku (Obr. 14) je vidět, že síť nezjišťuje pouze přítomnost, nebo nepřítomnost vzoru na vstupu, ale je možné z výstupu získat také informaci o tom, v kterém místě se vzor přibližně nachází a také kolik výskytů vzoru v ní je.

2 PRINCIPY ROZPOZNÁVÁNÍ OBLIČEJŮ

V této kapitole bych rád zmínil některé metody v současnosti používané k identifikaci osoby podle obličeje. Uvedené principy vycházejí z testování skutečných komerčních aplikací. Při testování existujících komerčních systémů se ukazuje, že rozpoznávání a identifikace obličejů je ovlivňováno řadou aspektů. Technické rušivé vlivy (změny osvětlení – stíny, pozadí scény, natočení, umístění v obraze, rotace,...) jsou relativně dobře normalizovatelné. Závažným problémem se však jeví změny v obličeji vlivem emočních výrazů a morfologických změn způsobených stárnutím.[1]

Automatizované systémy identifikace osob mohou být řešeny dvěma základními přístupy:

Strukturální - rozpoznávání jednotlivých dominantních částí obličeje (oči, ústa, nos...) předkládaného vzoru, změření antropometrických veličin, jejich normalizace vzhledem k předpokládaným rušivým vlivům (šum, rušení, poloha ve scéně, velikost...), porovnání s databází známých fotografií použitím klasifikačních algoritmů, statistické rozhodnutí o relativní podobnosti s takto vybranou množinou obrazů. [12]

Holistický - porovnání - identifikace vzorku pomocí globálních reprezentací opět s následným statistickým vyhodnocením relativní pravděpodobnosti. Příznačné pro tento přístup jsou kombinace metody backpropagation (metoda zpětného učení neuronové sítě), základní analýzy komponent (principal component analysis - PCA) a dekompozice jedinečných hodnot (singular value decomposition - SVD). Představa redukcionismu je obecná praxe v rozvoji inteligentních systémů - návrh řešení komplexních problémů prostřednictvím postupné dekompozice úkolu do následných modulů.[12]

2.1 Dynamické rozpoznávání z video sekvencí

Jde o metodu založenou na algoritmu USC (Univerzity of Southern California). Tento algoritmus byl nasazen v komerčních aplikacích, kde využívá technologii „jetů“. Jet je sada komplexních čísel vypočtená pro každý uzel mřížky proložené obličejem. Jety jsou základem pro vytvoření normalizované reprezentace obličeje použitím tzv. obličejového grafu. Přes region obličeje je proložen tzv. shlukový graf (Face Bunch Graph – FBG) s počtem 48 uzlů. Tyto body jsou rozloženy v hranách a křivkách dominantních částí obličeje.

Klasifikace může být provedena i ve škále barev s výpočtem váženého průměru a následně provedena klasifikace zkoumaného obrazu. Tímto je určena jednoznačná reprezentace zkoumaného obličeje a vyhodnocení proti databázi známých obličejů přiřazením bodového hodnocení shody od nejlepšího (100) po nejnižší (0). Systém má určité problémy proložit graf obličejem při bočním pohledu. Rozpoznání obličeje působí určité potíže, neboť technologie je založena na principu změn odstínu barev v po sobě následujících snímcích video sekvence, což znamená, že obličej musí být v pohybu buď příčně přes obraz nebo přibližován zoomem. Dalším zjištěným problémem je nízká pravděpodobnost rozpoznání morfologicky změněného obličeje, popřípadě emočně změněného obličeje, který není obsažen v databázi známých vzorů. [1]

2.2 Statické rozpoznávání z fotografií

Tato metoda využívá stejného matematického základu jako dynamické rozpoznávání s tím rozdílem, že mřížka – elastic graph (srovnávací elastický graf) – je rozložena rovnoměrně přes oblast obličeje.

Aplikace automaticky prokládá mřížku obličejem podle tří referenčních bodů – středů očí a úst. V případě, že body byly určeny chybně, je možné provést ruční korekci a znovu vytvořit tzv. „Phantomas Graf“. Pro co možná nejpřesnější vyhodnocení je nutné provést ruční korekci natočení obličeje podle osy Z (oči do vodorovné pozice). Dalším zjištěným problémem, stejně jako u předchozí aplikace, je relativně nízká pravděpodobnost rozpoznání morfologicky nebo emočně změněného obličeje. Obecně lze říci, že je reprezentace obrazových vlastností pomocí takto definované mřížky relativně úspěšnou; je nezávislá na rozumné míře posunutí, rotace a měřítku. Lokální změny v obraze se projeví v lokálních změnách jeho reprezentace.[1]

2.3 Geometrická reprezentace nosu a očí

Její princip je založen na geometrické reprezentaci obličeje v malé oblasti kolem očí a nosu. Tento princip využívá aplikace Imagis (Imagis Cascade Technologies Inc.). Systém vychází z předpokladu, že právě v této oblasti je možné určit jednoznačnou geometrickou

reprezentaci obličeje, neboť geometrické parametry jsou zde určovány pevnými tkáněmi – kostmi – jak vyplývá z anatomie lebky. Výhodou je zjednodušení a zrychlení výpočtu, určitá míra nezávislosti na maskování obličeje (vousy, brýle) a normalizace na rozumnou míru rotace obličeje.

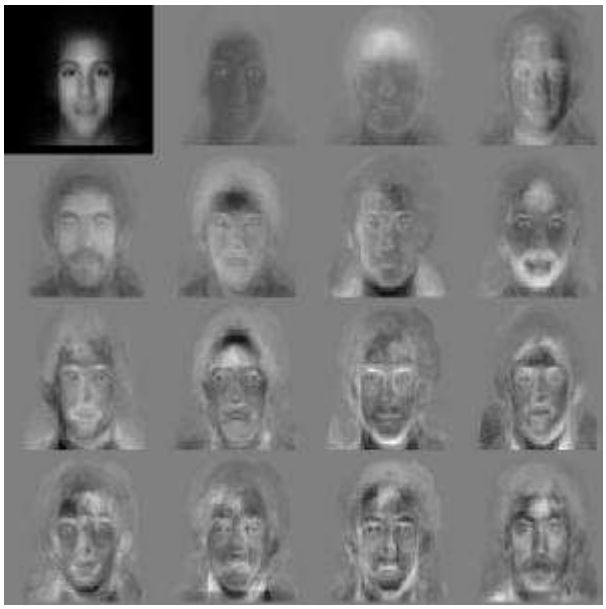
Zjištěným problémem, stejně jako u předchozí aplikace, je nízká pravděpodobnost rozpoznání morfologicky či emočně změněného obličeje.[1]

2.4 Systém založený na metodě 3D morfologického modelu

Vytváření 3D modelů je poměrně nákladná metoda pořízení galerie autorizovaných osob. Využívá se při tom 3D laserový scaner. Je však natolik zajímavá, že stojí zato ji zmínit. Lidská tvář je deformovanou plochou v 3D prostoru. Tato metoda je založena na morfingu a tzv. fittingu („lícování“) – deformaci tohoto modelu obličeje, který zakóduje tvar a strukturu v rámci parametrů modelu a na algoritmu, který obnoví tyto parametry z jednotlivého obrazu obličeje. Databáze známých vzorů obličejů se vytváří 3D snímačem, nebo aproximací fotografií z několika úhlů pohledu obličeje (například trojdílné policejní fotografie). Pro identifikaci obličeje je z modelu použit tvar a texturové parametry, které jsou odděleny od obrazových parametrů, jako je poloha a osvětlení.[1]

2.5 PCA – Základní komponentní analýza

PCA využívá vektorů tváře odvozených s kovarianční matice pravděpodobnostní distribuční funkce k vytvoření šablony vhodné pro srovnávání. Každá tvář lze rozdělit na tzv. eigenfaces (vzory tváří - matice jasových úrovní) a poté jde opět složit viz. (Obr. 15). Každá eigenface je reprezentována pouze číslem, takže se namísto obrázku ukládá pouze číslo. [7]



Obr. 15. Standardní eigenfaces používané pro rozložení obrazu. Převzato z [7]

2.6 LDA – Lineární diskrétní analýza

LDA je metoda, kdy se třídí pořízené obrazy tváří do skupin. Cílem je maximalizovat rozdíly mezi jednotlivými skupinami a minimalizovat rozdíly v každé skupině, každý blok snímků reprezentuje jednu třídu viz obrázek (Obr. 16).[7]

Obr. 16. Příklad šesti tříd užitím LDA. Převzato z [7]

K samotné analýze je použita metoda EBGM. Byla vyvinuta proto, že předešlé metody nemohou uvažovat nelineární charakteristiky jako je osvětlení okolí, pozice hlavy anebo výraz tváře (úsměv, zamračení). Na obličeji se definuji uzlové body, které se poté propojí a tím definují linie tváře v prostoru, vznikne tím souřadnicová síť obličeje. Samotné

rozpoznávání pak probíhá tak, že systém pomocí filtru uzlových bodů reaguje na jednotlivé snímané tváře a může je následně porovnávat a vyhodnocovat. Problémem je přesnost lokalizace orientačních bodů na tváři. [7]

3 ANALÝZA POUŽITELNOSTI NEOCOGNITRONU

V kapitole 2 byly uvedeny některé metody používané pro identifikaci obličejů. V zásadě pro žádnou z nich není vyloučeno použít k některým fázím analýzy nějaký druh neuronové sítě. Proto jsem neuvedl použití sítě neocognitron jako možnou metodu identifikace. Tato architektura by se dala využít i ve spojení s jinými uvedenými.

3.1 Výhody sítě neocognitron pro identifikaci

Hlavní výhodou architektury neocognitron je její schopnost rozpoznávat i posunuté, částečně zdeformované, pootočené, zašuměné, zmenšené a zvětšené vzory. I po takovýchto změnách je síť schopná naučený vzor rozlišit od vzorů jiných.

Další možnou výhodou v době moderních vícejádrových procesorů je snadná paralelizovatelnost výpočtu i na úrovni rozpoznávání jednoho vzoru.

Při správném nastavení všech parametrů může síť pracovat naprosto nezávisle na okolí a bez podpory uživatele. Existuje totiž ve variantě pro učení s učitelem i ve variantě samoorganizující. Nutno však dodat, že varianta využívající princip učení s učitelem má vyšší rozpoznávací schopnost.

3.2 Nevýhody sítě neocognitron pro identifikaci

Za největší nevýhodu sítě neocognitron pro rozpoznávání obličejů považuji stejnou vlastnost, která je zároveň její největší výhodou. Tím myslím schopnost rozpoznávat deformované vzory. Rozdíl mezi dvěma lidskými obličeji se může zdát dostatečně velký, ale pokud připustím nižší míru podobnosti u vstupních fragmentů, kterou potřebuji pro správnou identifikaci změněného výrazu ve tváři nebo pootočené hlavy, pak může lehce dojít k záměně i méně podobných obličejů.

Dalším rizikem tohoto použití je citlivost na větší změny rozměrů. Síť neocognitron celkem bez problémů akceptuje menší změny měřítka zkoumaného objektu. Čím je ale změna měřítka větší, tím více se snižuje hodnocení, takže pak opět hrozí záměna s jinou tváří.

3.3 Časová složitost

Síť neocognitron se vyznačuje velkým počtem ploch a buněk, ale také velkým počtem vah. Váhy v síti jsou sice sdílené, ale výpočet nad nimi stejně probíhá. Nejvíce buněk a vah je v první vrstvě. Proto bude tato pro zjištění doby trvání nejzvýznamější. K zjištění časové složitosti výpočtu potřebujeme znát pro jednotlivé vrstvy počet jejich buněk a počet vah.

Pro zjednodušení můžeme předpokládat, že jednotlivé vrstvy i trénovací vzory budou mít čtvercový tvar. Dále zavedeme pravidlo, že rozměr plochy vrstvy C bude vždy poloviční proti rozměru vrstvy S se stejným pořadím. Teoretický případ takové sítě je diagram (Obr. 17).

V diagramu jsou použity tyto symboly:

proměnná	Význam
n	Rozměr pole buněk
w	Rozměr pole vah
v	Počet ploch ve vrstvě
p	Počet váhových vektorů ve vrstvě

Směr šipek v diagramu znázorňuje postup výpočtu aktivit buněk v jednotlivých plochách první vrstvy. Časovou složitost vyjádříme hodnotou udávající počet výpočtů potřebných pro zjištění aktivity buněk v celé ploše a označíme ji písmenem S. Dílčí složitosti, ze kterých je složená pak písmenem S s indexem V, S, nebo C udávajícím typ vrstvy .

Složitost první vrstvy

S_V - Složitost vrstvy V bude:

$$S_V = w^2 \cdot n^2 \tag{6}$$

S_S - Složitost vrstvy S bude:

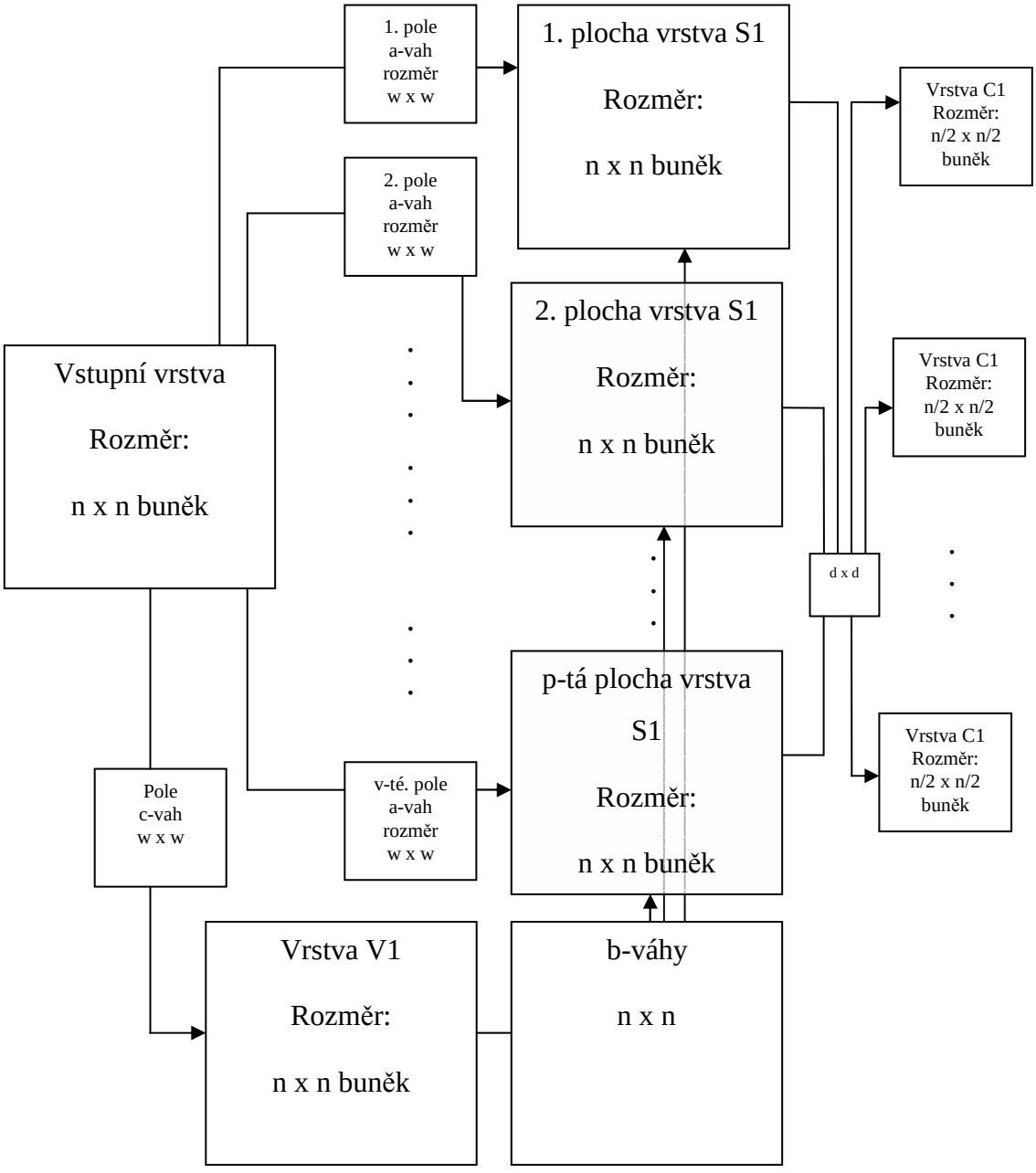
$$S_S = p \cdot (w^2 \cdot n^2) + n^2 \tag{7}$$

S_C - Složitost vrstvy C bude:

$$S_C = p \cdot \left(d^2 \cdot \left(\frac{n}{2} \right)^2 \right) = \frac{p \cdot (d \cdot n)^2}{4} \tag{8}$$

S – Celková časová složitost výpočtu jedné vrstvy pak bude:

$$S = S_V + S_S + S_C \tag{9}$$



Obr. 17. Typická struktura spojení vrstev V,S a C.

