VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

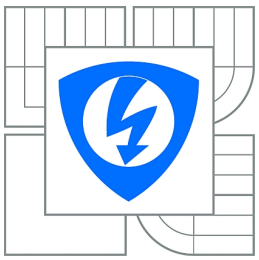
KOREKCE JASOVÝCH CHYB LED OBRAZOVEK S VYSOKÝM ROZLIŠENÍM
LUMINANCE ERROR CORRECTION OF HIGH DEFINITION LED SCREENS.

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Michal Komloši**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Adam Olejár**
SUPERVISOR

BRNO 2016



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Bakalářská práce

bakalářský studijní obor
Teleinformatika

Student:	Michal Komloši	***ID:***	164307
Ročník:	3	***Akademický rok:***	2015/2016

NÁZEV TÉMATU:

Korekce jasových chyb LED obrazovek s vysokým rozlišením

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte současné principy zobrazování LED obrazovek s vysokým rozlišením, u nichž dochází k jasovým zkreslením v důsledku tepelné roztažnosti jednotlivých bloků z nichž je obrazovka zkonstruována.
Navrhněte možné řešení problému a popište ho: vytvořte aplikaci, která bude umožňovat ručně korigovat chyby v statickém obraze formou softwarového nastavení jasové korekce v příslušné části obrazu. Aplikace následně umožní přehrávat video alebo snímky s použitím parametrů jasové korekce. Doporučeným nástrojem pro implementaci jsou knihovny OpenCV a prostředí MS VisualC ++.

DOPORUČENÁ LITERATURA:

[1] GONZALEZ R.C., WOODS R.E.: Digital Image Processing. New Jersey: Prentice-Hall, 2002.
[2] PRATA S.: Mistrovství v C++, Computer Press, Brno 2004, ISBN 80-251-0098-7.

Termín zadání:	1.2.2016	***Termín odevzdání:***	1.6.2016

Vedoucí práce: Ing. Adam Olejár
Konzultanti bakalářské práce:

doc. Ing. Jiří Mišurec, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

ABSTRAKT

Táto bakalárska práca popisuje a v prílohe ponúka možné softvérové riešenie jasových chýb, ktoré vznikajú na HD LED obrazovkách v dôsledku teplotnej rozťažnosti kabinetov, z ktorých sa LED obrazovky skladajú.

KĽÚČOVÉ SLOVÁ

LED, obrazovka, jas, jasové chyby, aplikácia, objektovo orientované programovanie

ABSTRACT

This bachelor's thesis describes and offers possible software solution of luminance errors, that are made on HD LED screens due to the temperature dependence of LED cabinets, which the LED screens are consist of.

KEYWORDS

LED, screen, brightness, brightness errors, application, object oriented programming

KOMLOŠI, Michal *Korekce jasových chyb LED obrazovek s vysokým rozlišením*: bakalárska práca. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2016. 45 s. Vedúci práce bol Ing. Adam Olejár,

PREHLÁSENIE

Prehlasujem, že som svoju bakalársku prácu na tému „Korekce jasových chyb LED obrazovek s vysokým rozlišením“ vypracoval(a) samostatne pod vedením vedúceho bakalárskej práce, využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor(ka) uvedenej bakalárskej práce ďalej prehlasujem, že v súvislosti s vytvorením tejto bakalárskej práce som neporušil(a) autorské práva tretích osôb, najmä som nezasiahol(-la) nedovoleným spôsobom do cudzích autorských práv osobnostných a/nebo majetkových a som si plne vedomý(-á) následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona č. 121/2000 Sb., o právu autorském, o právoch súvisejúcich s právom autorským a o zmeně niektorých zákonov (autorský zákon), vo znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhé, hlavy VI. diel 4 Trestného zákoníka č. 40/2009 Sb.

Brno ....

podpis autora(-ky)

POĎAKOVANIE

Rád by som poďakoval vedúcemu bakalárskej práce pánovi Ing. Adamovi Olejárovi, za odborné vedenie, konzultácie, trpezlivosť a podnetné návrhy k práci.

