Obchodní akademie, Vyšší odborná škola a Jazyková škola
s právem státní jazykové zkoušky Uherské Hradiště

MATURITNÍ PROJEKT

ROZPOZNÁNÍ OBJEKTŮ V OBRAZE

2012 Lukáš Machalík, 4.B

Prohlášení:

Souhlasím s tím, že s výsledky mé práce může být naloženo podle uvážení vedoucího maturitní práce a ředitele školy. V případě publikace budu uveden jako spoluautor.

Prohlašuji, že jsem na celé maturitní práci pracoval samostatně a veškeré použité zdroje jsem citoval.

V Uherském Hradišti, dne 9. března 2012

.....

podpis absolventa

Poděkování:

Děkuji vedoucímu práce Mgr. Jaroslavu Hodlovi za obětavou pomoc a podnětné připomínky, které mi během práce poskytl. Děkuji spolužákům, kteří se při testování aplikace skvěle pobavili.

RESUMÉ

Tato aplikace pro kreslení umí využít vaší webkamery ke snímání obrazu. Pohybem jakéhokoliv červeného, zeleného nebo modrého objektu se určuje pozice štětce. Můžete tak kreslit přímo do okna webkamery nejrůznější obrazce, měnit barvu a šířku štětce. Nakreslený obrázek můžete vymazat a začít s kreslením od znova.

OBSAH

ÚVOD **6**

1 UŽIVATELSKÁ PŘÍRUČKA **7**

1.1 Okno pro výběr webkamery 7

1.2 Hlavní okno 7

1.2.1 Vymazání plátna 9

1.2.2 Změna barvy štětce 9

1.2.3 Boční panel rychlého ovládání 10

1.3 Změna citlivosti 11

1.4 Ukončení aplikace 11

2 TECHNICKÉ ZPRACOVÁNÍ **12**

2.1 Připojení webkamery 12

2.2 Získání obrazu z webkamery 13

2.2.1 Práce s pamětí 13

2.2.2 Práce se snímkem 14

2.3 Nalezení objektu v obraze 14

2.3.1 Histogramový filtr 15

2.3.2 Rozpoznání pomocí barevného kanálu 16

2.4 Srovnání algoritmů 20

2.5 Kreslení pomocí rozpoznaného objektu 20

2.5.1 Boční panel rychlého ovládání 21

2.6 Hlavní okno aplikace 22

2.6.1 Obraz webkamery 22

2.6.2 Režim celé obrazovky 23

2.7 Seznam tříd aplikace 24

ZÁVĚR **25**

SEZNAM POUŽITÉ LITERATURY **26**

SEZNAM OBRÁZKŮ **27**

SEZNAM TABULEK **28**

SEZNAM PŘÍLOH **29**

ÚVOD

V listopadu 2010 společnost Microsoft uvedla na trh Kinect[1] jako doplněk herní konzole Xbox 360 a změnila tak dosud zavedené způsoby hraní her. Toto vstupní zařízení pracuje na principu rozpoznání objektu (člověka) a sledování pohybu pomocí několika kamer.

Tato technologie mě zaujala a začal jsem zkoumat, zda je v mých schopnostech podobný software naprogramovat. Jako nejlevnější přijde využití webkamery a z toho plynoucí rozpoznání objektu v obraze na základě barvy.

Úkolem této práce je vytvoření algoritmu a následně implementace v jednoduché aplikaci, která využívá webkameru pro snímání obrazu v reálném čase. Algoritmus analyzuje snímky a rozpozná objekt, nalezený objekt dále sleduje a zaznamenává jeho pohyb.

Pro jednoduchou demonstraci lze využít zaznamenávání pohybu objektu ke kreslení do obrazu webkamery.

---

[1] Kinect. In: Wikipedia [online]. 12.3.2012 [cit. 2012-03-18].
Dostupné z: http://en.wikipedia.org/wiki/Kinect

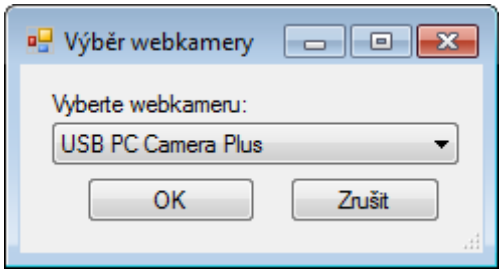
1 UŽIVATELSKÁ PŘÍRUČKA

Před spuštěním aplikace se ujistěte, že máte alespoň jednu webkameru připojenou k počítači. Musíte mít nainstalován .NET Framework 4. Dále budete potřebovat červený, zelený nebo modrý předmět, nejlépe barevně svítící. Na přiloženém CD naleznete také pomocné ideálně barevné obrázky, které lze nahrát do mobilního telefonu, spustit prohlížečem obrázků a použít je jako barevný objekt.

Aplikaci naleznete na přiloženém CD, spustíte ji poklepáním na soubor „Kresleni.exe“. Ve stejné složce jako aplikace musí být přiložen soubor „MCSystem.dll“ a soubor „DirectShowLib-2005.dll“.

1.1 Okno pro výběr webkamery

Máte-li více jak jednu webkameru, zobrazí se vám okno pro výběr webkamery. Vyberte tu, kterou chcete použít a volbu potvrďte stisknutím tlačítka „OK“. Stisknutím tlačítka „Zrušit“ ukončíte aplikaci.



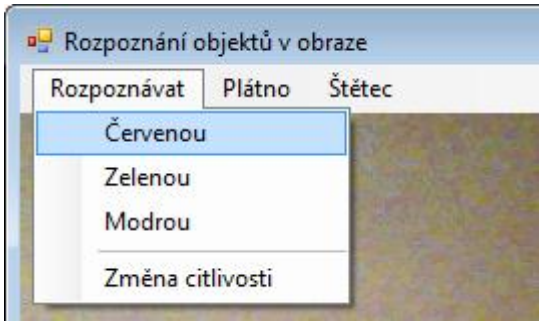
Obrázek 1 Okno pro výběr webkamery

1.2 Hlavní okno

V hlavním okně aplikace se zobrazuje panel nabídek, obraz z webkamery (plátno) a v obraze při bočním okraji okna naleznete boční panel rychlého ovládání.