Z uvedených vztahů vyplývá, že největší složitost je soustředěna do vrstvy S. Zde je složitost přibližně p-krát větší než u vrstvy V a nejméně 4-krát větší než u vrstvy C.

U skutečné sítě je rozdíl mezi složitostí vrstev C a S ještě vyšší, protože velikost vektoru d-vah bývá obvykle výrazně menší než velikost vektoru a-vah. Také je zřejmé, že s počtem charakteristických rysů poroste složitost lineárně, zatímco s jejich rozměrem a rozměrem vstupní oblasti poroste kvadraticky.

Složitost následujících vrstev

V následujících vrstvách je situace podobná, jen s tím rozdílem, že velikost vstupních ploch odpovídá velikosti výstupních ploch plochy předcházející. Pokud tedy vrstva C předchozí vrstvy má rozměr n/2, potom rozměr vrstvy S v následující vrstvě je také n/2.

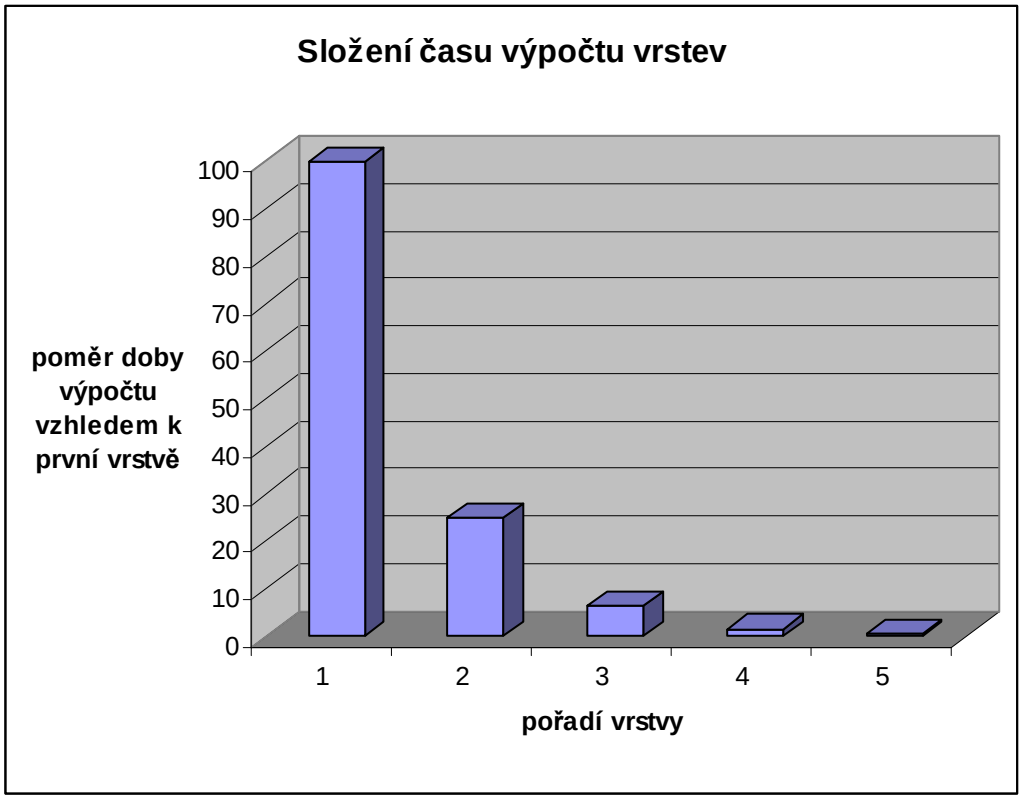
Ze vztahů pro výpočet první plochy pak plyne vztah pro každou následující vrstvu:

$$S_x = \frac{S_{(x-1)}}{4} \tag{10}$$

Následně celková složitost výpočtu sítě N s k vrstvami bude:

$$S_N = \sum_{x=1}^{k} \frac{S_x}{4^{(x-1)}} \tag{11}$$

Ze všech výše uvedených vztahů plyne závěr, že největší vliv na dobu výpočtu sítě bude mít první vrstva typu S. Doba výpočtu bude určena nejvíce velikostí vstupní oblasti a velikostí výukových fragmentů předloženého vzoru. Výpočet každé vrstvy zabere vždy čtvrtinový čas výpočtu předchozí vrstvy. Přehledněji je časová závislost vrstev vidět v grafu na obrázku (Obr. 18)



Obr. 18. Doba výpočtu v jednotlivých vrstvách sítě neocognitron.

3.4 Paměťová složitost

Pokud jde o paměťové nároky, tak v případě této sítě nebudou příliš velké. V paměti bude nutno udržet hodnoty aktivit buněk a hodnoty vah. Vzhledem k tomu, že váhy jsou společné pro všechny buňky v ploše, bude i paměťová náročnost malá a bude odpovídat prostému součtu všech buněk a vah.

Vztah pro výpočet paměťových nároků jedné vrstvy bude:

$$M_V = n^2 + w^2 \tag{12}$$

$$M_S = p \cdot \left(w^2 + n^2\right) + n^2 \tag{13}$$

$$M_C = p \cdot n^2 + d^2 \tag{14}$$

$$M = M_V + M_S + M_C = (p+2)\cdot n^2 + (p+1)\cdot w^2 + d^2 \tag{15}$$

Je vidět, že paměťová náročnost poroste lineárně se zvětšováním oblasti nebo počtu výukových fragmentů.

Paměťová náročnost pro celou síť N velikosti k vrstev pak bude:

$$M_N = \sum_{x=1}^{k} M_x \tag{16}$$

3.5 Shrnutí poznatků o složitosti sítě

Paměťová náročnost sítě neocognitron nebude pro program významná. Narůstá lineárně se zvětšováním vstupní plochy. Závažnější bude časová složitost, která narůstá rychleji a to v závislosti na počtu a velikosti učených fragmentů obrazu a velikosti vstupní plochy. I v tomto případě je nárůst lineární, ale 4 x strmější.

Vzhledem k tomu, že jak paměťová, tak časová náročnost roste s velikostí sítě lineárně, bude s největší pravděpodobností úloha algoritmicky dobře řešitelná.

II. PRAKTICKÁ ČÁST

4 IMPLEMENTACE ROZPOZNÁNÍ OBLIČEJŮ

V této části se budu věnovat konkrétní implementaci sítě neocognitron ve vlastní aplikaci pro osobní počítač. Vytvořím návrh programu a návrh struktury sítě pro rozpoznávání obličejů osob. Dále popíši možná rizika při použití této metody a navrhnu možnosti optimalizace.

4.1 Požadavky na program

4.1.1 Funkční požadavky

1. Umožnit přidání osoby k rozpoznatelným osobám.
2. Umožnit odstranění osoby z rozpoznávaných osob.
3. Při zjištění osoby před kamerou tuto identifikovat a zobrazit další informace.

4.1.2 Nefunkční požadavky

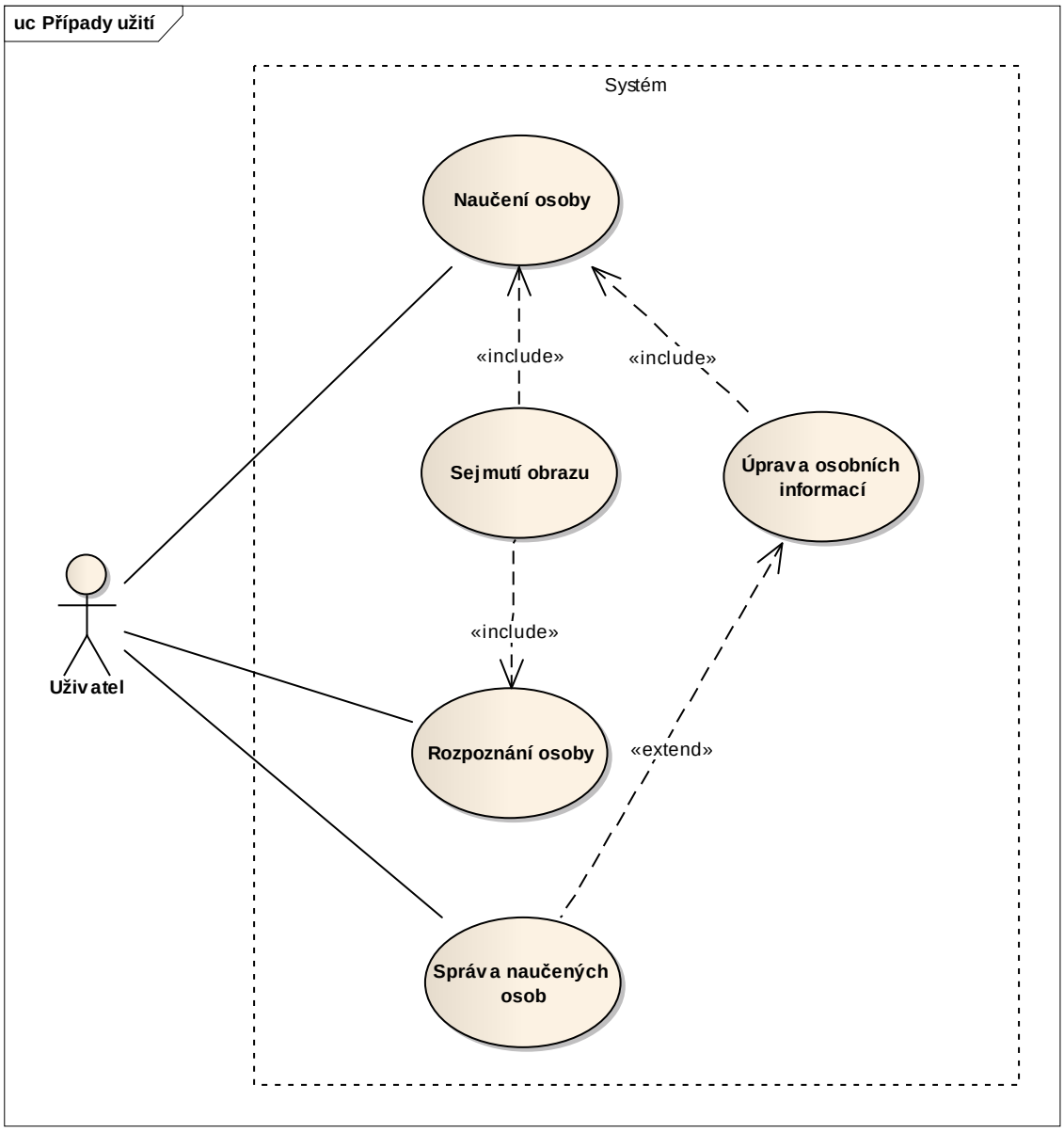
1. Možnost jednoduchého přenosu naučené osoby na jiný počítač.
2. Identifikaci provést v rozumném čase. Maximálně 0,5 sekundy na porovnání s jednou osobou v databázi.
3. Jednoduché uživatelské rozhraní.
4. Jednoduchá instalace.

4.2 Případy užití

V této kapitole popíši případy užití a jejich jednotlivé scénáře. Jedná se o popis interakce programu s uživatelem, proto zde nebudou podrobnosti týkající se přímo rozpoznávání a implementace sítě neocognitron.

4.2.1 Diagram USE-CASE

V diagramu USE-CASE na obrázku (Obr. 19) je znázorněna interakce programu s uživatelem. Tento diagram by měl shrnovat celkovou funkcionalitu programu viditelnou uživatelem. Program je složen ze čtyř základních funkčností znázorněných v elipsách v diagramu.



Obr. 19. USE-CASE diagram k navrhovanému programu.

Popis scénářů jednotlivých USE-CASE:

Naučení osoby

1. Provede se UC sejmutí obrazu.
2. Program zobrazí poslední sejmutý obrázek osoby.
3. Uživatel v obrázku označí obličej právě učené osoby.
4. Provede se UC úprava osobních informací.
5. Uživatel potvrdí uložení informací.
6. Program vytvoří neuronovou síť, naučí ji aktuální osobu, doplní osobní informace a data sítě uloží.

Úprava osobních údajů

1. Program se dotáže uživatele na základní osobní informace:
   Jméno, příjmení, adresu e-mail, fotografii.
2. uživatel požadované informace zadá.
3. Program uloží osobní informace.

Rozpoznání osoby

1. Uživatel spustí rozpoznávání.
2. Provede se UC sejmutí obrazu.
3. Systém postupně načte všechny uložené sítě neocognitron a provede pomocí nich výpočet nad sejmutým obrazem.
4. Systém vezme síť, která dávala nejlepší výsledek a pokud je tento vyšší než minimální požadovaná hodnota pro odlišení osoby, pak zobrazí informace o uživateli.

Správa naučených osob

1. Program zobrazí seznam dosud naučených osob.
2. Uživatel může v seznamu jednotlivé osoby označovat.
3. Systém vždy při označení osoby zobrazí její osobní informace.
4. Uživatel může osobu ze seznamu odstranit.

Sejmutí obrazu

1. Uživatel zaujme takovou polohu, aby kamera snímala celý jeho obličej.
2. Systém zobrazuje aktuální snímaný obraz a průběžně jej aktualizuje.
3. Podle zobrazeného sejmutého obrazu uživatel upravuje svou polohu.

4.3 Implementace

4.3.1 Volba programovacího jazyka

Pro implementaci programu jsem si zvolil jazyk C#, proto i funkce a třídy popsané dále odpovídají syntakticky tomuto jazyku. Přiložený zdrojový kód je také v jazyku C#.

Proč C# ?

Z principu funkce sítě neocognitron bylo zřejmé, že pro vyhodnocení vzoru bude nutné velké množství dílčích matematických operací. Celkový počet operací vychází z počtu vrstev, ploch, buněk a vah v síti viz. kapitola 3. Odhadovaný počet výpočtů pro síť byl v řádu 10^7 výpočtů pro jedno hodnocení. Od počátku tedy bylo zřejmé, že optimální pro tvorbu sítě bude takový jazyk, který poskytne nejrychlejší kód. V tomto směru by se zřejmě jevil nejvhodnější jazyk ASSEMBLER, protože produkuje přímo strojový kód procesoru. Na druhou stranu nejlepší programovací jazyk na řešení většiny problémů je ten, který programátor nejlépe ovládá. Takže jsem rozhodování zúžil na jazyky C++,C#, Visual Basic a Java.

Pro rozhodnutí, který jazyk zvolit, jsem si vytvořil jednoduchý programový cyklus o 50 milionech průchodech, který uvnitř neustále přepočítával průměrnou hodnotu svých

pořadových čísel. Používal operace sčítání, násobení a dělení na hodnotách s pohyblivou desetinou čárkou.

Z vyhodnocení tohoto postupu (který mohl být ovlivněn i jinými faktory než jen jazykem) vyšlo, že v jazyku C++ a C# byl běh přibližně stejně rychlý, v jazyku Visual Basic asi o 30% pomalejší a v jazyku Java asi o 50% pomalejší.

Zbývalo rozhodnutí mezi C# a C++. Vzhledem ke stejným výsledkům rychlostního testu jsem zvolil opět jazyk, který lépe ovládám, tedy C#.

4.3.2 Funkční celky a časový harmonogram vývoje

Program se skládá ze tří částí podle účelu a podle harmonogramu vytvoření:

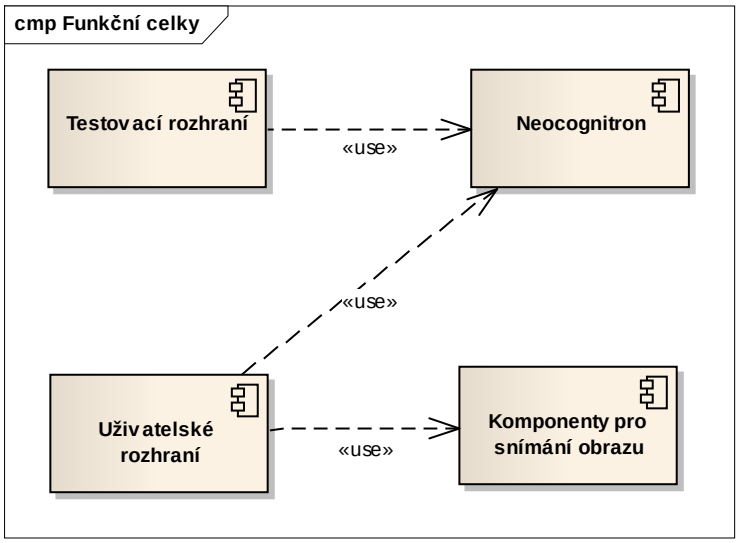
1. Funkce sítě neocognitron a její testovací rozhraní.
2. Funkce spojené s obsluhou digitální kamery.
3. Uživatelské rozhraní.

Blokové schéma programu je znázorněno v diagramu komponent na obrázku (Obr. 20).