Brno . . . .

podpis autora(-ky)

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic
http://www.six.feec.vutbr.cz

POĎAKOVANIE

Výzkum popsaný v tejto bakalárskej práci bol realizovaný v laboratóriách podporených projektom SIX; registračné číslo CZ.1.05/2.1.00/03.0072, operačný program Výzkum a vývoj pro inovace.

Brno . . . . . .

. . . . . .
podpis autora(-ky)

OBSAH

ZOZNAM OBRÁZKOV

ÚVOD

Stretnúť sa so zariadením, ktorého súčasťou je LED,t.j. Luminiscenčná dióda, nie je v roku 2016 naozaj nič výnimočné. Práve naopak, LED technológia sa stala veľmi bežnou a obľúbenou v rôznych technických odvetviach. Napríklad v automobilovom priemysle sa v posledných rokoch stali LED svetlá neodmysliteľnou súčasťou áut nielen prémiových značiek. Podobne, v reklamnom priemysle sú veľmi často používané RGB LED v obrazovkách určených na vonkajšie alebo vnútorné použitie.

Diódy LED zaznamenávajú podobne rýchle zlepšovanie svojich vlastností a znižovanie ceny, ako svojho času mikroprocesory. Majú však aj svoje nevýhody. Hlavným problémom v dosahovaní priaznivého pomeru ceny a svetla je cena kryštálu základného polovodiča. Ďalšou nevýhodou LED je závislosť jej parametrov na teplote okolia.

Na vysokú úroveň zobrazovania statického obrazu alebo videa sa dostali LED obrazovky. Sú veľmi často používané na rôznych spoločenských udalostiach, prezentáciách či v reklame. Väčší koncert alebo športovú udalosť si bez LED obrazovky azda ani nevieme predstaviť. Avšak, aj tento druh zobrazovania má svoje nedokonalosti. U LED obrazovky s vysokým rozlíšením totiž dochádza k jasovým chybám, ktoré sú zapríčinené teplotnou rozťažnosťou jednotlivých panelov, z ktorých sa obrazovka skladá.

Praktická časť tejto Bakalárskej práce popisuje navrhnutú softvérovú aplikáciu, ktorej účelom je manuálne kalibrovať jasové chyby na LED obrazovkách, ktoré vzniknú z dôvodu tejto teplotnej závislosti. Chyba sa javí na obrazovke ako miesto s odlišným jasom, než aký by mal byť na danej pozícií zobrazený. Kalibráciu je možné vykonávať v statickom obraze a výslednú úpravu program umožňuje aplikovať aj na užívateľom vybrané video.

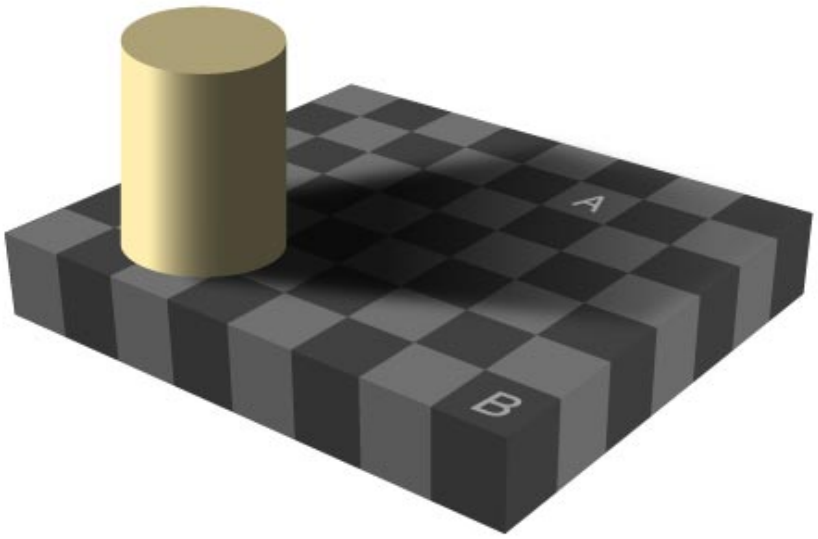
1 TEORETICKÁ ČASŤ

V prvej kapitole Bakalárska práca popisuje vnímanie jasu, farebné modely RGB, HSV(HSB), HSL a LED obrazovky. Ďalej sa bližšie zameriava na oblasti použitia LED obrazoviek, ich funkciu a konštrukciu. Od podkapitoly 1.6 je popisovaná teória tvorby softvérovej aplikácie a nástroje, ktoré boli použité na praktické vytvorenie kalibračného programu podľa zadania.