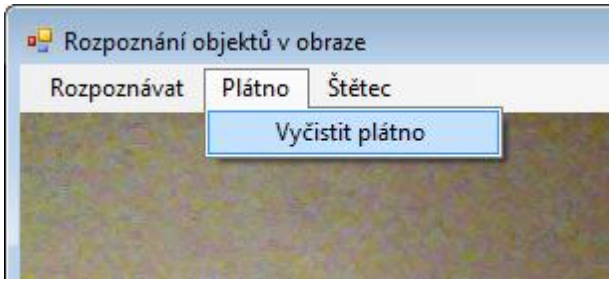
Obrázek 2 Rozložení hlavního okna aplikace

Nejprve klikněte v panelu nabídek na položku „Rozpoznávat“ a vyberte rozpoznávání takové barvy, jaké máte připravený objekt.



Obrázek 3 Rozevřená nabídka Rozpoznávat

Dále v panelu nabídek zvolte možnost „Plátno“ a zvolte volbu „Vyčistit plátno“.

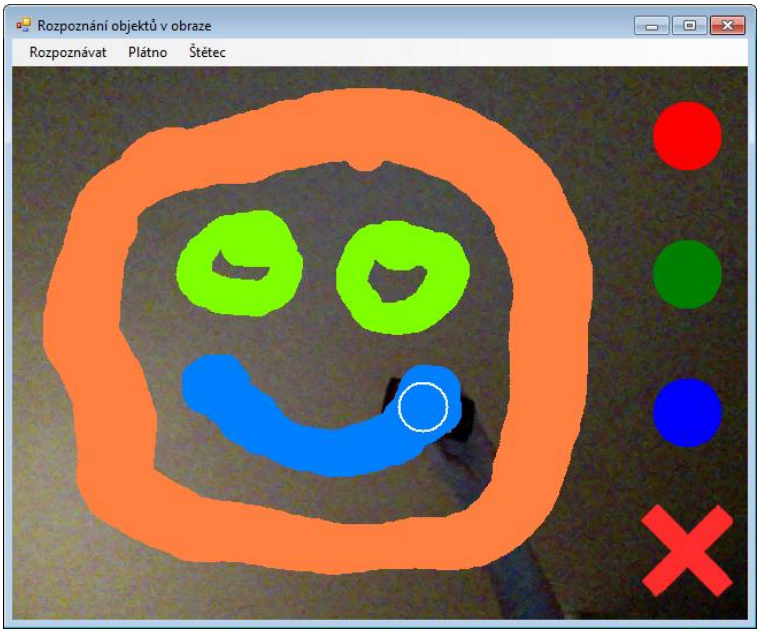


Obrázek 4 Rozevřená nabídka Plátno

Nyní si stoupněte před webkameru a připravený objekt zobrazte před webkamerou. Aplikace by měla objekt okamžitě rozpoznat. Pohybem objektu před webkamerou

zanecháváte za objektem stopu a tímto způsobem můžete kreslit libovolné obrazce. Velikost objektu také ovlivňuje velikost štětce, kterým kreslíte.

Pokud aplikace objekt nerozpoznala (nic se do obrazu webkamery nekreslí), přejděte na kapitolu 1.3 Změna citlivosti.



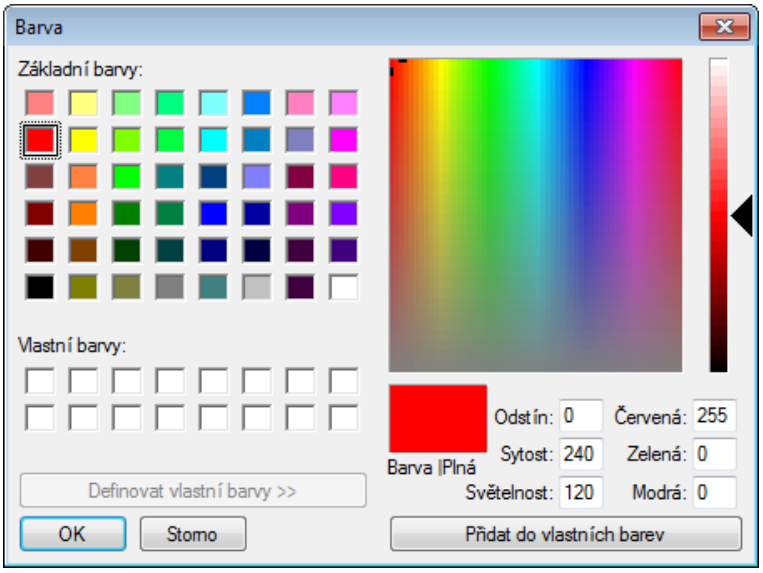
Obrázek 5 Ukázka jednoduché kresby vytvořené pohybem

1.2.1 Vymazání plátna

Vymazání nakresleného obrazce (vyčištění plátna) provedete zvolením možnosti „Plátno“ v panelu nabídek a volbou „Vyčistit plátno“.

1.2.2 Změna barvy štětce

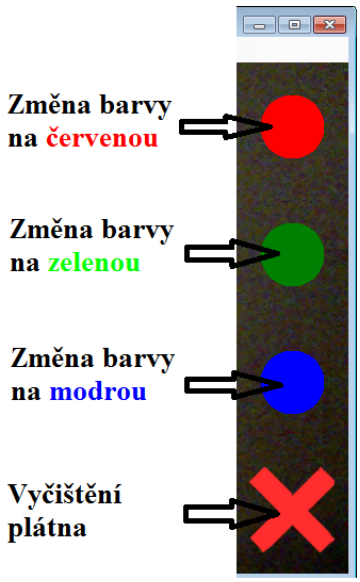
Změnu barvy štětce provedete zvolením možnosti „Štětec“ v panelu nabídek a zvolením volby „Změna barvy štětce“. Zobrazí se vám okno s výběrem barev. Změnu barvy potvrdíte tlačítkem „OK“. Pokud si barvu změnit nepřejete, můžete dialog zavřít tlačítkem „Storno“.



Obrázek 6 Okno pro změnu barvy štětce

1.2.3 Boční panel rychlého ovládání

Rychlou změnu barvy nebo vymazání plátna můžete také provést pomocí bočního panelu rychlého ovládání. Přesuňte sledovaný objekt do oblasti panelu (k pravému okraji okna) a přesunutím do blízkosti některého z barevných kruhů změníte barvu štětce na barvu kruhu. Podobně přesunutím na červený symbol X vyčistíte plátno.