V první fázi byla vytvořena funkčnost potřebná pro vytváření, učení a běh sítě neocognitron.

Ve druhé fázi byla vytvořeny funkce pro sejmutí obrazu z kamery a předání ke zpracování.

V poslední fázi jsem vytvářel uživatelské rozhraní. Uživatelské rozhraní využívá třídy WinForms knihovny .NET a stejně jako ostatní části bylo tvořeno v prostření MS Visual Studio 2008. Uživatelské rozhraní není příliš rozsáhlé, jak plyne z diagramu USE-CASE a jeho scénářů.

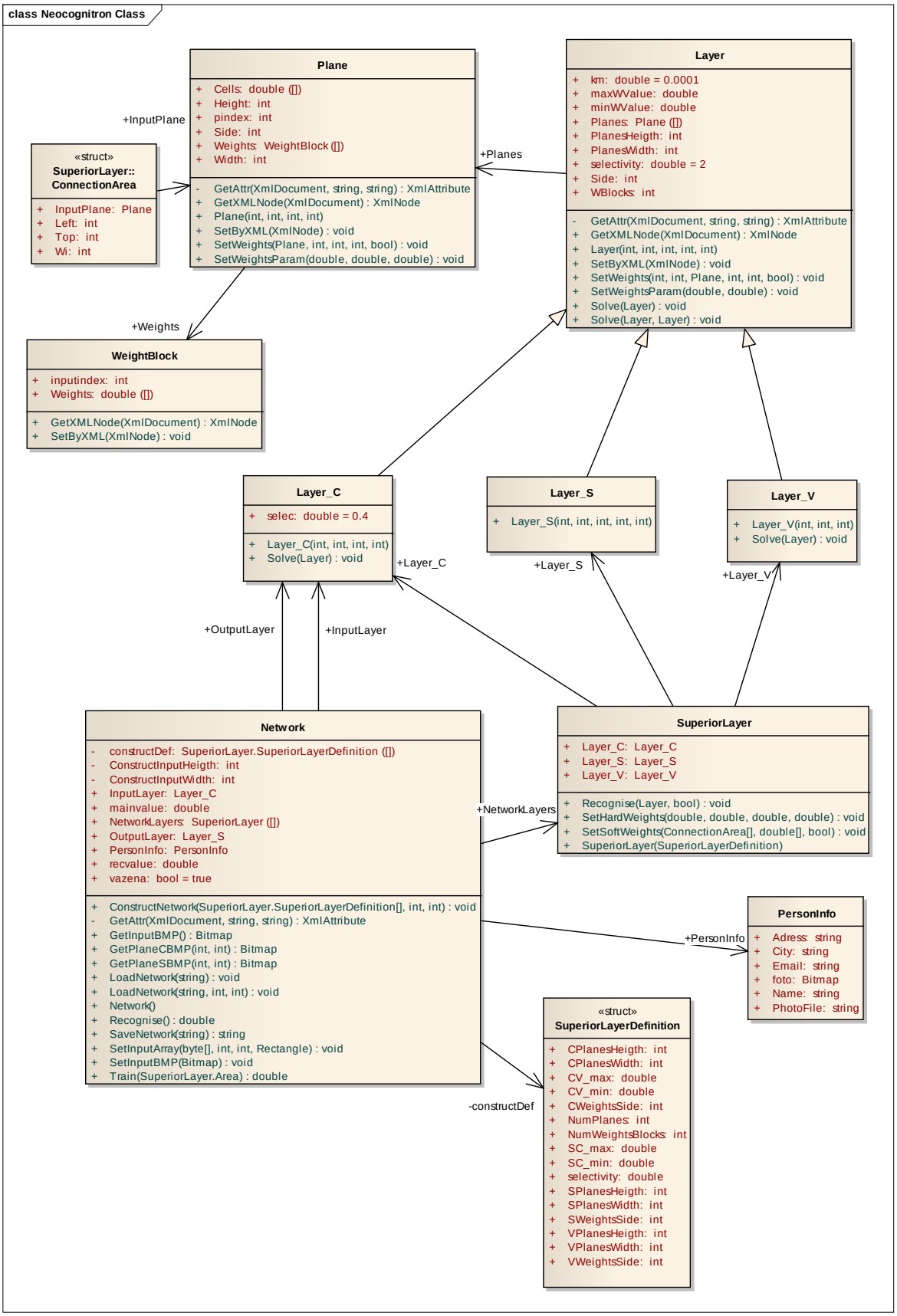


Obr. 20. Diagram komponent

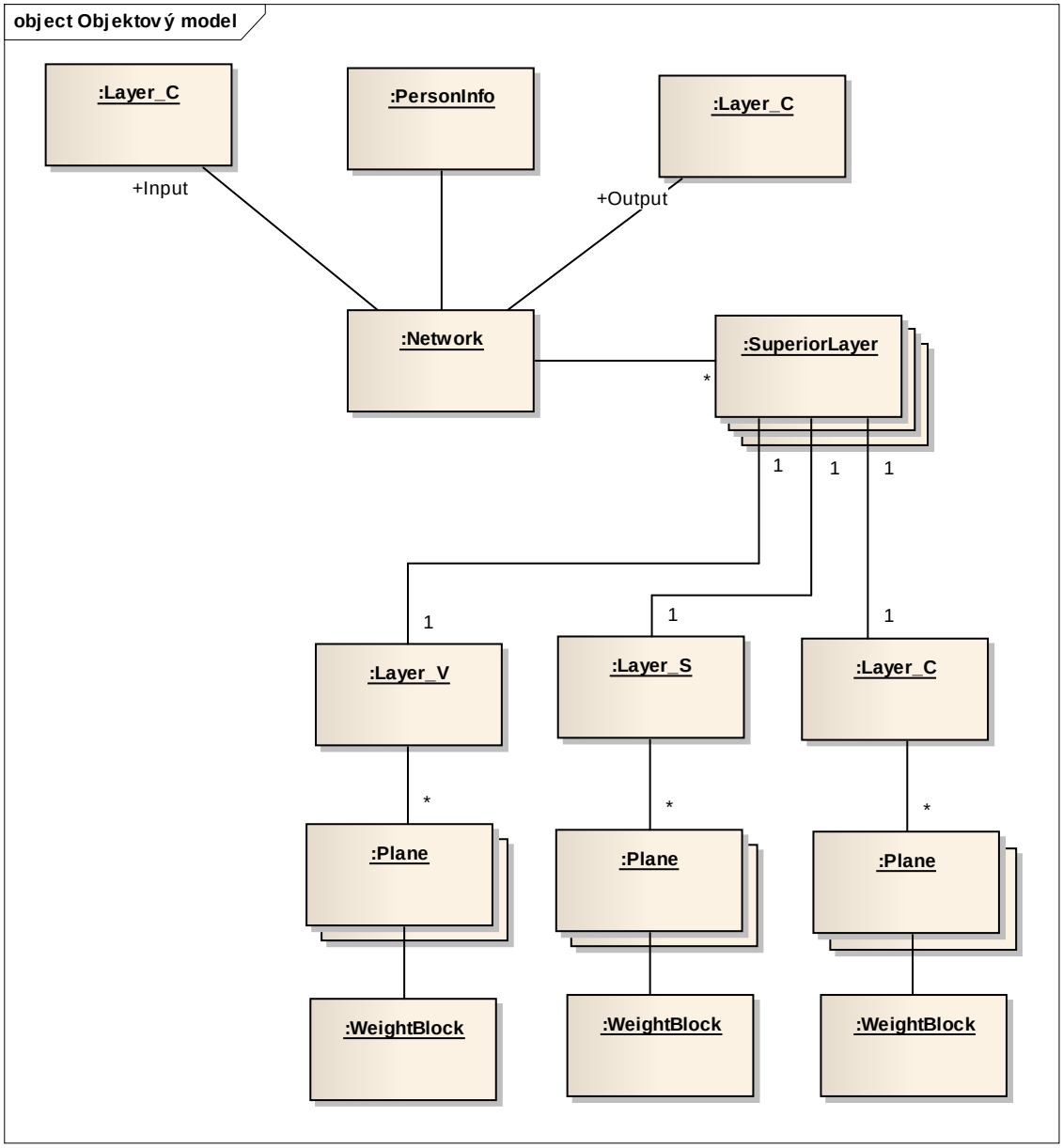
4.3.3 Objektová struktura implementace sítě neocognitron

Diagram tříd (Obr. 21) ukazuje z jakých částí se skládá komponenta **`Neocognitron`** z diagramu komponent (Obr.20). Pro konstrukci sítě a napojení na okolí slouží třída **`Network`**. Třída network reprezentuje neuronovou síť celkově. V ní se nacházejí dva veřejné odkazy na vstupní a výstupní vrstvu v podobě třídy **`Layer_C`**. Dále obsahuje kolekci tříd typu **`SuperiorLayer`**. Jednotlivé instance třídy **`SuperiorLayer`** reprezentují jakési nadvrstvy. Každá tato nadvrstva obsahuje třídy **`Layer_V`, `Layer_S`** a **`Layer_C`**.

Každá z uvedených tříd **`Layer_V`,`Layer_S`** a **`Layer_C`** je potomkem třídy **`Layer`**. Třída **`Layer`** zajišťuje vše, co se týká konkrétní vrstvy. Obsahuje také kolekci ploch této vrstvy typu **`Plane`**. V třídě **`Layer`** jsou implementovány základní funkčnosti. Nastavení vah, výpočet aktivity buněk ve svých plochách na základě vstupu. Třída **`Plane`** zapouzdřuje konkrétní plochu ve vrstvě.



Obr. 21. Diagram tříd sítě neocognitron.

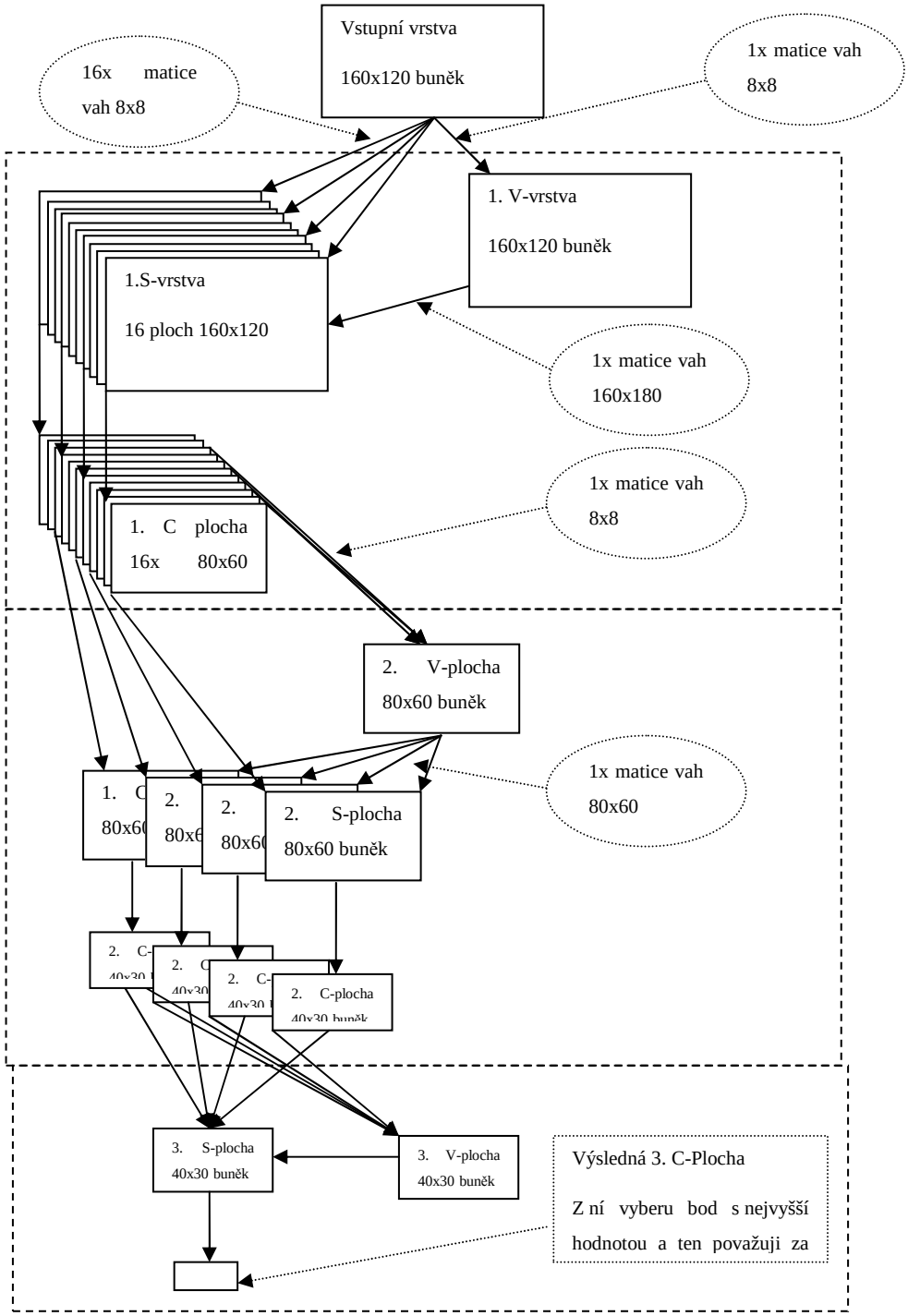


Obr. 22. Diagram instancí sítě neocognitron.

Diagram instancí (Obr. 22) zpřehledňuje stromovou strukturu instancí jednotlivých tříd ve vybudované síti neocognitron. Stejným způsobem je pak organizováno ukládání stavu sítě viz. Kapitola 4.3.7 Ukládání a obnova sítě.

4.3.4 Struktura použitá v mé implementaci

Pro mou implementaci jsem zvolil pevnou strukturu sítě pro všechny naučené osoby. Struktura se skládá ze vstupní vrstvy, tří nadvrstev a výstupní vrstvy. Rozpis prvků v jednotlivých vrstvách ukazuje diagram (Obr. 23).



Obr. 23. Struktura neocognitron použitá pro implementaci.

Počty buněk v plochách diagramu (Obr. 23) jsou uvedeny jen jako přibližný příklad. Jejich skutečný počet se teoreticky může lišit při každém vyhodnocení, jak bylo uvedeno v kap. 1.5.2. V průběhu testování jsem tyto hodnoty různě měnil, abych zjistil, jak velký vliv mají na činnost programu.

4.3.5 Vybudování sítě

Před tím, než bude možné síť naučit a použít, je třeba vytvořit její instanci. Pro vytvoření instance sítě je ale zapotřebí dost velké množství parametrů, určujících její strukturu. Proto zavádím strukturu nazvanou **`SuperiorLayerDefinition`**. V této struktuře se soustředí všechny parametry pro vytvoření jedné nadvrstvy (SuperiorLayer) v síti. K vytvoření instance celé sítě pak postačí zavolat funkci `ConnstructNetwork()` s parametrem typu **`SuperiorLayerDefinition[]`**. Funkce postupně vytváří jednotlivé prvky **`SuperiorLayerDefinition`** a v konstruktoru předává definice dále.

4.3.6 Učení sítě

Síť vytvořená funkcí `ConnstructNetwork()` má již nastaveny hodnoty vah c a d. Tyto váhy se nastavují na pevnou hodnotu při konstrukci sítě a učením se již nedají ovlivnit. Zbývá nastavit váhy, které učení podléhají.

Průběh učení:

1. Do vstupní vrstvy se nastaví analyzovaný obraz. Musí mít stejný rozměr jako vstupní vrstva. Proto je dobré před voláním `ConnstructNetwork()` znát rozměry analyzovaného obrazu.
2. Zavolá se funkce `Train()` v síti. Tato funkce rozdělí oblast vzoru na malé fragmenty a postupně je naučí první vrstvu. Následně se převezme výsledek z první vrstvy C a výpočet postupuje k další vrstvě. Spouští se postupně v jednotlivých vrstvách sítě a nastavuje jejich váhy. Po nastavení všech vah program provede vyhodnocení vstupního učeného vzoru jako vstupu přes celou síť a vrátí jako výsledek nejvyšší aktivitu buňky v poslední vrstvě C.
3. Po naučení program uloží stav sítě. Ten je určen počtem ploch v jednotlivých vrstvách sítě a velikostí a počtem bloků vah v síti. Pro stav sítě nejsou potřeba

rozměry ploch a buňky v plochách, ani hodnoty pevných vah. Jediné co je potřeba pro funkci sítě jsou hodnoty nastavitelných vah a její struktura (počty vrstev, počet vzorů v jednotlivých vrstvách atd.).