1.1 Vnímanie jasu

Celkom prirodzene je jas určitého objektu ovplyvnený jeho povrchom a intenzitou svetla, ktoré objekt odráža, alebo ktoré samotný objekt vyžaruje. Čím intenzívnejšie bude svetlo, tým vyšší bude jas objektu. Okrem týchto fyzikálnych veličín vstupuje do hry aj subjektívna vlastnosť ľudského videnia. Subjektívny jas oko totiž vyhodnocuje jednak v kontexte s jasom okolia ale aj v kontexte farby.

Najlepšie asi všetko ukáže príklad jednoduchého optického klamu, ktorý interpretuje rozdiel medzi subjektívnym vnímaním a skutočným množstvom svetla, ktoré reálne predmet odráža alebo vyžaruje. Rovnaký objekt, obklopený tmavými tónmi, sa nám bude totiž zdať jasnejší, než objekt, ktorý je obklopený svetlými predmetmi.2



Obr. 1.1: *Jasový klam. Políčka označené A a B majú úplne rovnaký jas aj farbu*

1.2 RGB model

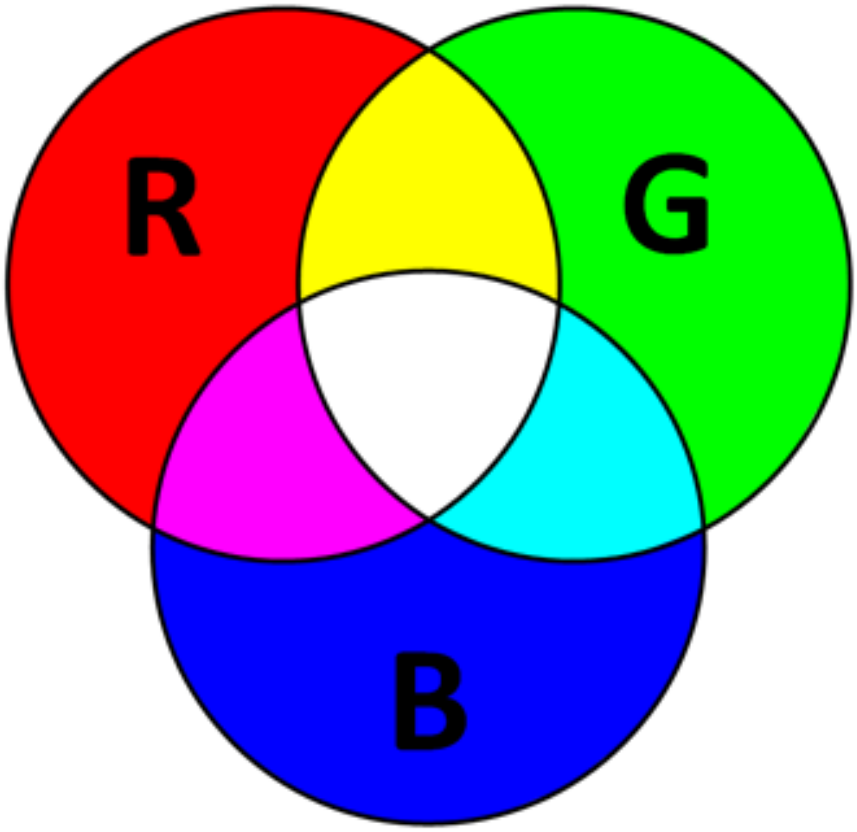
Ide o aditívny farebný model, pri ktorom svetlo želanej farby vzniká zmiešavaním červeného, zeleného a modrého svetla (**R**ed,**G**reen,**B**lue) s vhodnou intenzitou. Používa sa vo farebnej fotografii, osvetľovacej a zobrazovacej technike. Citlivosť ľudského oka je na jednotlivé farebné tóny rôzna.

Jasová rovnica:

$$U_Y = 0,3U_R + 0,59U_G + 0,11U_B$$

Hodnota U_Y udáva intenzitu jasového signálu.