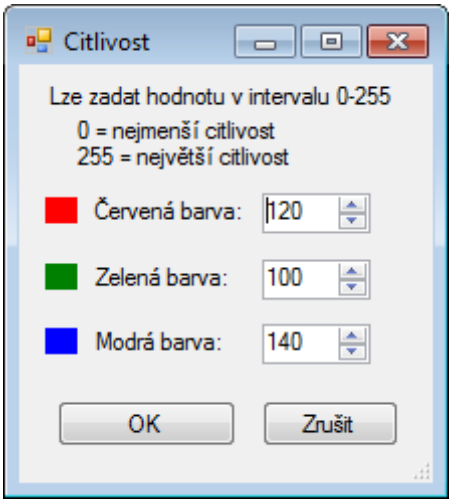


Obrázek 7 Popis položek bočního panelu

1.3 Změna citlivosti

Pokud se aplikaci nedaří rozpoznat objekt, můžete změnit citlivost s jakou aplikace rozpoznává objekt.

Změnu citlivosti provedete zvolením možnosti „Rozpoznávání“ a výběrem volby „Změna citlivosti“. Zobrazí se okno, které obsahuje pole pro zadání hodnoty citlivosti pro každou z možných rozpoznávaných barev. V polích jsou předvyplněné aktuální hodnoty citlivosti. Citlivost můžete zadat v intervalu 0 až 255.



Obrázek 8 Okno pro změnu citlivosti

Zmenšením hodnoty se citlivost rozpoznávané barvy sníží. Pokud má aplikace problém s rozpoznáním objektu, snižte hodnotu citlivosti. Změny hodnoty měňte postupně o malé hodnoty, například o 5 nebo 10 jednotek.

Zvětšením hodnoty se citlivost rozpoznávané barvy zvýší, můžete tak zabránit rozpoznání objektů, které nejsou dostatečně barevné (okolním objektům, šumu).

Nové hodnoty citlivosti potvrdíte kliknutím na tlačítko „OK“. Citlivost snímání rozpoznávaného objektu se ihned změní. Pokud si nové hodnoty uložit nepřejete, můžete dialog zavřít tlačítkem „Zrušit“.

1.4 Ukončení aplikace

Aplikaci ukončíte kliknutím na zavírací tlačítko v pravém horním rohu okna. Po zavření okna bude práce s webkamerou přerušena a můžete ji bezpečně odpojit.

2 TECHNICKÉ ZPRACOVÁNÍ

Úkolem je vytvoření vhodného algoritmu pro zpracování obrazu a nalezení barevného objektu v obraze. Implementace algoritmu je realizována v jazyce C# a demonstrována na jednoduché aplikaci, která dovoluje snímaným pohybem objektu před webkamerou kreslit do obrazu webkamery a výsledný obraz zobrazovat na obrazovce.

2.1 Připojení webkamery

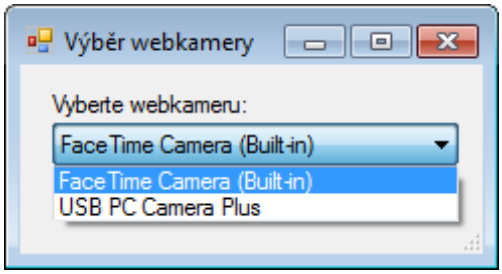
Pro připojení webkamery jsem dlouho hledal vhodnou knihovnu, která by uměla jednoduše pracovat s webkamerou a zároveň byla výkonově vhodná pro snímání a práci se snímky v reálném čase. Nenalezení vhodné knihovny pro jazyk Java je jedním z hlavních důvodů, proč jsem zvolil jazyk C#.

Po vyzkoušení nejrůznějších knihoven jsem zvolil svobodnou knihovnu MCSystem[2], která má velmi jednoduché rozhraní pro práci s webkamerou, umožňuje manuální práci s pamětí a je také velmi rychlá. Největší nevýhodou knihovny je nedostatečná dokumentace.

Knihovna získává obraz pomocí ovladačů DirectX. Sama zjistí všechny dostupné webkamery a jejich vlastnosti jako typ kamery, rozlišení, barevná hloubka a další.

Pomocí knihovny nejdříve zjistím všechny dostupné webkamery. V případě, je-li rozpoznána více než jedna webkamera, je zobrazeno malé okno s výběrem webkamery, kterou chceme použít.

Je-li zjištěna pouze jedna webkamera, zvolí se právě ta a uživateli se okno pro výběr webkamery nezobrazí.



Obrázek 9 Rozevřený seznam webkamer pro výběr

[2] MCSystem. Firielentul [online]. 2011 [cit. 2012-03-18].
Dostupné z: http://firielentul.canis-ebrius.com/MCSystem/index.html

Ke zvolené webkameře se připojíme a spustíme snímání obrazu.

2.2 Získání obrazu z webkamery

Knihovna webkamery umožnuje přímo vykreslovat obraz na obrazovku (předám jí instanci PictureBoxu, kterou ovládne a o vykreslování se stará sama). Jelikož do takto vykreslovaného obrazu nejde nic dále vkládat (kreslit), získávám a vykresluji obraz webkamery manuálně. Pomocí časovače zachycuji snímek webkamery několikrát za sekundu (každých 30 ms – necelých 30 fps). Snímky pak po zpracování stejně často vykresluji na obrazovku, takže uživatel nepozná, že se nejedná o plynulý přímý obraz z webkamery.

2.2.1 Práce s pamětí

Kvůli rychlosti a efektivitě práce s pamětí jsem se rozhodl vkládat jednotlivé snímky manuálně do paměti a přímo z paměti je vykreslovat. Využívám k tomu třídu Marshal, která dovoluje práci s unmanaged pamětí[3].