4.3.7 Ukládání a obnova sítě

Stav sítě se ukládá do textové struktury XML. Každá třída, která obsahuje část, kterou je třeba ukládat a obnovovat, má implementovánu metodu `GetXMLNode()` a metodu `SetByXML()`. Metoda `GetXMLNode()` vytvoří větev XML dokumentu a vrátí ji jako výstup. Volající objekt pak větev zahrne do své struktury a tu buď opět předá volajícímu, nebo (v případě třídy **`Network`**) celou strukturu uloží do souboru. Metoda `SetByXML()` dělá přesný opak. Převezme větev XML dokumentu a podle hodnot a atributů nastaví síť.

4.3.8 Vybavování

Při potřebě identifikovat obličej v obraze systém postupně načítá uložené parametry sítě pro jednotlivé osoby, podle načtených uložených parametrů vytvoří instanci sítě a na její vstupní vrstvu vloží analyzovaný obraz.

Poté zavolá v instanci sítě metodu `Recognise()`, ta provede kaskádovitě výpočet jednotlivých vrstev podobně jako tomu bylo při učení sítě. Nakonec ve výstupu opět vrátí nejvyšší aktivitu buňky v poslední vrstvě C. Poměr této hodnoty a uložené hodnoty získané metodou `Train()` určuje poměr shodnosti s naučeným vzorem.

Ze všech naučených sítí program vybere tu, která vrátila nejvyšší hodnotu. Pokud tato hodnota přesahuje minimální požadovanou podobnost, pak je obličej považován za identifikovaný.

Při výpočtu hodnot aktivit buněk probíhá několik do sebe vnořených cyklů. Doba potřebná pro vybavení sítě se vstupní vrstvou 160 x 120 bodů je sumarizována v tabulce (Tab. 1). Tabulka ukazuje dobu v počtu provedených výpočtů pro jednotlivé vrstvy navržené sítě s procentuálním vyjádřením části časového úseku připadajícího na jednotlivé nadvrstvy.

Tab. 1. Rozdělení časů potřebných pro výpočet v jednotlivých vrstvách sítě.

	Počet ploch	velikost matice vah	počet buněk v ploše	Počet operací	Celkem	v %
1. V-Vrstva	1	64	19200	1228800		
1.S-Vrstva	16	64	19200	19660800		
1.C-Vrstva	16	25	4800	1920000		
1.S-Vrstva z plochy V	1	1	19200	19200	22828800	93%
2. V-Vrstva	1	64	4800	307200		
2.S-Vrstva	4	64	4800	1228800		
2.C-Vrstva	4	25	1200	120000		
2.S-Vrstva z plochy V	1	1	4800	4800	1660800	7%
3. V-Vrstva	1	64	1200	76800		
3.S-Vrstva	1	64	1200	76800		
3.C-Vrstva	1	9	300	2700		
3.S-Vrstva z plochy V	1	1	1200	1200	157500	1%
Celkem:					24647100	

Z hodnot uvedených v (Tab. 1) plyne, že pro jedno vybavení sítě je třeba provést téměř 25 milionů výpočtů. V této původní variantě s obrazem velikosti 160 x 120 program dosahoval rychlost vyhodnocení přibližně 450ms. Vzhledem k tomu, že i tato doba byla ještě příliš dlouhá, přistoupil jsem ke zmenšení plochy obrazu na velikost 107 x 80. Touto změnou nedošlo k výraznému zhoršení schopnosti rozpoznávání, ale doba na vyhodnocení jednoho obličeje klesla na 120ms.

4.4 Snímání obrazu

Ke snímání obrazu jsem použil existující komponentu vytvořenou pro MS Visual Studio. Tato komponenta má z mého pohledu několik nevýhod a dvě z nich jsou pro mě důležité. První spočívá v tom, že komponenta pro přenos obrazu z kamery do programu používá clipboard systému Windows. Druhá vážná nevýhoda je nemožnost nastavení rozlišení snímaného obrazu.

Použitím clipboardu je znemožněno využívat jej při běhu programu jiným aplikacím a dále to zpomaluje získání obrazu.

Nemožnost nastavení rozlišení snímaného obrazu lze poměrně dobře obejít jeho dodatečnou transformací. Tato funkce ale opět zapírá čas systému a navíc může způsobit ztrátu informace v obraze. Komponenta sice disponuje funkcí k nastavení rozměru. Tato funkce správně posílá zprávu systému Windows, ale nastavení se neprojeví. Může to být problém kamery, kterou jsem měl k dispozici nebo problém ovladačů. Tato kamera je již staršího data výroby.

Nicméně i přes tyto vlastnosti byla komponenta použitelná pro práci programu. Doba pro sejmutí obrazu se v celkovém čase příliš neprojeví a při testování se bez schránky obejdu. Obraz dodávaný kamerou je v rozměru 120x120 bodů. Tento rozměr nelze změnit, ale je dostačující, protože jej transformací převádím na rozměr menší.

4.4.1 Sejmutí obrazu

Snímání probíhá ve dvou fázích:

1. Sejmutí obrazu. Tato akce je vyvolána časovačem každých 300ms. Po sejmutí nového obrázku je tento porovnán s obrazem sejmutým před ním. Porovnává se jasová intenzita všech bodů ve všech barevných složkách.
2. Předzpracování obrazu. V této fázi se obraz transformuje na požadovanou velikost, převede se do odstínů šedé a vyváží se do rozsahu od černé do bílé.

Porovnání sejmutého obrazu s předchozím ad.1 probíhá podle vzorce:

$$L = \sum_{y=0}^{w-1} \sum_{x=0}^{h-1} \left(\left| l_r(x,y) - l'_r(x,y) \right| + \left| l_g(x,y) - l'_g(x,y) \right| + \left| l_b(x,y) - l'_b(x,y) \right| \right) \qquad (17)$$

kde

L je celkový rozdíl dvou obrazů

l je bod v prvním porovnávaném obraze

l’ je bod v druhém porovnávaném obraze

r je označuje červenou složku barvy

g je označuje zelenou složku barvy

b je označuje modrou složku barvy

x je sloupec v obraze, ve kterém se nachází porovnávaný bod

y je řádek v obraze, ve kterém se nachází porovnávaný bod

w je šířka obrazu v počtu bodů

h je výška obrazu v počtu bodů

Výsledkem L je číslo udávající celkový rozdíl jasových složek všech bodů v obrazech. Pokud tento rozdíl překročí stanovenou mez, je indikováno, že došlo k zásadní změně v obrazu. Tato změna vyvolá požadavek na vyhledání obličejů v obraze.

Předzpracování obrazu ad. 2.

Při požadavku na vyhledání obličeje je obraz nejprve upraven. Úprava spočívá v převzorkování na rozměr 107 x 80 bodů. To je provedeno pomocí vestavěné funkce třídy **`Bitmap`** knihovny .NET. Převzorkovaný obraz je poté převeden na stupně šedi. Při převodu na odstíny šedé nerespektuji vnímání lidského oka a nepreferuji žádnou barevnou složku před jinou. Domnívám se, že by tím docházelo k větší ztrátě informace než prostým zprůměrňováním barevných složek. Je pravda, že takto vytvořený černobílý obraz není na pohled tak hezký, ale nikde se nezobrazuje a programu pro vyhodnocení to nevadí. Ukázka původního sejmutého obrazu a obrazu převedeného na stupně šedé je na obrázku (Obr.24)

a) sejmutý obraz

b) transformovaný obraz

Obr. 24. Ukázka sejmutého a transformovaného obrazu.

Hodnoty jednotlivých bodů v transformovaném obraze jsou uloženy v rozsahu hodnot <0,1>, kde 0 znamená černá a 1 znamená bílá. Po převodu je ještě provedeno vyvážení za účelem zvýšení kontrastu obrazu.

Vyvážení probíhá tak, že je v obraze zjištěna minimální l_{min} a maximální l_{max} hodnota a poté jsou všechny body přepočítány podle výrazu:

$$l' = \frac{(l - l_{\min})}{l_{\max} - l_{\min}} \tag{18}$$

kde

l' je výsledná hodnota jasové úrovně bodu

l je původní hodnota jasové úrovně bodu

l_{min} je nejnižší hodnota jasové úrovně bodu v obraze

l_{max} je nejvyžší hodnota jasové úrovně bodu v obraze

Nový obraz dosáhne maximálního kontrastu bez ztráty informace. Tento obraz je již předán k analýze.

4.5 Uživatelské rozhraní

V této kapitole si kladu za cíl popsat všechna okna programu sloužící k interakci s běžným uživatelem. Okno pro testování a nastavování parametrů bude uvedeno v kapitole věnující se testování.

4.5.1 Popis úvodní obrazovky a nastavení programu

Po spuštění programu se uživateli zobrazí úvodní obrazovka. Úvodní obrazovka obsahuje hlavní menu pro spouštění jednotlivých funkcí programu, stavový řádek pro přehled o aktuálně prováděné akci, prostor okna je použit pro zobrazení náhledu obrazu snímaného kamerou a zobrazení naposled identifikované osoby.

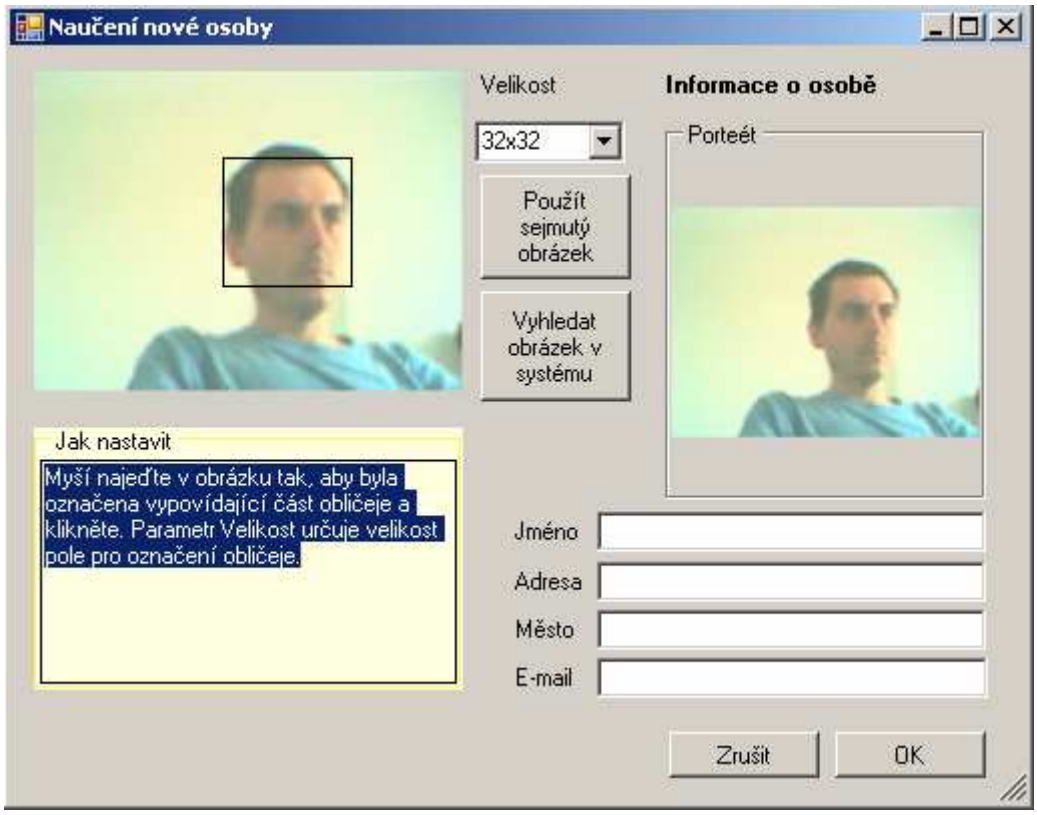
Pohled na úvodní obrazovku je na obrázku (Obr. 25.)

Obr. 25. Úvodní obrazovka programu.

4.5.2 Naučení osoby

Funkce naučení osoby se spustí položkou „Učení“ v menu „Program“.

Program zobrazí dialogové okno pro naučení nové osoby (Obr. 26). Okno obsahuje naposled sejmutý obraz, informační oblast s popisem požadované činnosti a dále textová pole pro zadání osobních údajů osoby pro naučení. Dvě tlačítka uprostřed slouží ke změně fotografie osoby.



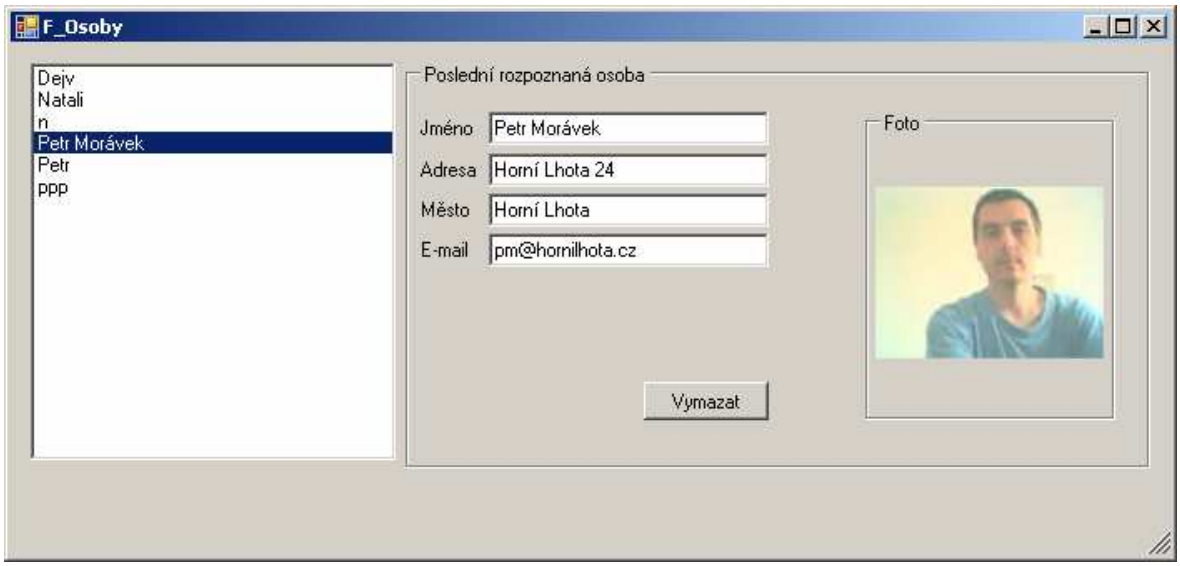
Obr. 26. Okno pro učení nové osoby.

V okně pro učení nové osoby (Obr. 26), je třeba myší označit na sejmutém obraze oblast, kde se nachází obličej. Oblast by neměla obličej přesahovat. Pokud je velikost čtverce příliš velká nebo malá, je třeba zvolit jiný rozměr. Pokud jsou všechny údaje nastaveny, můžeme spustit učení tlačítkem „OK“. To provede vytvoření sítě, její naučení a uložení.

4.5.3 Správa naučených osob

Na osoby, které se již program naučil, se můžeme podívat přes menu „Nastavení“ – „Osoby“. Tato volba nám zobrazí okno pro správu osob. Zde se zobrazí seznam naučených osob, ve kterém je možno listovat. Při kliknutí na řádek seznamu se v pravé části zobrazí osoba příslušející k tomuto záznamu a její osobní údaje. Uživatel zde může změnit informace o osobě, nemá už ale možnost zasáhnout do parametrů rozpoznávání.

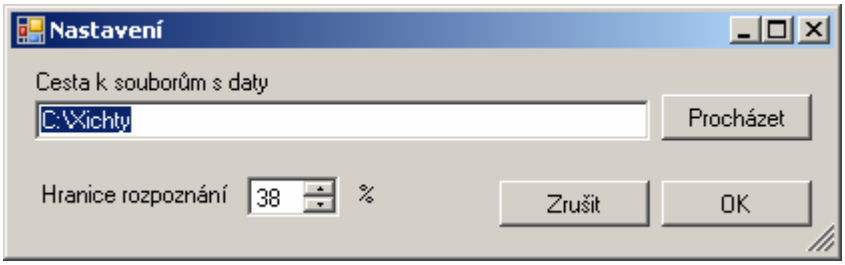
Pohled na okno pro správu osob je na obrázku (Obr. 27)



Obr. 27. Pohled na okno pro správu osob.

4.5.4 Identifikace osoby

Rozpoznávání se spouští z hlavního menu „Program“ volbou „Běh“. Průběh rozpoznávání je vidět na hlavní obrazovce. Úroveň přesnosti pro rozpoznání můžeme ovlivnit parametrem shodnosti. Najdeme jej v menu „Nastavení“ volba „Prostředí“. Hodnota určuje, kolik procent shody musí obrázek vykazovat, aby mohl být považován za rozpoznanou osobu. Pohled na okno nastavení je na obrázku (Obr. 28.).