V prípade najbežnejšej 8-bitovej kvantizácie, umožňujúcej 256 diskrétnych hodnôt, môžu jednotlivé farebné zložky nadobúdať celočíselné hodnoty z intervalu 0–255. Čierna farba ma v modeli RGB hodnotu R:0 G:0 B:0. Naopak, pre bielu farbu sú hodnoty intenzity farebných zložiek maximálne – R:255 G:255 B:255.



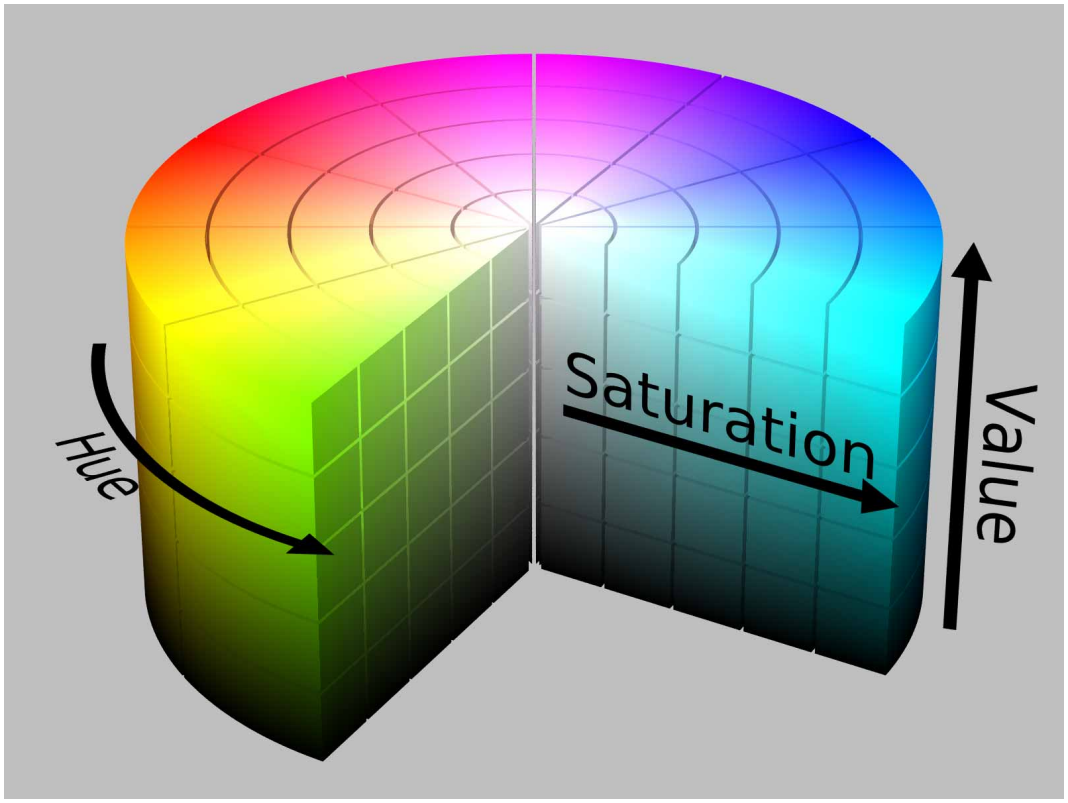
Obr. 1.2: *Model RGB*

1.3 HSV farebný model

HSV (**H**ue, **S**aturation, **V**alue), tiež známy ako HSB (**H**ue, **S**aturation, **B**rightness), je farebný model, ktorý vytvoril v roku 1978 **Alvu Ray Smith**. Tento farebný model najviac zodpovedá ľudskému vnímaniu farieb. Pozostáva z troch zložiek (nie sú to základné farby):

1. **Hue**– farebný tón, prevládajúci. Čiže odtieň– farba odrazená alebo prechádzajúca objektom. Meria sa ako poloha na štandardnom farebnom kruhu (0 ° až 360 °). Všeobecne sa odtieň označuje názvom farby.
2. **Saturation**– sýtosť farby, prímes inej farby. Niekedy tiež chroma, sila alebo čistota farby, predstavuje množstvo šedej v pomere k odtieňu, meria sa v percentách od 0% (šedá) do 100% (plne sýta farba). Na farebnom kruhu vzrastá sýtosť od stredu k okrajom. Napríklad červená s 50% sýtosťou bude ružová.
3. **Value**– hodnota jasu, množstvo bieleho svetla. Relatívna svetlosť alebo tmavosť farby. Jas vyjadruje, koľko svetla farba odráža.