Při připojení webkamery po spuštění aplikace alokuji v paměti místo ve velikosti jednoho snímku:

```
IntPtr scan0 = IntPtr.Zero;
scan0 = Marshal.AllocCoTaskMem(webcam.DataLength);
```

Jakmile knihovna webkamery poskytne nový snímek, na alokované místo v paměti se nakopíruje:

```
byte[] data = webcam.SnapshotByte();
Marshal.Copy(data, 0, scan0, webcam.DataLength);
```

Výslednou bitmapu (třída Bitmap) pro vykreslení snímku na obrazovku pak nově nevytvářím pixel po pixelu (velmi časově náročné), ale vytvořím ji z dat v paměti:

```
Bitmap novy = new Bitmap(..., ..., ..., ..., scan0);
```

Při ukončení nebo krachu aplikace bezpečně alokovanou paměť uvolním:

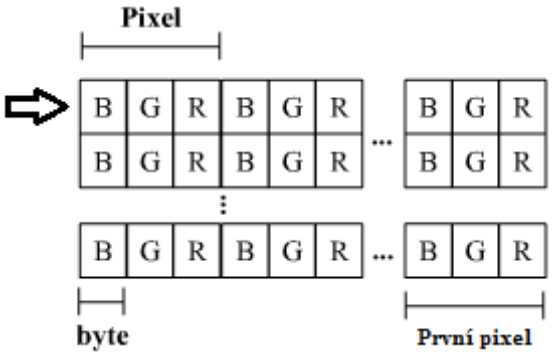
```
Marshal.FreeCoTaskMem(scan0);
```

---

[3] Marshal Class. MSDN Library [online]. 2012 [cit. 2012-03-18]. Dostupné z: http://msdn.microsoft.com/cs-cz/library/system.runtime.interopservices.marshal.aspx

2.2.2 Práce se snímkem

Snímek je knihovnou předán aplikaci jako pole bajtů ve formátu RGB, vždy jeden pixel je reprezentován po sobě jdoucími třemi bajty – hodnotou červené (R), zelené (G) a modré barvy (B). Pole bajtů je ovšem v opačném pořadí, tedy první bajt pole reprezentuje B hodnotu posledního (pravého dolního) pixelu a poslední bajt pole reprezentuje R hodnotu prvního pixelu (levý horní pixel snímku).



Obrázek 10 Znázornění uspořádání pole (zalomeno na řádky)

(zdroj: http://www.switchonthecode.com/tutorials/csharp-tutorial-convert-a-color-image-to-grayscale)

Kvůli obrácenému pořadí pole a zjednodušení práce se snímkem jsem se vytvořil třídu Obrazek, která zapouzdřuje snímek (pole bajtů) a má metody pro čtení (zjištění) červené/zelené/modré hodnoty pixelu na zadané pozici.

Vzorce pro výpočet indexů RGB hodnot v poli odpovídající pixelu na zadané pozici (x, y):

$$\text{Index červené barvy} = \text{velikost pole} - 1 - (\text{šířka} * 3 * y) - (x * 3) \quad (1)$$

$$\text{Index zelené barvy} = \text{velikost pole} - 1 - (\text{šířka} * 3 * y) - (x * 3) - 1 \quad (2)$$

$$\text{Index modré barvy} = \text{velikost pole} - 1 - (\text{šířka} * 3 * y) - (x * 3) - 2 \quad (3)$$

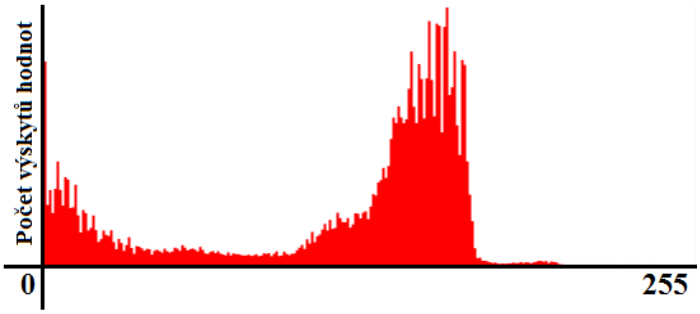
Poznámka: Vzorce jsou využity ve třídě Obrazek v metodách getR(), getG() a getB().

2.3 Nalezení objektu v obraze

Díky webkameře mám možnost rozpoznat objekt pouze pomocí nalezení určité odlišnosti v barevnosti objektu vůči okolí. Cílem je tedy nalézt nejvhodnější algoritmus pro základní nalezení objektu podle předem dané barvy.

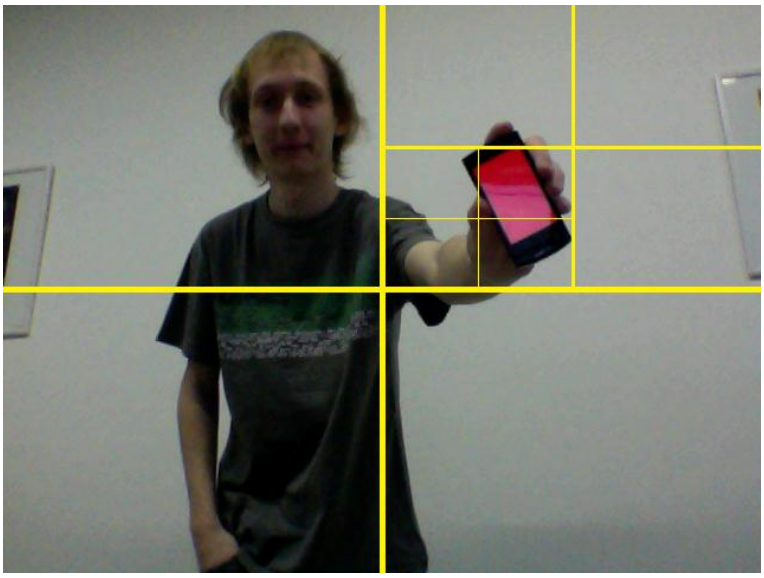
2.3.1 Histogramový filtr

Jako nejvhodnější se může zpočátku jevit využití techniky pro tento účel používané – histogramový filtr. Pojem histogram pochází se statistiky[4]. Představuje počet jednotlivých hodnot barvy vyjádřené v grafu. Na ose X jsou zobrazeny úrovně barvy (0-255) a na ose Y je znázorněn počet hodnot výskytu dané úrovně barvy.



Obrázek 11 Ukázka histogramu

Algoritmus rozdělí obraz na 4 stejně velké oblasti a pro každou z těchto částí vypočítá histogram dané barvy. Vyhodnotí se, která z oblastí má největší průměr hodnot histogramu, a tato část se dále (rekurzivně) rozdělí na 4 stejně velké oblasti. Dále se pokračuje stejným postupem.



Obrázek 12 Ukázka tří průchodů rozdělení obrazu na oblasti

[4] Histogram. In: Wikipedia [online]. 21. 2. 2012 [cit. 2012-03-18]. Dostupné z: http://cs.wikipedia.org/wiki/Histogram

Výsledkem je pozice oblasti, která obsahuje objekt. Pozice je pouze orientační, protože souřadnice nenáleží objektu, ale oblasti okolo. Z tohoto důvodu se pro další zpřesnění používají další filtry.