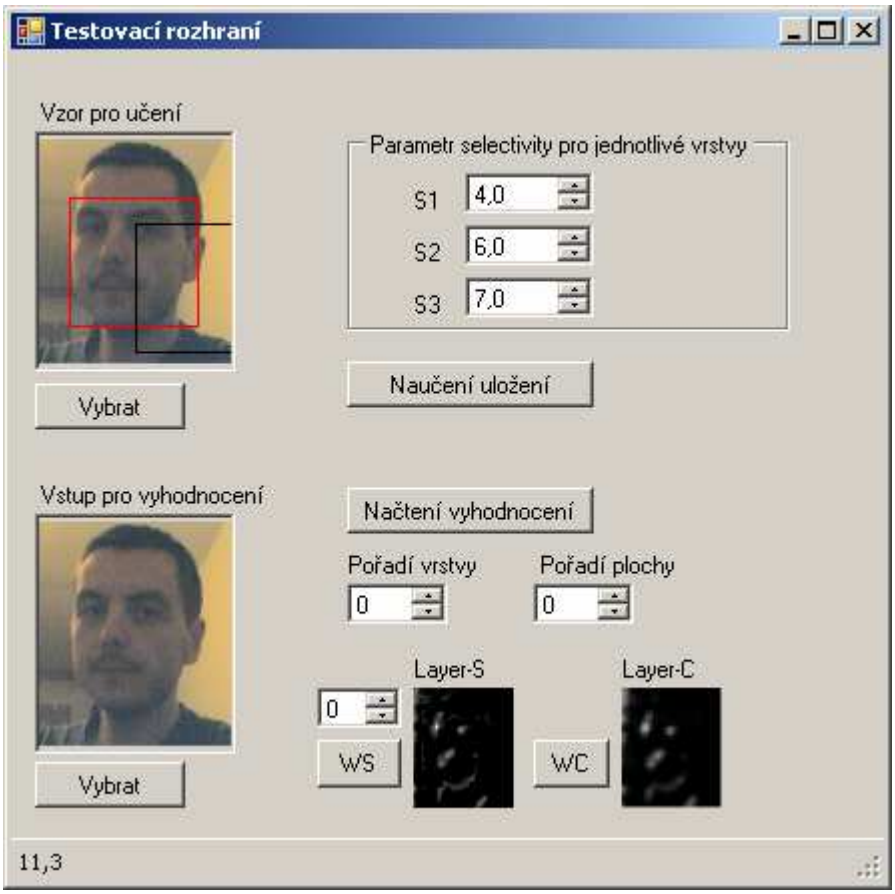


Obr. 28. Okno s parametrem hranice rozpoznání.

Pokud je spuštěn režim identifikace, pak systém neustále snímá obraz z kamery a podrobí jej srovnání se všemi naučenými vzory. Jestliže v některém obraze nalezne příslušný vzor, zobrazí na hlavní obrazovce související informace a údaj o stupni shody vyjádřený v procentech.

5 TESTOVÁNÍ A VÝSLEDKY IMPLEMENTACE

Jednotlivé třídy a jejich funkce byly v průběhu vývoje postupně testovány na syntaktickou a algoritmickou správnost a na schopnost dávat předpokládaný výsledek. Zcela zásadní funkcí tohoto projektu je funkce pro naučení a rozpoznání osoby pomocí neocognitron. Tato funkce byla vytvářena jako první. Vzhledem k tomu, že ještě nebylo hotovo uživatelské rozhraní a také by se k tomuto účelu příliš nehodilo, vytvořil jsem si pro testování a sledování vlivu jednotlivých parametrů samostatné testovací rozhraní (Obr. 29), které jsem poté nechal přístupné i v menu uživatelského rozhraní pro studijní účely.



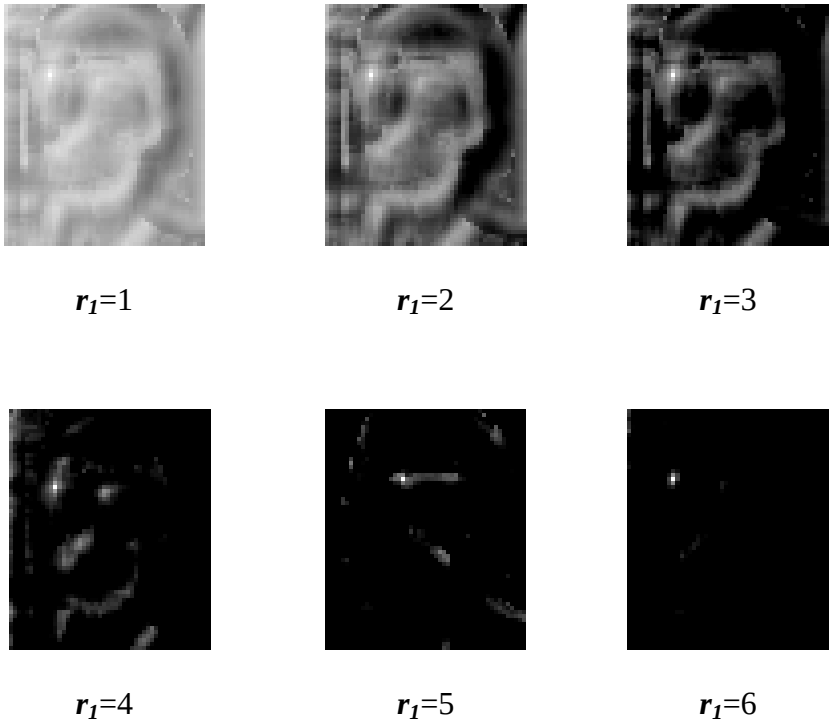
Obr. 29. Testovací rozhraní pro sledování mezivýsledků.

Testovací rozhraní nepoužívá vstup z digitální kamery, ale pouze dodané statické obrázky. Tato vlastnost lépe vyhovovala pro otestování předkládaných vzorů.

Na obrázku (Obr. 29.) je vidět pohled na testovací okno po naučení vzoru. Vstup obrázku pro naučení vzoru je z obrázku s nadpisem „Vzor pro učení“. Tlačítko pod obrázkem slouží k jeho výměně za jiný. Parametry selektivity vpravo od vzoru slouží k nastavení parametrů selektivity (ve vzorcích označován r_l , kde l je pořadové číslo vrstvy) v jednotlivých vrstvách sítě. Tyto parametry ovlivňují to, jak výrazně se podobnost vstupu projeví.

Změna parametru selektivity

Zvyšování této hodnoty vede k tomu, že síť aktivuje pouze buňky, které mají na vstupu přesnější obraz vzoru. Naopak snižování těchto parametrů vede k aktivaci buněk, které vzoru odpovídají méně. Pro optimální fungování je třeba parametry nastavit na takovou hodnotu, která ještě pozná vstupní vzor, ale zároveň jej odliší od jiných. Ukázka vlivu parametru selektivity na výstup je vidět na (Obr. 30).



Obr. 30. Ukázka vlivu parametru selektivity.

Obrázek (Obr. 30) ukazuje celkem šest vizualizovaných výstupů první vrstvy sítě. Hodnota parametru selektivity použitá pro daný výstup je rovna číslu pod každým obrázkem.

Vizualizace je provedena tak, že hodnota aktivity buňky odpovídá určitému odstínu šedé barvy, přičemž buňky s aktivitou 0 mají barvu černou a buňky s aktivitou 1 mají barvu bílou.

Z vizualizace je vidět, že v případě r_1=1 by pravděpodobně síť za podobný vzor považovala téměř cokoliv. Naopak při r_1=6 by již jakákoli si minimální změna způsobila nerozpoznání vzoru.

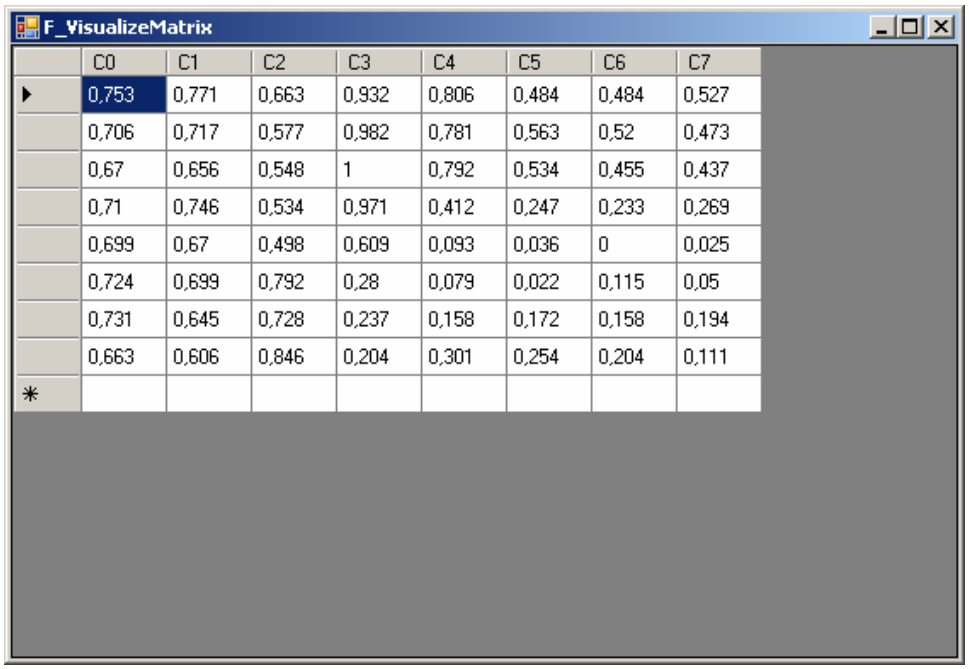
Testovací rozhraní jsem používal zejména k optimálnímu nastavení parametrů selektivity jednotlivých vrstev. V této implementaci je mám nastaveny konstantou. Připouštím, že se v jiných podmínkách mohou lišit. Proto bych rád do příští verze zahrnul i jejich případné automatické nastavování, například podle poměru výsledných nízkých a vysokých hodnot aktivit buněk nebo podle jejich průměru.

Nastavení parametrů selektivity je v této verzi programu následující:

$$r_1=4, \quad r_2=6, \quad r_3=7$$

Sledování mezivýsledků

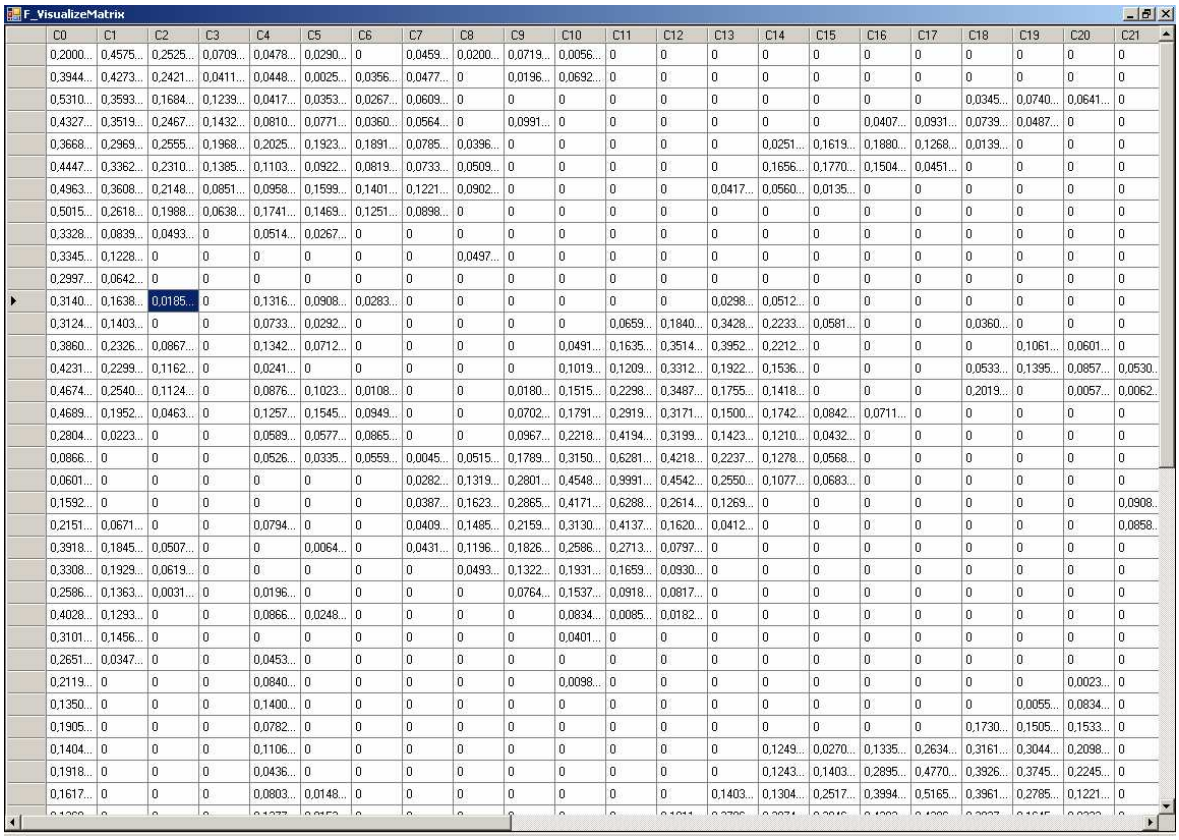
Kromě vizualizace aktivit buněk a nastavování parametrů jsem potřeboval také ověřit, zda se správně nastavují váhy v jednotlivých vrstvách sítě. K tomuto účelu jsem použil zobrazení hodnot v tabulce. Tabulku s hodnotami příslušných vah (Obr.31) vyvolá tlačítko „WS“ pro váhy buněk z vrstvy S nebo „WC“ pro váhy buněk z vrstvy C v testovacím rozhraní.



	C0	C1	C2	C3	C4	C5	C6	C7
▶	0,753	0,771	0,663	0,932	0,806	0,484	0,484	0,527
	0,706	0,717	0,577	0,982	0,781	0,563	0,52	0,473
	0,67	0,656	0,548	1	0,792	0,534	0,455	0,437
	0,71	0,746	0,534	0,971	0,412	0,247	0,233	0,269
	0,699	0,67	0,498	0,609	0,093	0,036	0	0,025
	0,724	0,699	0,792	0,28	0,079	0,022	0,115	0,05
	0,731	0,645	0,728	0,237	0,158	0,172	0,158	0,194
	0,663	0,606	0,846	0,204	0,301	0,254	0,204	0,111
*								

Obr. 31. Zobrazení hodnot vah.

Další užitečnou schopností, pomocí které jsem mohl sledovat chování, je vizualizace hodnot aktivit všech buněk v tabulce obrázek (Obr.32). Tato se nevyvolá tlačítkem, ale dvojklikem na obrázek s výstupem plochy S. Pro plochu C jsem si testovací výstup netvořil, protože plocha C je jen rozmazáním a zmenšením plochy S, takže mi postačovala obrazová informace.



Obr. 32. Zobrazení aktivity všech buněk jedné plochy vrstvy S.

Celkově shrnuto mi testovací rozhraní posloužilo k odladění jednotlivých funkcí sítě, sledování jejího chování a odezvy na různé předložené vzory a k empirickému zjištění nejvhodnějších parametrů konstant selectiviy pro jednotlivé vrstvy.

5.1 Rychlost

Testování rychlosti probíhalo také v testovacím rozhraní. Při provedení akce učení sítě nebo vybavování se po jejím skončení zobrazilo hlášení s dobou trvání akce v milisekundách.

Po doladění parametrů a optimalizaci provádění cyklů jsem se s dobou pro vyhodnocení jednoho obrázku dostal na 120~130 milisekund. To je ale pouze samotná doba vyhodnocení jednoho snímku. Celá doba zpracování jednoho hodnotícího cyklu je složena z několika dílčích časů.

T_S – Čas potřebný k sejmutí obrázku z kamery a převedení do formátu použitelného pro vybavování sítí.

T_L – Čas potřebný pro načtení uložených parametrů sítě a vytvoření její instance.

T_V – Čas potřebný pro vybavování sítě.

T – Celkový čas pro identifikaci osoby.

Celkový čas pro identifikaci z počtu k osob je potom dán vztahem:

$$T = Ts + k \cdot \left(T_L + T_V\right) \tag{19}$$

Při testování na počítači s jednojádrovým procesorem Celeron 2,4 GHz, paměť 2GB a operačním systémem Windows 7 jsem v konečné fázi dosahoval časů viz. (Tab. 2)

Tab. 2. Doba trvání jednotlivých částí výpočtu.

Veličina	Doba trvání
T_S	90 ms
T_L	70 ms
T_V	130 ms

Výsledný čas T při 10ti uložených tvářích v databázi činil:

T = 90 + 10 (70+130) = 2090 ms

5.1.1 Faktory ovlivňující rychlost učení

Učení sítě probíhá přivedením obrazu na vstup sítě a postupným nastavováním vah a vybavováním v dalších vrstvách. Celkový počet výpočtů je ve výsledku stejný jako při vybavování sítě. Při učení je jen jedna akce, která ve vybavování neprobíhá a to je samotné nastavení hodnot vah. Tato akce ale nezabere příliš mnoho času.