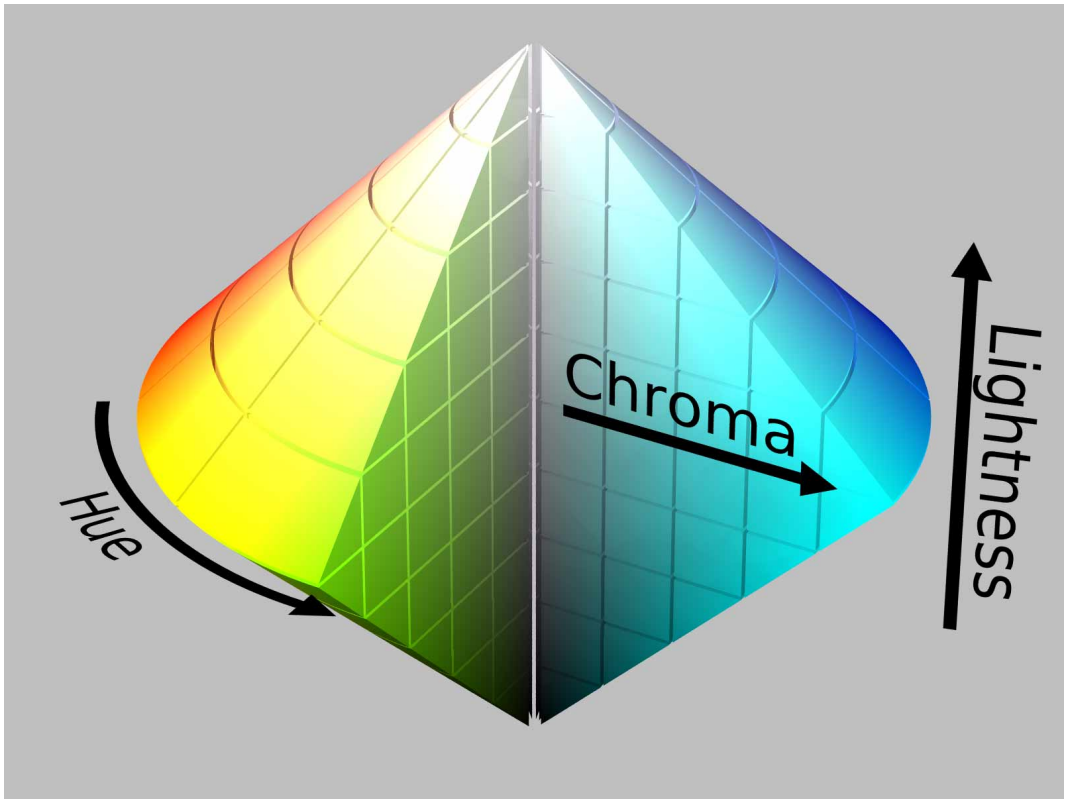
14, 15, 16



Obr. 1.3: *Farebný model HSV*

1.4 HSL farebný model

Skratka HSL je vytvorená z vlastností, ktoré tento model tvoria – „**H**ue, **S**aturation, **L**ightness“. Takže rovnako, ako farebný model HSB, resp. HSV, obsahuje vlastnosti Hue a Saturation (odtieň a sýtosť). Odlišujú sa iba v tom, že prvý model používa jas (**Brightness**) a druhý používa svetlosť (**Lightness**). V modeli HSV (HSB) dostaneme čiernu farbu tým, že nastavíme jas na nulu, a bielu tak, že jas nastavíme na maximálnu hodnotu (nezávisle od sýtosti). V modeli HSL dosiahneme čiernu, ak je svetlosť aj sýtosť 0 a bielu, ak je svetlosť aj sýtosť maximálna. 14, 16