Implementace histogramového filtru však nezvládala zpracovávat obraz dostatečně rychle. Výpočet histogramu je příliš časově složitý, algoritmus tak není vhodný pro zpracování obrazu v reálném čase (několik snímků za sekundu).

2.3.2 Rozpoznání pomocí barevného kanálu

Při studiu dalších metod jsem objevil na výukové materiály k základům počítačového vidění v rámci umělé inteligence[5]. Zaujaly mě zejména algoritmy pro převod obrázku do černobílého spektra a algoritmy pro detekci hran.

Vzorec pro převod barev do odstínu šedé[6]:

$$\text{Šedá} = R * 0{,}299 + G * 0{,}587 + B * 0{,}114 \quad (4)$$

Vzorec pro hranovou detekci (gradient image)[7]:

$$E = \sqrt{(I_x)^2 + (I_y)^2} \quad (5)$$

Tyto známé algoritmy lze implementovat pro generování nového obrázku, kde jednotlivý pixel není reprezentován třemi bajty RGB, ale pouze jedním bajtem s hodnotou odstínu šedé (0 – černá, 255 – bílá). Vznikne tak černobílý rastr.

Barevný kanál

Tyto vzorce mě dovedli k myšlence vytvořit ze snímku „barevný kanál“. Barevný kanál tvoří souvisle vyplněnou barevnou plochu (rastr), nejedná se však o běžný kanál

---

[5] Grayscale Images. In: Intro to AI [online]. 22.11.2011 [cit. 2012-03-18]. Dostupné z: https://www.ai-class.com/course/video/videolecture/187

[6] Color graphics to grayscale algorithm. In: CodeBack.net [online]. 22.11.2011 [cit. 2012-03-18]. Dostupné z: http://codeback.net/color-graphics-to-grayscale-algorithm

[7] Gradient Images. In: Intro to AI [online]. 22.11.2011 [cit. 2012-03-18]. Dostupné z: https://www.ai-class.com/course/video/videolecture/191

vyjadřující odstín konkrétní barvy (jedna z RGB)[8], ale tuto hodnotu dále upravuji (snižuji) o hodnotu zbylých dvou barev barvového prostoru RGB.

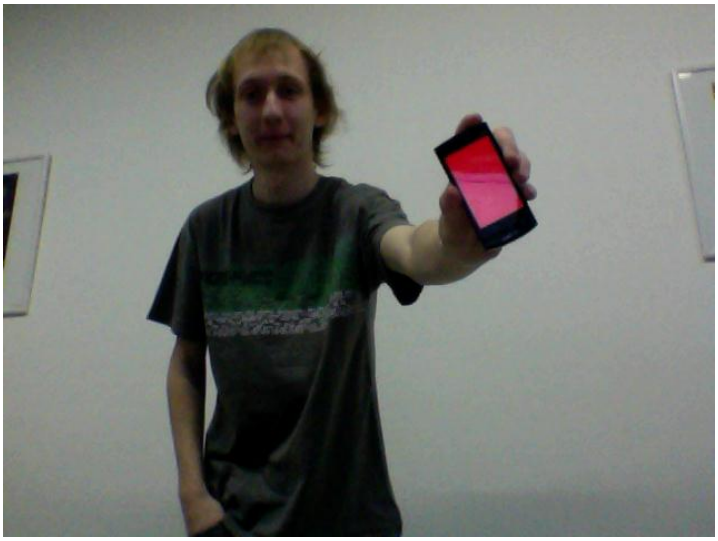
V případě, že vytvářím takto upravovaný kanál červené, velká hodnota kanálu znázorňuje skutečně červené pixely v obraze a nebude znázorňovat například bílé objekty v obraze.

Vzorec pro výpočet červeného kanálu pixelu snížený o hodnoty zelené a modré barvy:

$$C = \sqrt{R^2 - G^2 - B^2} \tag{6}$$

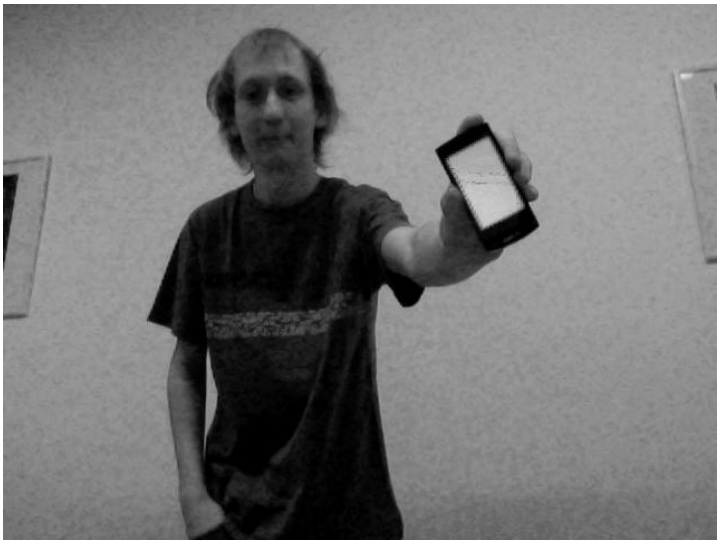
Poznámka: Pokud je hodnota pod odmocninou záporná, mohu s jistotou říct, že daný pixel není červené barvy, neprovádím výpočet odmocniny a za C dosadím nulu (C = 0).

Výsledkem výpočtu je hodnota v intervalu 0 až 255, kde čím větší hodnota, tím větší je pravděpodobnost, že daný pixel znázorňuje červenou barvu. Pixel je označen za červený, je-li pravděpodobnost větší než předvolená citlivost pro červenou barvu.