Rychlost naučení tváře by bylo možné bez změny struktury sítě ovlivnit jen změnou velikosti vstupního obrázku.

Rychlost učení není kritickou částí systému. K učení dojde jen jednou a není příliš důležité zda bude trvat 200 nebo 300 ms.

5.1.2 Faktory ovlivňující rychlost rozpoznávání

Stejně jako v případě učení, je nejdůležitějším faktorem ovlivňujícím vybavování velikost vstupní plochy. Tu by bylo možno ovlivnit lokalizací obličeje před samotnou akcí identifikace. Lokalizaci jsem v programu nepoužil, ale v případě tvorby následující verze by to jistě stálo za úvahu.

Dalším faktorem ovlivňujícím rychlost rozpoznávání je načtení a konstrukce sítě, která trvá 70 ms. Tato poměrně dlouhá doba je způsobena diskovými operacemi a parsováním XML souboru a následně vytvářením instance sítě. Celkem bez problémů by bylo možno tuto dobu úplně eliminovat. Řešení je relativně jednoduché. Stačí vytvořit všechny instance naučených sítí hned při spuštění akce rozpoznávání a používat je až do jejího skončení.

5.1.3 Schopnost rozpoznávání

Rozpoznávání obličejů vytvořeným programem není úplně optimální. Síť jsem naučil 10 různých obličejů členů rodiny a známých.

Při stejných nebo podobných světelných podmínkách se síť chovala celkem spolehlivě. Při záběru kamery na rozpoznanou osobu vždy určila tu správnou. Výsledná zobrazená procentuální podobnost byla v rozmezí 45% až 65%.

Jakmile došlo ke snížení intenzity světla, začala kamera dodávat velmi zrnitý obraz. Podobnost stejné osoby klesla přibližně na rozsah 25%-35% a překrývala se s rozsahem podobnosti u osob, které neodpovídaly snímané osobě. V takových podmínkách se síť často pletla. Přibližně každé čtvrté vyhodnocení bylo nesprávné. Několikrát se také stalo, že síť rozpoznala osobu, i když před kamerou nikdo neseděl.

Skutečnost, že při špatném osvětlení a zrnitém obrazu nedává program příliš dobré výsledky nepovažuji za příliš vážný nedostatek. Tato situace by se dala při praktickém nasazení vyřešit dobrým osvětlením nebo kvalitnější kamerou.

Dalším dle mého názoru vážnějším nedostatkem je citlivost sítě na změnu velikosti obličeje v obraze. Pokud se obličej před kamerou velikostně liší více než o 20% pak je již pro síť prakticky nerozpoznatelný.

Tuto vlastnost by bylo možné eliminovat zvětšením a zmenšením vstupního vzoru při učení a vytvořením několika sítí ke každé tváři. Doba vyhodnocení by se sice znásobila, ale rozsah změny měřítka by byl výrazně větší.

5.2 Uživatelské rozhraní

Uživatelské rozhraní není nijak rozsáhlé, jednotlivé funkce jsem testoval sám jejich opakovaným spouštěním a používáním. Jeho vlastnosti mohu posoudit jen subjektivně. Všechny funkce pracovaly tak, jak jsem očekával a program se nedostal do neočekávaného stavu.

5.2.1 Funkční vlastnosti

Funkční vlastnosti specifikované v kapitole 4.2, tedy naučení osoby, rozpoznání osoby a správu osob, jsem testoval opakovaným učením několika osob, jejich následnou identifikací a vyřazováním ze seznamu.

Naučení nové osoby probíhalo bezproblémově a rychle.

Identifikace osoby byla do značné míry ovlivněna intenzitou osvětlení. Program rovněž nebyl schopen osobu identifikovat, pokud byla od kamery výrazně dále nebo blíže než při učení.

Rychlost identifikace klesá s narůstajícím počtem naučených osob. Při testování jsem měl nejvíce současně naučených osob 10. Identifikace v tomto případě trvala přibližně 2 sekundy. Pokud by měl být systém určen pro identifikaci uživatele počítače, pak by byl tento počet dostačující pro domácí nebo firemní využití. Nebyl by ale použitelný například pro identifikaci studenta ve školní učebně. V tom případě by totiž musel obsahovat všechny studenty. Při počtu např. 1000 studentů by pak identifikace trvala přibližně 3,5 minuty. Navíc by se zvyšovala pravděpodobnost, že nesprávně rozpozná podobný obličej.

5.3 Zhodnocení dosažení cílů

Výsledky rozpoznávání nebyly tak dobré, jak jsem zpočátku očekával. Identifikační schopnost navrhnutého řešení je velmi závislá na kvalitě snímaného obrazu, na kvalitě a směru osvětlení a na vzdálenosti osoby od kamery. Všechny tyto faktory snižují schopnosti programu správně určit osobu.

Síť neocognitron byla původně navržena na rozpoznávání znaků písma a v tomto směru jistě dosahuje mnohem lepších výsledků. Jednotlivé znaky abecedy se od sebe výrazně liší. Lidské obličeje jsou si ale do značné míry podobné, proto je jejich odlišení náročnější.

I přes tyto nedostatky bylo v programu dosaženo všech funkčních i nefunkčních požadavků.

5.3.1 Zhodnocení dosažení funkčních požadavků

1. Umožnit přidání osoby k rozpoznatelným osobám: Do programu lze jednoduše přidávat teoreticky neomezený počet osob.Popis použití je v kapitole 4.5.2. a v uživatelské příručce Příloha PI.
2. Umožnit odstranění osoby z rozpoznávaných osob: Osoby ze seznamu naučených osob lze odebírat. Popis použití je v kapitole 4.5.3. a v uživatelské příručce Příloha PI.
3. Při zjištění osoby před kamerou tuto identifikovat a zobrazit další informace: Program s určitými omezeními identifikuje osobu před kamerou. Popis je v kapitole 4.5.4. Omezení spočívá v citlivosti na špatných podmínkách okolí.

5.3.2 Nefunkční požadavky

1. Možnost jednoduchého přenosu naučené osoby na jiný počítač:
   Každá naučená osoba je uložena ve dvou souborech. Jeden s příponou XML obsahuje definici sítě a informaci o osobě. Druhý s příponou JPG obsahuje osobní fotografii osoby. Pro přenos na jiný počítač stačí tyto dva soubory zkopírovat do příslušného adresáře. Není třeba nic dalšího nastavovat.
2. Identifikaci provést v rozumném čase. Maximálně 0,5 sekundy na porovnání s jednou osobou v databázi: Požadavek byl splněn. Doba porovnání s jednou osobou je 200 ms viz kapitola 5.1.
3. Jednoduché uživatelské rozhraní: Uživatelské rozhraní obsahuje tři základní okna viz. kapitola 4.2. Podrobněji je ovládání programu popsáno v uživatelské příručce v Příloze PI.
4. Jednoduchá instalace: Program nevyžaduje instalaci. Stačí zkopírovat kamkoliv spustitelný soubor. Program k ukládání naučených osob využívá adresář na disku, který je standardně přednastaven, ale je možné jej změnit, viz. kapitola 4.2. Vlastní parametry si program ukládá do registru systému. Pokud v příslušných klíčích nejsou, použije přednastavené parametry a klíče vytvoří. Přesto je k programu vytvořen i instalační balíček pro Microsoft installer, viz. Příloha PI.

5.3.3 Návrhy na zlepšení

Z dosažených výsledků a získaných dat lze říct, že je možné produkt vylepšovat ve dvou směrech. První směr je zvyšování rychlosti identifikace, druhý je zlepšení přesnosti identifikace.

Zvyšování rychlosti – Možnosti ke zvýšení rychlosti byly podrobně popsány v kapitolách 5.1.1 a 5.1.2. Pro přehlednost uvádím v tabulce (Tab. 3) jejich výčet s orientační hodnotou předpokládaného možného urychlení.

Tab. 3. Souhrn možností zrychlení programu.

Způsob úpravy	Teoretické urychlení
Udržování instancí naučených sítí v operační paměti.	až 70ms na uloženou osobu
Zmenšení oblasti prohledávání vhodnou separací místa s obličejem.	30 – 50 ms na uloženou osobu
Použití paralelních výpočtů na více jádrech.	Podle počtu jader

- Udržování instancí naučených sítí v operační paměti: 70ms je doba potřebná k vytvoření instance sítě v počítači Pentium4 2,4GHz.
- Zmenšení oblasti prohledávání vhodnou separací místa s obličejem 30-50ms. Tato hodnota je odhad možného zrychlení. Teoreticky by bylo možno dosáhnout i více, ale nějaký další čas zabere samotná separace obličeje z obrazu.
- Použití paralelních výpočtů na více jádrech: Algoritmus se skládá především z samostatných cyklů, které by šly poměrně snadno paralelizovat. Výsledný čas by se tedy v optimálním případě dělil počtem jader.

Při využití všech zmíněných možností by bylo možné např. na 4 jádrovém procesoru dosáhnout rychlosti 20ms na jednu uloženou osobu. Tento čas je 10x menší, než se mi podařilo dosáhnout v této implementaci.

Zvyšování přesnosti – Přesnost identifikace je v tomto programu klíčová a bylo by dobré jí dosáhnout i za cenu snížení rychlosti zpracování. To je možné jen do určité míry. Zvýšení přesnosti identifikace o 1% neospravedlní snížení rychlosti na polovinu, proto je třeba najít rozumný kompromis.

Možnosti zlepšení přesnosti identifikace:

- **Vypuštění nevhodných fragmentů obrazu z prvků pro učení.** Program v první vrstvě k učení použije 16 čtvercových políček obličeje. Může se stát, že některé pole obsahuje téměř rovnoměrnou jednolitou barvu s malými rozdíly. Toto pole potom není vhodné pro učení. Domnívám se, že výsledek ovlivňuje negativně, protože jej lze najít na více místech obrazu.
- **Ukládání sítě naučené na více měřítek obrazu.** Každý nový rozměr uložené sítě umožní zpřesnit identifikaci vzdálené nebo přiblížené osoby. Také ale zabere více času pro identifikaci. Bude nutné otestovat, jak velký vliv na přesnost tato úprava bude mít.
- **Použití kvalitní kamery s ostrým obrazem.** K tvorbě programu jsem použil velmi nekvalitní kameru, která dodávala zrnitý a neostrý obraz. Použitím kvalitnější kamery by mělo dojít ke zlepšení identifikační schopnosti bez dalších nároků.
- **Použití stálého osvětlení z jednoho směru.** Dobré osvětlení ze stálého nejlépe čelního směru výrazně ovlivní kvalitu sejmutého obrazu i u kamery se slabším CCD čipem. Lze také využít infračervené světlo. Infračervené světlo člověk nevidí, ale CCD čipy jsou na něj citlivé.

Všechny zde uvedené návrhy na zvýšení rychlosti i přesnosti, které lze realizovat softwarově, plánuji realizovat ve druhé verzi programu.

ZÁVĚR

V teoretické části této práce jsem stručně popsal základní princip fungování neuronových sítí obecně a strukturu a funkci architektury neuronové sítě neocognitron. Bylo zde uvedeno, jaké typy buněk, ploch a vrstev architektura používá, k čemu slouží a jaké jsou její výhody a nevýhody. Rovněž jsem uvedl předpoklady a omezení pro použití této architektury při rozpoznávání obrazů lidských tváří.

V praktické části práce jsem vytvořil seznam požadavků na program využívající rozpoznávání obličejů, dále seznam případů užití s podrobným popisem interakce programu s uživatelem.

V další části byl sestaven návrh modelu tříd a funkčností jednotlivých objektů.

Implementační řešení je zakončeno popisem jednotlivých funkcí a dialogových oken programu. Tyto funkce jsou podrobněji popsány v Příloze PI.

Nakonec jsem se věnoval průběhu testování a zhodnocení dosažených výsledků včetně srovnání s původními požadavky na program. Toto srovnání ukázalo, že všechny původně plánované funkce byly implementovány. Zároveň jsem v průběhu testování odhalil několik možností na zlepšení výkonu a přesnosti rozpoznávání programu. Tyto možnosti plánuji zahrnout do následující verze programu.

ZÁVĚR V ANGLIČTINĚ

In the theoretical part of this work I have briefly described the basic principles of neural networks in general and the structure and function of neural network architectures Neocognitron. Was here indicated what types of cells, surfaces and layers of architecture is used, what they are and what are its advantages and disadvantages. I also stated the conditions and restrictions on the use of this architecture for pattern recognition of human faces.

In the practical part I created a list of demands to the program that uses face recognition, the list of use case describes in detail the interaction program with the user. In the next section was drafted model classes and functionality of individual objects. Implementing a solution is completed by a description of functions and dialogs of the program. These functions are described in Annex I.

Eventually I worked during the testing and evaluation of achievements, including a comparison with the original program requirements. This comparison showed that all the originally planned features were implemented. At the same time I discovered during testing of several options to improve performance and accuracy of the recognition program. These options include the plan in the next version.

SEZNAM POUŽITÉ LITERATURY

[1] **Biometrické metody v bezpečnostní praxi.** Článek v časopise 3pól, Prosinec 2006. Dostupný z WWW: <http://www.tretipol.cz/501-biometricke-metody-v-bezpecnostni-praxi-(2)>

[2] **ZELINKA, I.** ***Umělá inteligence I: Neuronové sítě a genetické algoritm.y*** 1. vydání VUT v Brně, nakladatelství VUTIUM **1998**. ISBN 80-214-1163-5.

[3] **Jirsík, V.,Hráček, P. Neuronové sítě, expertní systémy a rozpoznávání řeči**., skriptum Fakulty elektrotechniky a komunikačních technologií, VUT Brno

[4] **HNÁTEK, J.** ***COURSEWARE*** [online]. 2002 [cit. 2010-05-10]. ***Jeden perceptron - klasifikace.*** Dostupné z WWW: <http://neuron.felk.cvut.cz/courseware/data/chapter/36nan028/s04.html>**.**

[5] **FUKUSHIMA K.:** ***Neocognitron: a self-organizing neural network model for a mechanism of patterns recognition unaffected by shift in position***, Biological Cybernetics, 32:193-202,**1980**

[6] **VELÍNSKÝ, T.** ***Neuronová síť Neocognitron: Výukové pásmo*** [online]. [cit. 2010-05-04]. Dostupný z WWW: <http://neuron.felk.cvut.cz/courseware/data/chapter/velinsky2000/>.

[7] **FUKUSHIMA, K., MIYAKE S., ITO, T.*****: Neocognitron: a neural network model for a mechanism of visual pattern recognition.*** IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-13(Nb. 3):pp.826--834, September/October **1983**.

[8] **FUKUSHIMA K.:** ***Visual pattern recognition with neural networks***, **1992**, ISBN 978-3-540-56346-4.

[9] **BOSE, N.K., LIANG, P.:** ***Neural Network Fundamentals with Graphs, Algorithms, and Applications,*** McGraw-Hill Series in Electrical and Computer Engineering**, 1996**, ISBN 0-07-006618-3.

[10] **TAKIMOTO, H., MITSUKURA, Y., AKAMATSU, N., KHOSLA, R.:** ***Face Identification Method Using the Face Shape***, **2003**, ISBN 978-3-540-40803-1.

[11] **BISHOP, Ch. M.:** ***Pattern Recognition and Machine Learning***, Springer, 2006, ISBN 0-387-31073-8.

[12] **ŠČUREK, Radomír**. ***Biometrické metody identifikace osob v bezpečnostní praxi***. VŠB TU Ostrava **2008**.

[13] **HINNER, J.** ***Detekce a rozpoznávání obličejů osob a jejich identifikační význam***. Časopis Kriminalistika ročník XXXVI1/2003

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

S-cell	Neuron ve vrstvě S
C-cell	Neuron ve vrstvě C
V-cell	Neuron ve vrstvě V
U0	Vstupní vrstva
US1	První vrstva typu S
USC	Univerzity of Southern California
FBG	Face Bunch Graph
PCA	Principial component analysis
LDA	Lineární diskrétní analýza
.NET	Knihovna oběktových tříd pro MS Windows
SVD	singular value decomposition
UC	USE-CASE (případ užití)
CCD	Charge-Coupled Device – čip pro snímání obrazu.