Obr. 1.4: *Farebný model HSL*

1.5 LED obrazovky

Na vrchol dokonalosti, v oblasti zobrazovacích jednotiek na báze LED, patria plnofarebné veľkoplošné LED obrazovky. Tie dokážu s využitím efektu aditívneho miešania svetla vhodne zoskupených RGB LED, vyžarujúcich v červenej, zelenej a modrej oblasti spektra, vytvoriť v podstate akúkoľvek výslednú farbu svetla. Ich jas pri tom dosahuje také hodnoty, že môžu byť úspešne využívané aj počas intenzívneho denného osvetlenia (iné, ako zobrazovacie jednotky LED to dnes neumožňujú).

Zobrazovacia technika LED sa tak stáva účinným prostriedkom poskytovania informácií alebo zábavy (športové udalosti, koncerty a pod.) až stovkám tisíc osôb súčasne pred jedinou veľkoplošnou obrazovkou. Takéto obrazovky, s plochou až 200 metrov štvorcových, nachádzajú široké uplatnenie napríklad v reklame, pri významných udalostiach, alebo môžu byť užitočné k predávaniu pokynov verejnosti v rámci systému civilnej obrany.

Plnofarebné veľkoplošné obrazovky LED, určené pre použitie vo vonkajších podmienkach, dosahujú jas 5000 až 8000 cd/m^2; v budúcnosti sa predpokladajú jasy väčšie než 10 000 cd/m^2.

1.5.1 Hlavné oblasti použitia LED obrazoviek

LED obrazovky môžeme v zásade rozdeliť do dvoch skupín podľa použitia na:

1. **Exteriérové LED obrazovky**
   - Vonkajšia reklama
   - Digitálne billboardy
   - Vonkajšie kultúrne akcie (koncerty, festivaly alebo súťaže)
   - Dopravné ukazovatele a pod.
2. **Interiérové LED obrazovky**
   - Konferencie a prezentácie
   - Interiérová reklama
   - Živé video–prenosy
   - 3D video a pod.

1.5.2 Princíp fungovania LED obrazoviek

LED obrazovka je zobrazovacie zariadenie, ktorého aktívnym prvkom sú svetelné diódy, poskladané do siete (matice), pokrývajúce celú plochu obrazovky.

LED obrazovka je postavená na princípe aditívneho miešania farieb, keď každý jednotlivý plnofarebný bod obrazovky tvorí trojica LED - červená, zelená a modrá. Pri sledovaní veľkoplošnej LED obrazovky z určitej vzdialenosti farebný svit všetkých troch LED splynie vďaka obmedzenej rozlišovacej schopnosti ľudského oka, a pozorovateľ ho vníma ako jeden farebný bod. Čím väčší je rozstup medzi jednotlivými LED, tým väčšia je aj minimálna pozorovacia vzdialenosť.

LED sú osadené do plochy obrazovky s tienidlami. Tienidlá sú výstupky tvoriace striešku nad LED, ktorá zabraňuje dopadu slnečného žiarenia a chráni pred mechanickým poškodením. Čierna farba podkladu obrazovky zaisťuje optimálne podmienky pre maximálne využitie farebnej škály a intenzity vyžarovaného svetla. Rozdielnou intenzitou svitu jednotlivých LED možno docieliť zobrazenie až 68 miliárd

farieb.

Obrazové dáta sú produkované v počítači prostredníctvom riadiacej aplikácie, ktorá každej LED priraďuje odlišnú intenzitu svitu. Táto informácia je zasielaná do riadiacej jednotky vo vnútri samotnej obrazovky. Všetko sa deje v reálnom čase pri obnovovacej frekvencii 600 Hz, teda 600 krát za sekundu. Vysoká obnovovacia frekvencia zaručuje, že obraz LED obrazovky zaznamenaný akoukoľvek videotechnikou nebliká, čo je absolútne nevyhnutné pri TV prenosoch zo športových podujatí alebo koncertov.