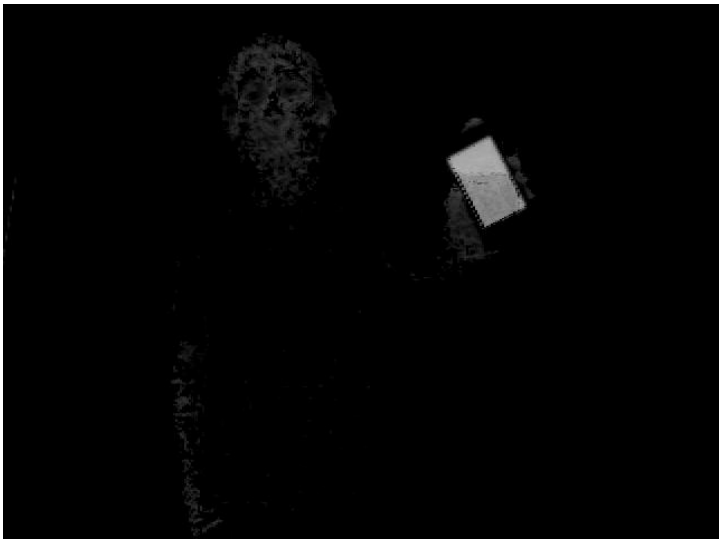


Obrázek 13 Původní RGB snímek webkamery

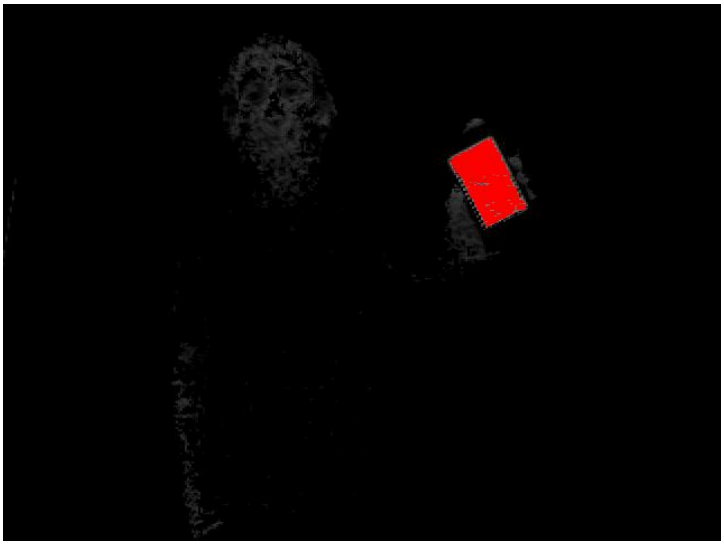
[8] Kanál (digitální obraz). In: Wikipedia [online]. 2001- [cit. 2012-03-18]. Dostupné z: http://cs.wikipedia.org/wiki/Kanál_(digitální_obraz)



Obrázek 14 Klasický kanál červené barvy (v aplikaci nevyužito)



Obrázek 15 Upravený kanál červené (znázornění hodnot C všech pixelů)



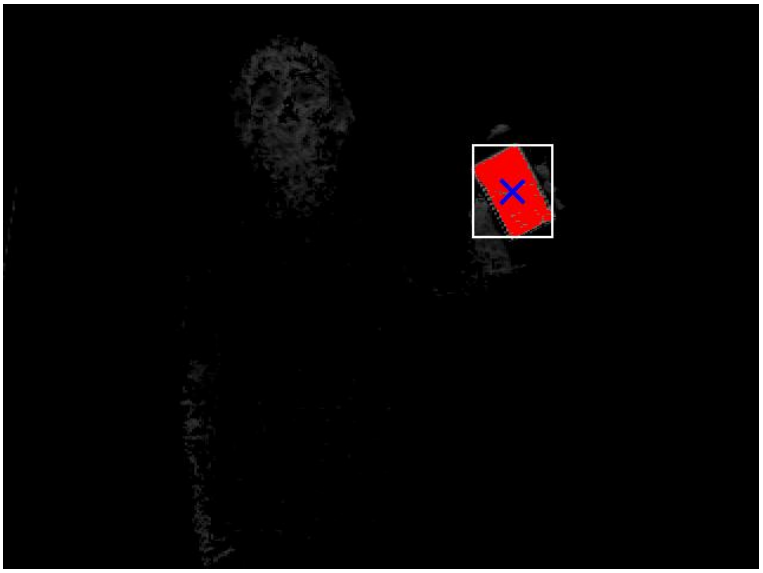
Obrázek 16 Červeně zvýrazněny pixely hodnot C větších než citlivost

Zjištění pozice objektu

Pokud budou nalezeny v kanálu pixely s pravděpodobností větší než předvolená citlivost, je tato množina pixelů ohraničena obdélníkem, a to nalezením nejsevernějšího (červený pixel nejvýše ve snímku), nejvýchodnějšího, nejjižnějšího a nejzápadnějšího pixelu v kanálu.

Poznámka: Pokud se žádná množina červených pixelů v obrázku nevyskytuje, prohlásím, že ve snímku objekt s hledanou barvou není a pokračuji ve vyhodnocování dalšího snímku.

Souřadnice této oblasti pak slouží k určení souřadnic středu objektu (polovina stran obdélníku) a rozměrů objektu (šířka a výška).

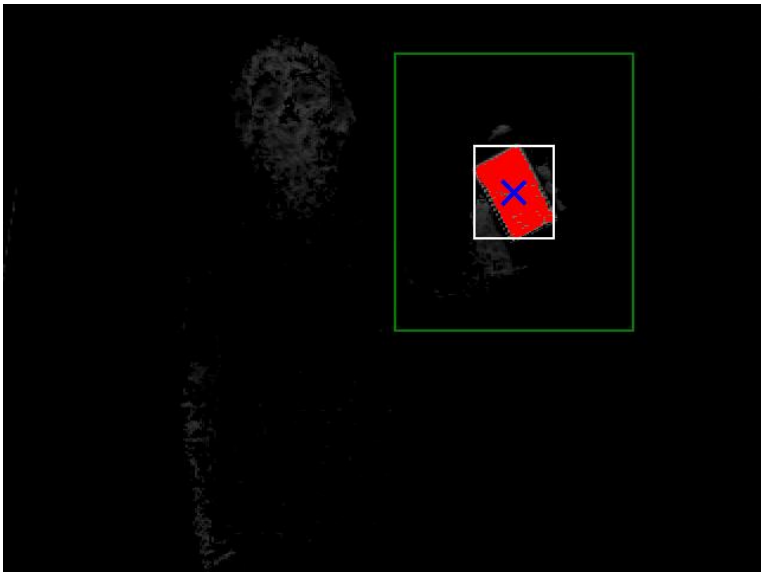


Obrázek 17 Bíle vyznačen obdélník okolo objektu. Modře vyznačen střed objektu.