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

Příloha P I: Uživatelská příručka k programu

Příloha P II: Instalační CD s programem a zdrojovým kódem

Příloha P I: Uživatelská příručka k Programu

Neocognitron

Face

Identification

Verze 1.0.0.0

Uživatelská příručka

Obsah:

1 INSTALACE

1.1 Minimální požadavky

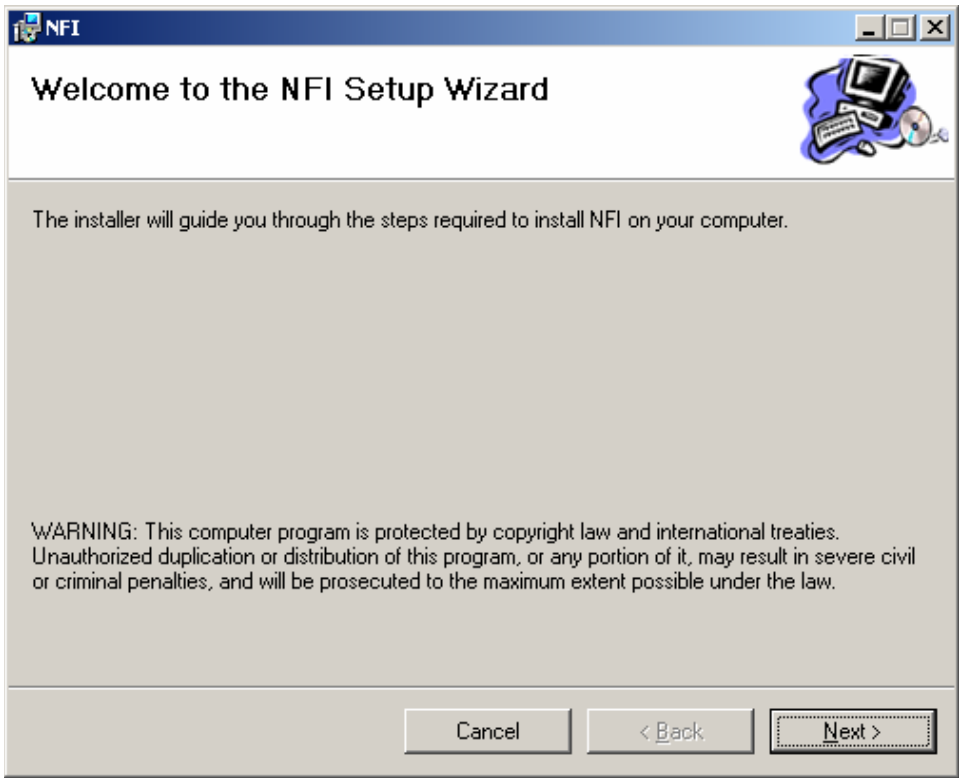
Pro správný běh programu je třeba:

Počítač:	IBM kompatibilní
Paměť RAM:	512 MB a více
Místo na HDD:	10MB
Operační systém:	Windows XP,Windows Vista, Windows 7
Instalované knihovny	.NET Verze 3.5
Příslušenství:	Webkamera

1.2 Postup instalace

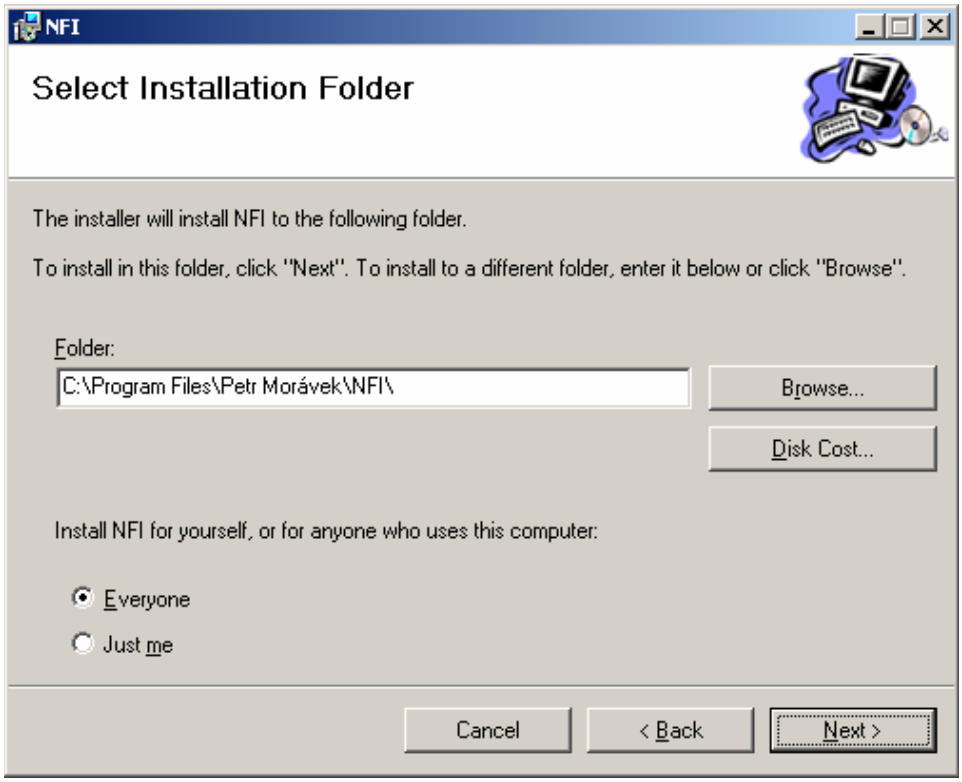
Instalace programu v systému není nutná. Na datovém nosiči je ale dodán i instalátor, který uloží program do složky programů, zaregistruje jej v systému Windows a vytvoří k němu zástupce.
Instalaci spustíte spuštěním instalátoru NFI_Install_1.msi.

Po spuštění instalace se zobrazí uvítací dialogové okno. (Obr. 1)



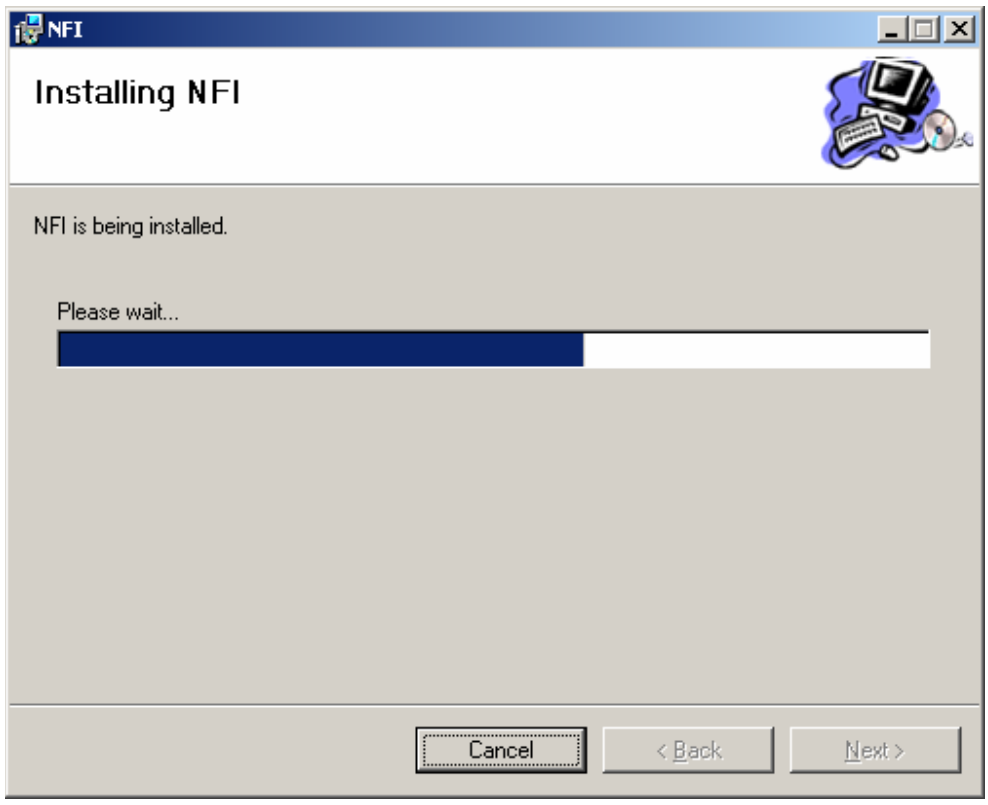
Obr. 1. Uvítací dialog.

Pokračujte tlačítkem „Next“.
Následuje okno pro výběr složky pro umístění instalovaného programu (Obr. 2)



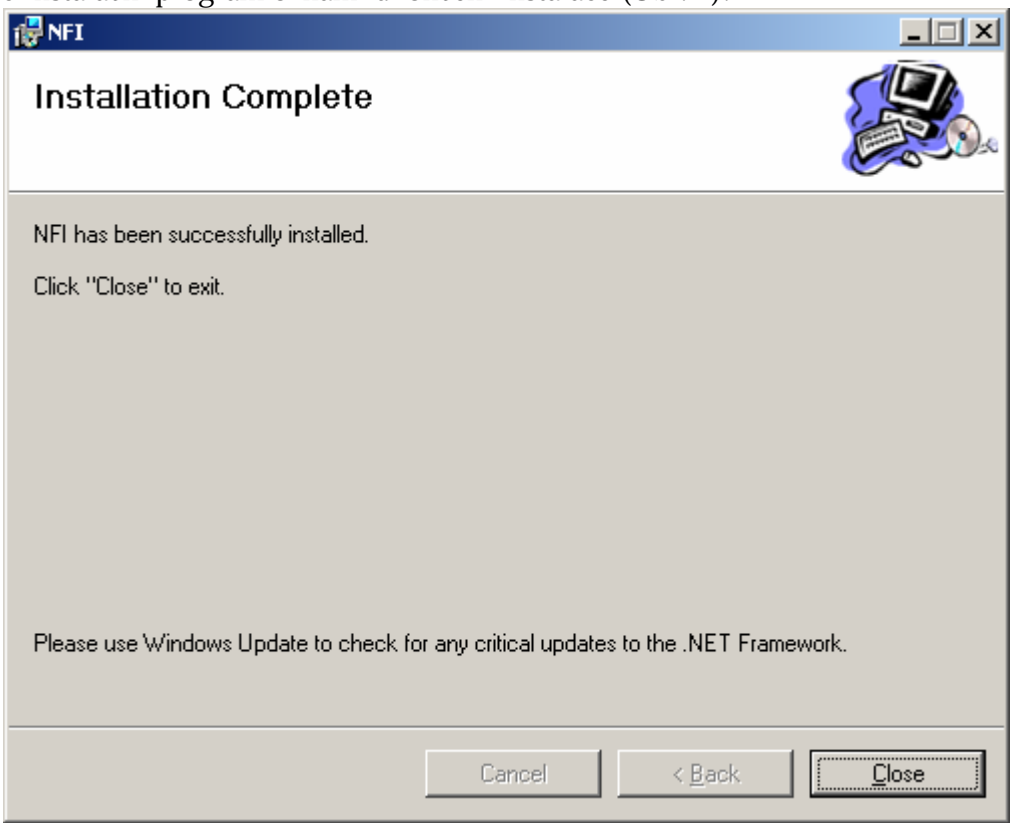
Obr. 2. Okno pro výběr složky a typu instalace.

Pokud chcete složku, nebo typ instalace změnit, můžete tak učinit. Jinak pokračujte tlačítkem „Next“. Poté bude spuštěna instalace a můžete vidět její průběh (Obr. 3).



Obr. 3. Průběh instalace programu.

Nakonec Instalační program oznámí ukončení instalace (Obr. 4).



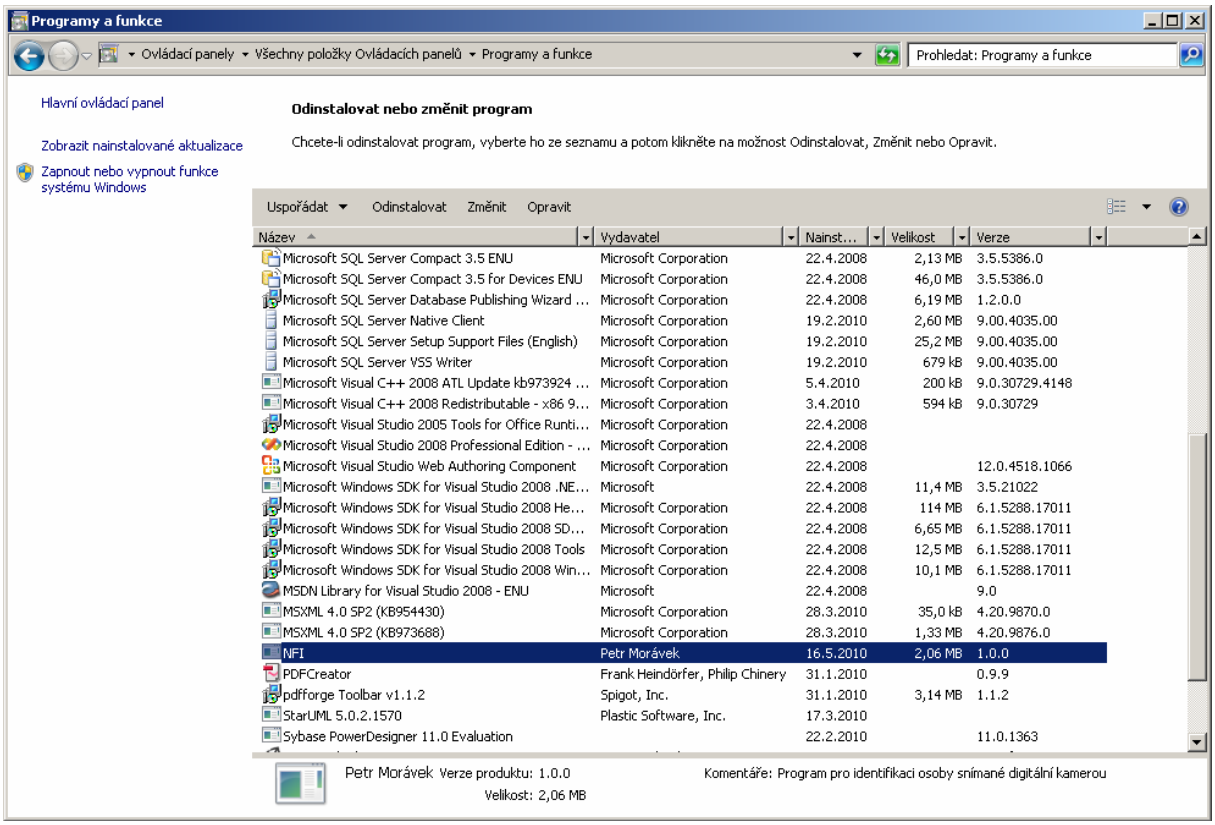
Obr. 4. Konec instalace

Nyní můžete instalační program uzavřít tlačítkem „Close“.

Program můžete spustit zástupcem v Start – Programy – Morávek – NFI.

1.3 Odinstalace

Program nemá svůj vlastní odinstalátor, takže pro případnou odinstalaci využijte funkce Přidat/odebrat programy v ovládacích panelech systému Windows viz (Obr.5).



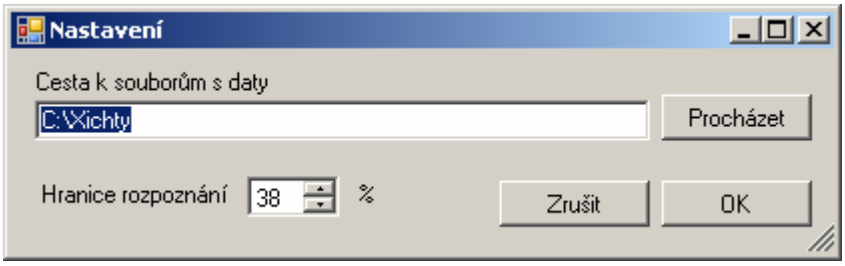
Obr. 5. Ovládací panely systému Windows

2 NASTAVENÍ

K funkci programu je třeba nastavit dva volitelné parametry. Prvním z nich je cesta ke složce na disku, do níž se budou ukládat naučené osoby. Druhý parametr je úroveň shody potřebná pro rozpoznání.

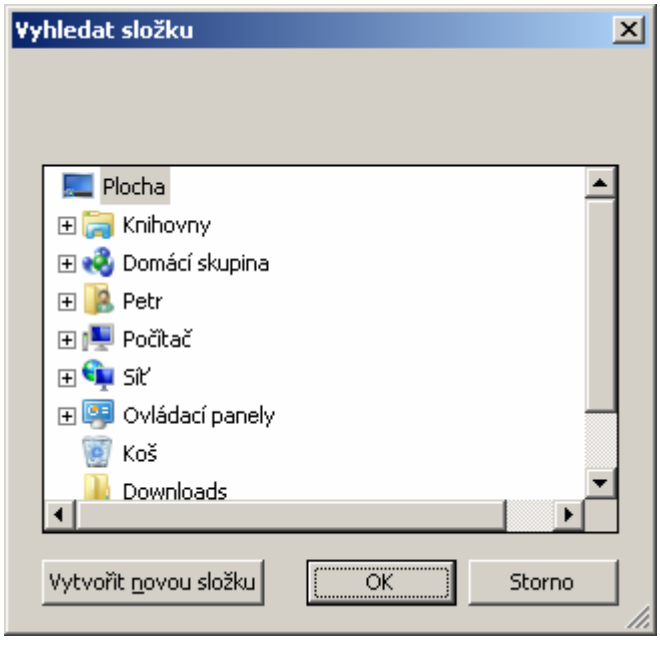
2.1 Nastavení složky pro ukládání

Spusťte program NFI a v hlavním menu zvolte Nastavení – Prostředí.
Zobrazí se malé dialogové okno s nastavením parametrů (Obr. 6.).