1.5.3 Konštrukčné zloženie a časti veľkoplošných LED obrazoviek

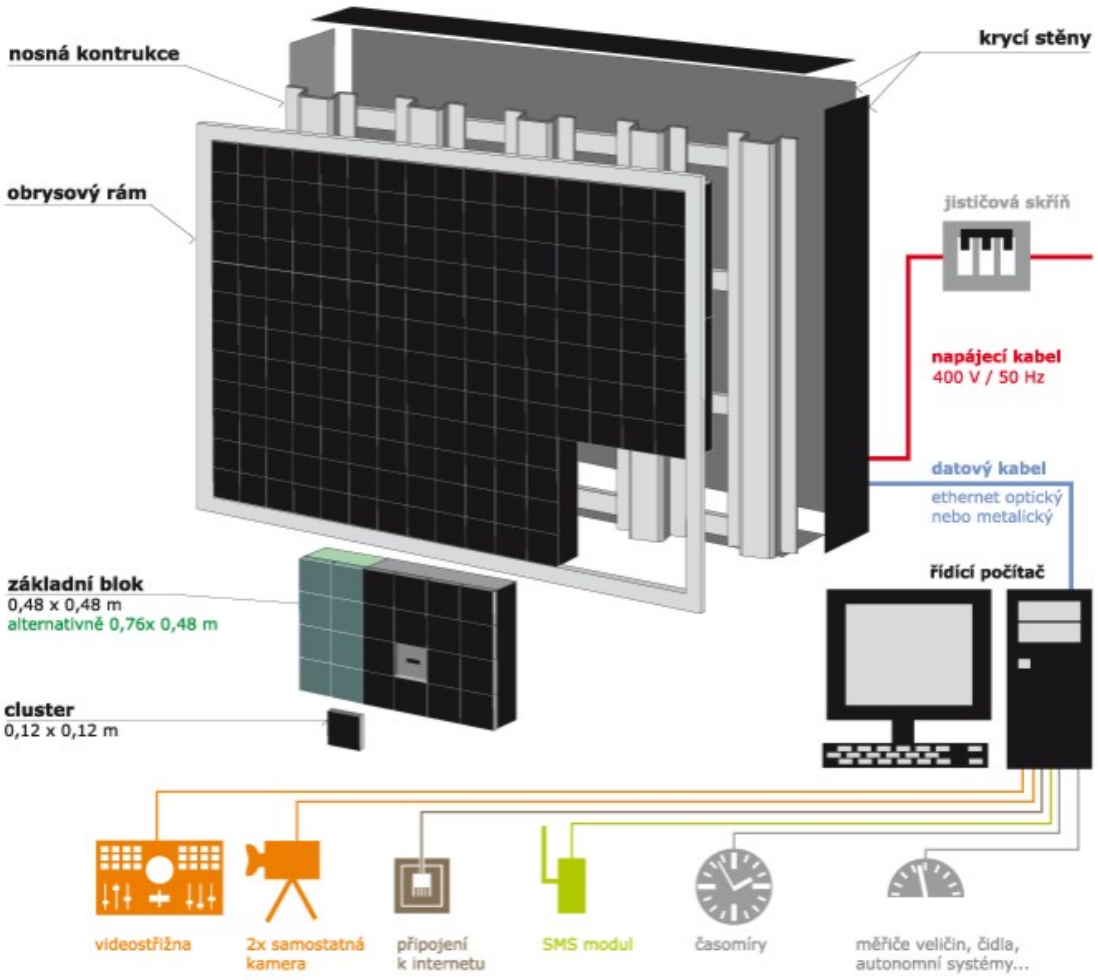
Cluster je najmenší konštrukčný prvok veľkoplošných LED obrazoviek. Je to vodotesný a prachotesný segment, ktorý obsahuje LED usporiadané do matice, riadiacu elektroniku a obal z odolného plastu. Cluster má zdvojený konektor so základným blokom, čím je eliminovaná strata dát pre LED. Poškodený cluster sa dá ľahko vymeniť aj za chodu obrazovky.

Základný blok je základný funkčný prvok veľkoplošných LED obrazoviek. Skladá sa zo základnej dosky s riadiacou elektronikou, primárneho i záložného napájacieho zdroja a pevného kovového obalu. Jeho čelná strana je vysadená 16 clustermi (4x4). Zo základných blokov možno zložiť veľkoplošnú LED obrazovku ľubovoľného tvaru aj veľkosti - jediným obmedzením je rozmer základného bloku 48 x 48 cm.