Vyhodnocování dalších snímků

V případě nalezení objektu ve snímku v následujícím snímku již nevytvářím barevný kanál z celého snímku, ale pouze z okolí sledovaného objektu. Zkoumám tak plochu s menším rozlišením a nalezení objektu je rychlejší.

Při nalezení nového objektu v okolí sledovaného objektu upravím pouze souřadnice pozice objektu na nové. Dále pokračuji ve vyhodnocování dalších snímků. Při nenalezení objektu v okolí prohlásím objekt za ztracený (zmizelý) a v následujícím snímku již opět hledám objekt od začátku (barevný kanál se tvoří z celého snímku).



Obrázek 18 Zeleně vyznačeno okolí objektu pro vyhodnocování dalších snímků

2.4 Srovnání algoritmů

Tabulka 1 Srovnání algoritmů

	Histogramový filtr	**Barevný kanál**
Náchylnost na šum	okolní šum ignoruje	závislá na citlivosti
Časová složitost	velká	zvládá v reálném čase
Spolehlivost	nutnost dalšího zpřesnění	nalezne střed objektu

2.5 Kreslení pomocí rozpoznaného objektu

Získané souřadnice a rozměry rozpoznaného objektu dále využívám pro jednoduché kreslení obrazců přímo do snímků webkamery. O shromažďování grafiky a její následné vykreslení se stará třída Platno, reprezentuje tedy kreslící plátno jako obrazovou vrstvu umístěnou nad snímek z webkamery. Třída uchovává také atributy aktuální šířky a barvy štětce.

Aktuální šířka štětce se určuje podle rozměrů rozpoznaného objektu – použije se vždy menší rozměr z dvojice šířka výška.

Dle souřadnic a šířky štětce se vykresluje okolo snímaného objektu v obraze bílý kruh, zobrazuje tak viditelně pozici a velikost rozpoznaného objektu.

Obrázek 19 Bíle znázorněn štětec na pozici sledovaného objektu

Každý průchod časovače na plátno přidá vyplněný kruh (třída Bod) na pozici snímaného objektu o velikosti šířky štětce. Několik bodů za sebou tak působí dojmem nakreslené čáry. Body jsou vkládány do dynamického pole. Při volbě vymazání kreslícího plátna stačí pouze vyprázdnit obsah tohoto pole bodů.

Obrázek 20 Kreslení pohybem snímaného objektu

2.5.1 Boční panel rychlého ovládání

Třída Platno se také stará o ovládání a vykreslování bočního panelu. Je umístěn v poslední svislé šestině obrazu (vpravo) a dále je rozdělen vodorovně na čtvrtiny.

Pokud je rozpoznána pozice snímaného objektu, která zasahuje do bočního panelu, nevykreslí se nový bod, ale je provedena volba na základě toho, ve které čtvrtině se snímaný objekt nachází.

2.6 Hlavní okno aplikace

Hlavní okno aplikace je realizováno pomocí třídy HlavniOkno. Okno po spuštění zobrazuje dva hlavní prvky aplikace – řádek nabídek a PictureBox.

Dále obsahuje časovač (třída Timer[9]), který spouští v pravidelném intervalu celý proces získání snímku, zpracování, nalezení objektu a vykreslení snímku.

Velikost okna se určí podle rozlišení webkamery.

2.6.1 Obraz webkamery

Zobrazení snímků webkamery je realizováno pomocí komponenty třídy PictureBox[10].

Každý průchod časovače vytvoří bitmapu[11] z dat v paměti (viz. kapitola 2.2.1 Práce s pamětí). Z bitmapy se získá instance třídy Graphics[12], do které je vložena grafika kreslícího plátna (viz. kapitola 2.5 Kreslení pomocí rozpoznaného objektu). Výsledná bitmapa se předá PictureBoxu k vykreslení.

---

[9] Timer Class. MICROSOFT. MSDN Library [online]. 2012 [cit. 2012-03-18]. Dostupné z: http://msdn.microsoft.com/cs-cz/library/system.windows.forms.timer.aspx

[10] PictureBox Class. MICROSOFT. MSDN Library [online]. 2012 [cit. 2012-03-18]. Dostupné z: http://msdn.microsoft.com/cs-cz/library/system.windows.forms.picturebox.aspx

[11] Bitmap Class. MICROSOFT. MSDN Library [online]. 2012 [cit. 2012-03-18]. Dostupné z: http://msdn.microsoft.com/en-us/library/system.drawing.bitmap.aspx

[12] Graphics Class. MICROSOFT. MSDN Library [online]. 2012 [cit. 2012-03-18]. Dostupné z: http://msdn.microsoft.com/en-us/library/system.drawing.graphics.aspx

2.6.2 Režim celé obrazovky

Přepnutí do režimu celé obrazovky (fullscreen) je realizováno manuálně[13] v několika krocích:

1. Skrytí panelu nabídek
2. Odstranění okolních grafických prvků okna
3. Maximalizace okna
4. Umístění nad všechny okna
5. Roztáhnutí PictureBoxu přes celou obrazovku se zachováním poměru obrazu
6. Umístění PictureBoxu na střed

Obraz se roztáhne správně při otočení obrazovky na šířku i na výšku. Černé pruhy okolo jsou dány černým pozadím okna. Přepnutí zpět do okna se děje opačným způsobem.

Obrázek 21 Režim celé obrazovky na širokoúhlém monitoru

[13] C# a prohlížeč obrázků.. In: Fórum Živě.cz [online]. 12. 12. 2007 [cit. 2012-03-18]. Dostupné z: http://forum.zive.cz/viewtopic.php?f=922&t=1020045

2.7 Seznam tříd aplikace

Třídy a jejich vlastnosti jsou podrobně okomentovány ve zdrojových kódech.
Stručný výčet tříd:

- Program
- HlavniOkno
- VyberOkno
- CitlivostOkno
- Obrazek
- Objekt
- BarevneRozpoznavany
- Platno
- Bod

Třída Program je main metodou aplikace. Je zavolána jako první při spuštění programu. Spouští hlavní okno.

Třída HlavniOkno představuje hlavní okno aplikace. Stará se o připojení webkamery, vykreslení plátna a práci s pamětí.

Třída VyberOkno představuje dialog pro výběr webkamery. Pokud je rozpoznána pouze jedna webkamera, je vybrána a okno se uzavře.

Třída CitlivostOkno představuje okno pro změnu citlivosti pro rozpoznávání jednotlivých barev.