Obr. 6. Okno pro nastavení parametrů.

Pokud máte určitou složku, do níž s přejete ukládat. Můžete ji vepsat ručně nebo vyhledat tlačítkem „Procházet“.
Pokud složka neexistuje, můžete ji vytvořit volbou „Vytvořit novou složku“ Při procházení složek viz (Obr. 7.).



Obr. 7. Procházení složek systému.

2.2 Nastavení úrovně shody pro rozpoznání.

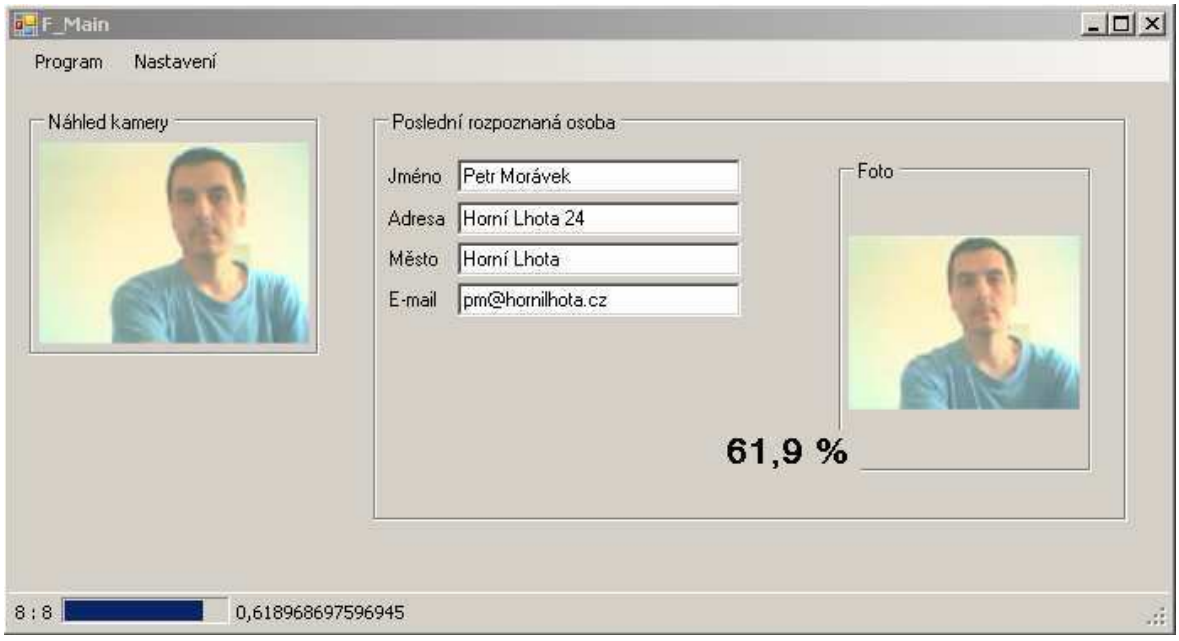
Tuto volbu můžete změnit ve stejném okně, ve kterém nastavení složky. V položce „Hranice rozpoznání zadejte na kolik procent má snímaná osoba odpovídat naučené, aby ji systém označil za uloženou osobu. Tuto hodnotu je třeba si vyzkoušet, zejména podle toho, na kolik procent systém běžně osoby ohodnotí. Mě se osvědčila hodnota kolem 40%.

3 UŽIVATELSKÉ FUNKCE

3.1 Popis úvodní obrazovky

Po spuštění programu se uživateli zobrazí úvodní obrazovka. Úvodní obrazovka obsahuje hlavní menu pro spouštění jednotlivých funkcí programu, stavový řádek pro přehled o aktuálně prováděné akci, prostor okna je použit pro zobrazení náhledu obrazu snímaného kamerou a zobrazení naposled identifikované osoby.
Pohled na úvodní obrazovku je na obrázku je na (Obr. 8.)

Vlevo v rámečku „Náhled osoby“ se zobrazuje poslední snímek sejmutý z kamery. Podle tohoto náhledu můžeme upravit směr snímání kamery tak, aby zabíral celý obličej pozorované osoby.



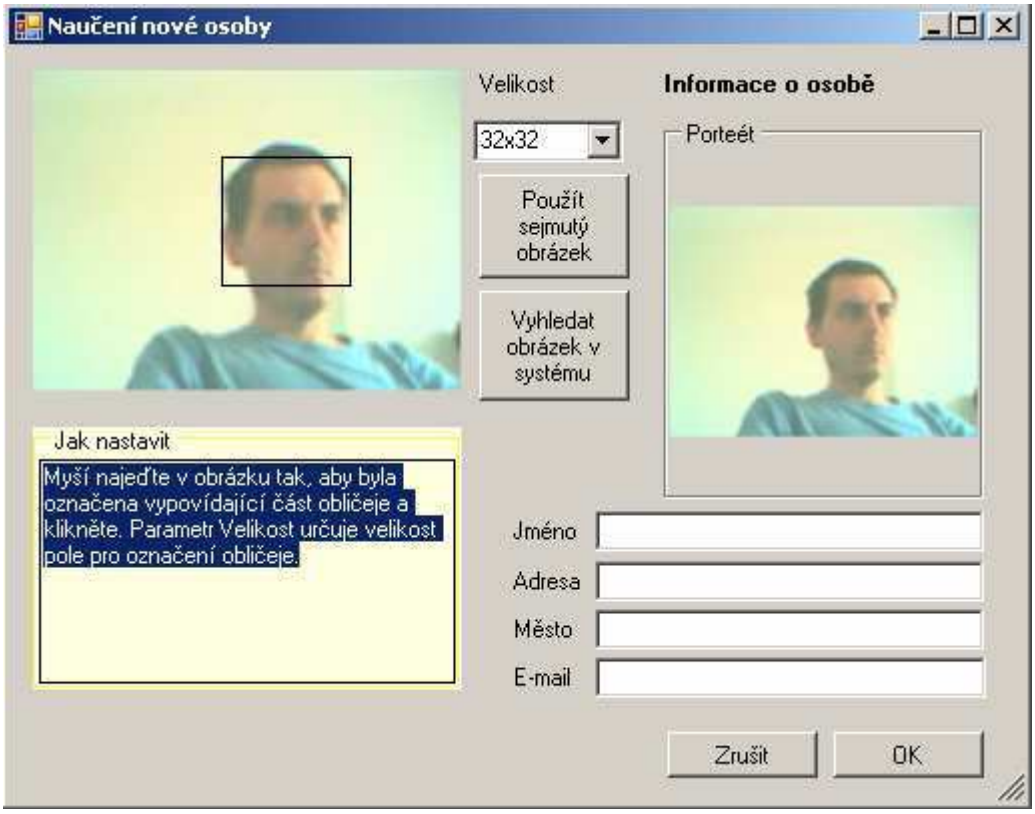
Obr. 8. Úvodní obrazovka programu.

Vpravo v rámečku “Poslední rozpoznaná osoba“ jsou zobrazena osobní data osoby, která byla naposled rozpoznána s minimální přesností nastavenou viz kapitola 2.2.
Na obrázku (Obr. 8.) byl obličej rozpoznán s přesností 61,9 %, což je indikováno velkým číslem vedle portrétu nalezené osoby vpravo.

Stavový řádek obsahuje ukazatel průběhu rozpoznávání a taktéž poměrové číslo identifikace naposled testované osoby.

3.2 Naučení osoby

Funkce naučení osoby se spustí položkou „Učení“ v menu „Program“.
Program zobrazí dialogové okno pro naučení nové osoby (Obr. 9.). Okno obsahuje naposled sejmutý obraz z náhledového okna vlevo, proto musí být před spuštěním učení spuštěn náhled snímání (Menu Program – Náhled). Pod obrázkem je informační oblast s popisem požadované činnosti. Dále vpravo textová pole pro zadání osobních údajů osoby pro naučení a pole s fotografií osoby. Jako výchozí fotografie je použit sejmutý obraz, ale je možné fotografii vyměnit. Dvě tlačítka uprostřed slouží ke změně fotografie osoby, nebo vrácení zpět sejmutého obrázku.



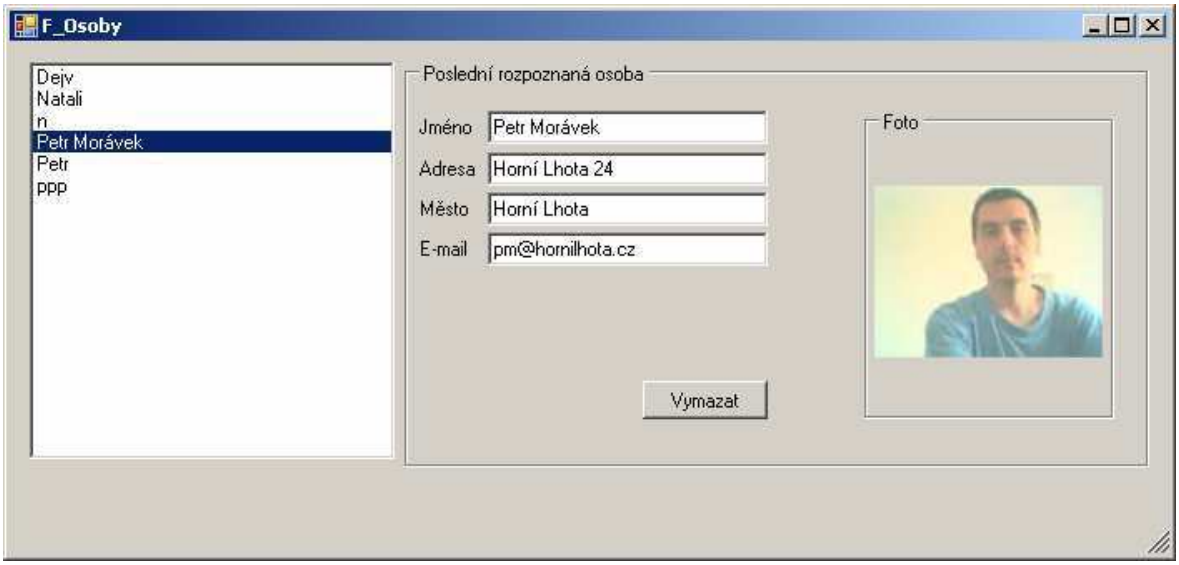
Obr. 9. Okno pro učení nové osoby.

V okně pro učení nové osoby (Obr. 9) je třeba myší označit na sejmutém obraze oblast, kde se nachází obličej. Oblast by neměla obličej přesahovat. Pokud je velikost čtverce příliš velká nebo malá, je třeba zvolit jiný rozměr v poli „Velikost“ vedle obrázku. Pokud jsou všechny údaje nastaveny. Můžeme spustit učení tlačítkem „OK“. To provede vytvoření sítě, její naučení a uložení.

3.3 Správa naučených osob

Na osoby, které se již program naučil se můžeme podívat přes menu „Nastavení - Osoby“. Tato volba nám zobrazí okno pro správu osob. Zde se zobrazí seznam naučených osob, ve kterém je možno listovat. Při kliknutí na řádek seznamu se v pravé části zobrazí osoba příslušející k tomuto záznamu a její osobní údaje. Uživatel zde může změnit informace o osobě, nemá už ale možnost zasáhnout do parametrů naučené sítě. Pokud není naučená osoba správně rozpoznávána, je dobré ji ze seznamu odstranit.

Pohled na okno se seznamem osob je na obrázku (Obr. 10.)



Obr. 10. Pohled na okno pro správu osob.

K vymazání aktuální osoby slouží tlačítko „Vymazat“. K výběru zobrazované osoby slouží seznam vlevo. Ten je sestaven se jmen naučených osob. Výběrem osoby v seznamu se automaticky zaktualizují údaje vpravo.

3.4 Rozpoznávání

Rozpoznávání osoby se spustí v hlavním menu volbou „Program – Běh“.
Po spuštění, začne systém v intervalu 500 milisekund snímat obrázek z digitální kamery a podrobovat jej identifikaci postupně s každou naučenou osobou. Doba potřebná na identifikaci může být různá na různých počítačích. Na počítači s procesorem Pentium4 trvá rozpoznání 150-200 milisekund.
Z toho plyne, že při 3 a více naučených osobách program překročí snímací interval. To ničemu nevadí. Jen se prodlouží doba mezi jednotlivými snímacími cykly. Cyklus snímání je pak stejný jako doba vyhodnocování snímků.

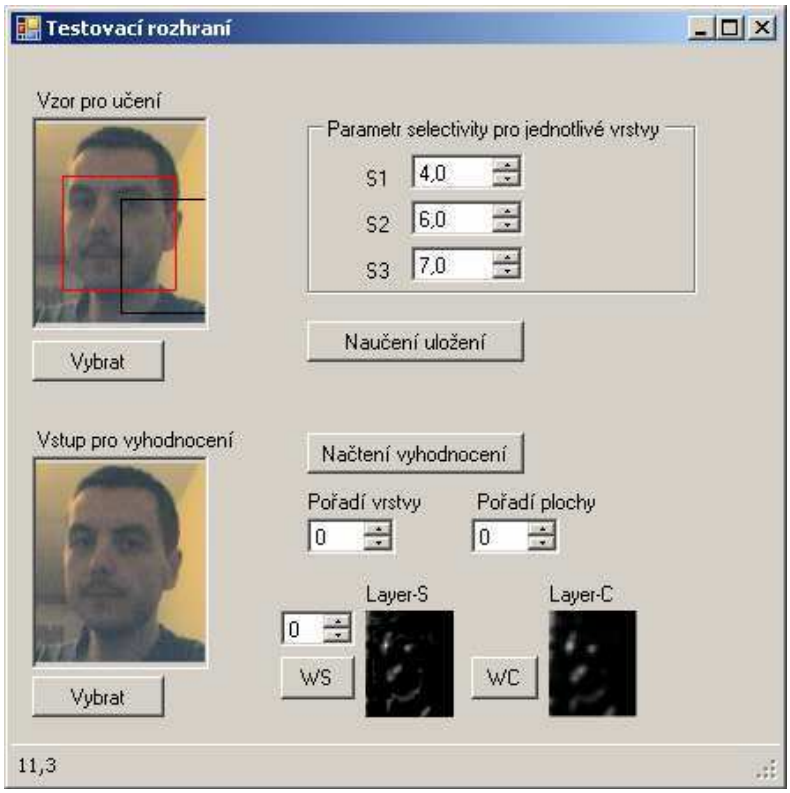
Například pokud Doba vyhodnocení snímků je 850 ms, potom po 850 milisekundách je nasnímán další snímek a začne jeho zpracování. Systém nečeká na konec dalšího 500ms intervalu.

Průběh rozpoznávání aktuálního snímku je indikován progressbarem ve stavovém řádku. Pokud snímaný obraz obsahoval obličej některé z naučených osob, je vybraná osoba zobrazena vpravo společně s dalšími informacemi o ní viz. (Obr. 8.).

Rozpoznávání lze ukončit přepnutím opět na náhled v menu „Program – Náhled“.

3.5 Testovací rozhraní

Testovací rozhraní slouží k vizualizaci hodnot vzniklých při učení sítě. Spustí se z menu „Nastavení – Test“. Testovací rozhraní nepoužívá vstup z digitální kamery, ale pouze dodané statické obrázky. Obsahuje jedno dialogové okno pro nastavení parametrů a spuštění funkcí pro učení a zobrazení výsledků.



Obr. 11. Testovací rozhraní pro sledování mezivýsledků.

Na (Obr. 11.) je vidět pohled na testovací okno po naučení vzoru. Vstup obrázku pro naučení vzoru je z obrázku s nadpisem „Vzor pro učení“. Tlačítko pod obrázkem slouží k jeho výměně za jiný. Parametry selektivity vpravo od vzoru slouží k nastavení parametrů selektivity.

Tlačítko „Naučení a uložení“ spustí funkci, která z obrázku „Vzor pro učení“ vygeneruje síť neocognitron a uloží ji do seznamu sítí. Její název si zapamatuje.

Tlačítko „Načtení vyhodnocení“ Načte naposled uloženou síť a pomocí ní vyhodnotí obrázek „Vstup pro vyhodnocení“.

Dva malé tmavé obrázky Layer-S a Layer-C zobrazují vizualizaci aktivity buněk ve vrstvách S a C po výpočtu nad obrázkem. Zobrazované vrstvy lze přepínat v ovládacích prvcích „Pořadí vrstvy“ a „pořadí plochy“.

Dvojklikem na obrázek Layer-S lze v tabulce zobrazit číselné hodnoty aktivit buněk.

Tlačítka WS a WC slouží k zobrazení číselných hodnot příslušných vah.