Třída Obrazek reprezentuje datovou strukturu získaného snímku jako objekt (rozšíří vlastnosti pole bajtů o další vlastnosti).

Třída Objekt je abstraktní třídou reprezentující snímaný objekt v obraze. Deklaruje základní vlastnosti pro zjištění pozice objektu, jeho šířky a výšky. Při nalezení objektu se uchovává jedna instance objektu, pohybem se objektu mění atributy a zmizením objektu v obraze instance zaniká.

Třída BarevneRozpoznavany je potomkem třídy Objekt a reprezentuje rozpoznávání objektu v obraze na základě barevného kanálu.

Třída Platno reprezentuje kreslící plátno jako obrazovou vrstvu umístěnou nad snímek z webkamery.

Třída Bod reprezentuje jednotlivý nakreslený bod na kreslícím plátně.

ZÁVĚR

Výsledná aplikace splňuje všechny požadavky na způsob zpracování.

Aplikace se umí připojit k webkameře a snímat obraz v reálném čase. Podařilo se mi najít vhodný algoritmus pro rozpoznání objektu v obraze na základě barevnosti a dále zaznamenávání jeho pohybu. Rozpoznává objekty tří barev – červené, zelené a modré.

Implementaci algoritmu demonstruji na jednoduché aplikaci, která umožňuje kreslit do obrazu webkamery pomocí pohybu snímaného objektu před webkamerou. Lze měnit barvu štětce intuitivně pomocí bočního panelu nebo pomocí výběrového nástroje, který umožňuje vybrat libovolnou RGB barvu z palety barev. Tloušťka štětce se mění dynamicky závisle na zjištěné velikosti snímaného objektu.

Díky programování aplikace jsem se seznámil s jazykem C# a objevil jeho možnosti v práci s grafikou.

SEZNAM POUŽITÉ LITERATURY

(1) Kinect. In: Wikipedia [online]. 12.3.2012 [cit. 2012-03-18].
Dostupné z: http://en.wikipedia.org/wiki/Kinect

(2) MCSystem. Firielentul [online]. 2011 [cit. 2012-03-18].
Dostupné z: http://firielentul.canis-ebrius.com/MCSystem/index.html

(3) Marshal Class. MSDN Library [online]. 2012 [cit. 2012-03-18]. Dostupné z:
http://msdn.microsoft.com/cs-cz/library/system.runtime.interopservices.marshal.aspx

(4) Histogram. In: Wikipedia [online]. 21. 2. 2012 [cit. 2012-03-18].
Dostupné z: http://cs.wikipedia.org/wiki/Histogram

(5) Grayscale Images. In: Intro to AI [online]. 22.11.2011 [cit. 2012-03-18].
Dostupné z: https://www.ai-class.com/course/video/videolecture/187

(6) Color graphics to grayscale algorithm. In: CodeBack.net [online]. 22.11.2011
[cit. 2012-03-18]. Dostupné z: http://codeback.net/color-graphics-to-grayscale-algorithm

(7) Gradient Images. In: Intro to AI [online]. 22.11.2011 [cit. 2012-03-18].
Dostupné z: https://www.ai-class.com/course/video/videolecture/191

(8) Kanál (digitální obraz). In: Wikipedia [online]. 2001- [cit. 2012-03-18].
Dostupné z: http://cs.wikipedia.org/wiki/Kanál_(digitální_obraz)

(9) Timer Class. MICROSOFT. MSDN Library [online]. 2012 [cit. 2012-03-18].
Dostupné z: http://msdn.microsoft.com/cs-cz/library/system.windows.forms.timer.aspx

(10) PictureBox Class. MICROSOFT. MSDN Library [online]. 2012 [cit. 2012-03-18].
Dostupné z: http://msdn.microsoft.com/cs-cz/library/system.windows.forms.picturebox.aspx

(11) Bitmap Class. MICROSOFT. MSDN Library [online]. 2012 [cit. 2012-03-18].
Dostupné z: http://msdn.microsoft.com/en-us/library/system.drawing.bitmap.aspx

(12) Graphics Class. MICROSOFT. MSDN Library [online]. 2012 [cit. 2012-03-18].
Dostupné z: http://msdn.microsoft.com/en-us/library/system.drawing.graphics.aspx

(13) C# a prohlížeč obrázků.. In: Fórum Živě.cz [online]. 12. 12. 2007 [cit. 2012-03-18].
Dostupné z: http://forum.zive.cz/viewtopic.php?f=922&t=1020045

SEZNAM OBRÁZKŮ

Obrázek 1 Okno pro výběr webkamery 7

Obrázek 2 Rozložení hlavního okna aplikace 8

Obrázek 3 Rozevřená nabídka Rozpoznávat 8

Obrázek 4 Rozevřená nabídka Plátno 8

Obrázek 5 Ukázka jednoduché kresby vytvořené pohybem 9

Obrázek 6 Okno pro změnu barvy štětce 10

Obrázek 7 Popis položek bočního panelu 10

Obrázek 8 Okno pro změnu citlivosti 11

Obrázek 9 Rozevřený seznam webkamer pro výběr 12

Obrázek 10 Znázornění uspořádání pole (zalomeno na řádky) 14

Obrázek 11 Ukázka histogramu 15

Obrázek 12 Ukázka tří průchodů rozdělení obrazu na oblasti 15

Obrázek 13 Původní RGB snímek webkamery 17

Obrázek 14 Klasický kanál červené barvy (v aplikaci nevyužito) 18

Obrázek 15 Upravený kanál červené (znázornění hodnot C všech pixelů) 18

Obrázek 16 Červeně zvýrazněny pixely hodnot C větších než citlivost 18

Obrázek 17 Bíle vyznačen obdélník okolo objektu. Modře vyznačen střed objektu. 19

Obrázek 18 Zeleně vyznačeno okolí objektu pro vyhodnocování dalších snímků 20

Obrázek 19 Bíle znázorněn štětec na pozici sledovaného objektu 21

Obrázek 20 Kreslení pohybem snímaného objektu 21

Obrázek 21 Režim celé obrazovky na širokoúhlém monitoru 23

SEZNAM TABULEK

Tabulka 1 Srovnání algoritmů 20

SEZNAM PŘÍLOH

Přílohou je CD obsahující spustitelný soubor s aplikací, zdrojové kódy a dokumentaci v PDF.