Nosná konštrukcia spája základné bloky do jedného mechanicky odolného celku. Možno zvoliť pevnú konštrukciu, ktorá je vhodná pre trvalé inštalácie napr. na fasády domov, alebo rozoberateľnú konštrukciu, ktorá umožňuje meniť tvar obrazovky podľa potreby (moduly o rozmere cca 1x1 m možno ľubovoľne pospájať do požadovaného tvaru).

Riadiaci počítač spracováva obrazové informácie z rozmanitých vstupov pomocou aplikácie `ViewLab` a následne ich zasiela riadiacej elektronike obrazovky. Funkciu riadiaceho počítača zvládne štandardné PC vybavené kartami pre zber video signálu (S-Video i kompozitné video), digitálne a analógové TV, satelitné vysielanie a pod. Aplikácia ViewLab umožňuje kombinovať video signál z kamier s grafikou a textami, reklamnými oznámeniami, informačnými blokmi vrátane online spravodajstva z internetu alebo dátami z SMS hlasovania.

Dátový kábel zabezpečuje komunikáciu medzi riadiacim počítačom a veľkoplošnou LED obrazovkou. Používa sa buď klasický metalický sieťový kábel, alebo kábel optický pre vzdialené riadenie (v rádoch kilometrov).3, 8



Obr. 1.5: *Konštrukcia veľkoplošnej LED obrazovky (prevzaté od 8)*

1.5.4 HD LED Obrazovky

HD LED obrazovky sú svojou konštrukciou podobné rentálovým obrazovkám. Ľahko sa skladajú vďaka špeciálnej konštrukcií navrhnutej pre rýchlu inštaláciu a demontáž. Základný rozdiel je v ich rozlíšení, kde najjemnejšia rozteč diód je iba 1,488 mm, čo u HD LED obrazoviek ponúka neuveriteľne realistický obraz aj z úplnej blízkosti. Použitie týchto obrazoviek je všade tam, kde je kladený dôraz na vierohodný obraz v čo najväčšom rozlíšení a realistickom podaní farieb. 9 Pre vnútorné LED obrazovky sa takmer výhradne používa technológia SMD. Rozostup diód sa pohybuje od 2,6mm až po 10mm. Možná je tiež virtuálizácia obrazovky, vďaka ktorej môžeme dosiahnuť dvojnásobný počet pixelov. Obrazovky sú pripravené aj pre 3D zobrazenie.

Obr. 1.6: *LED Display P1.9 (prevzaté od 10)*

1.5.5 Popis riešeného problému

Na LED obrazovkách s vysokým rozlíšením môže vznikať jasové skreslenie v dôsledku teplotnej rozťažnosti jednotlivých blokov (kabinetov), z ktorých je obrazovka skonštruovaná. Táto teplotná závislosť spôsobuje zmenšovanie alebo zväčšovanie vzdialenosti jednotlivých vzájomne susediacich kabinetov obrazovky, čo má za následok chyby zobrazenia, ktoré sa javia ako miesta (čiary) s odlišnou veľkosťou jasu.

Vhodné riešenie tohto problému môže byť manuálna kalibrácia miesta s chybou pomocou softvérovej aplikácie, ktorá dokáže zmeniť hodnoty jasu pixelov zobrazovaného obrázku presne na miestach, kde sa toto skreslenie nachádza. Výhodou softvérového riešenia oproti hardvérovému je v jeho jednoduchosti, cene alebo použiteľnosti na rôzne LED obrazovky (rozlíšenie, rozmery apod.) s rovnakým problémom.

1.6 Microsoft Visual Studio

Microsoft Visual Studio je vývojové prostredie (IDE) od Microsoftu. Môže byť použité pre vývoj konzolových aplikácií a aplikácií s grafickým rozhraním spolu s aplikáciami Windows Forms, webovými stránkami, webovými aplikáciami a webovými službami ako v strojovom kóde, tak v riadenom kóde na platformách Microsoft Windows, Windows Mobile, Windows CE, .NET, .NET Compact Framework a Microsoft Silverlight.

Visual Studio obsahuje editor kódu podporujúce IntelliSense a refaktorovanie. Integrovaný debugger pracuje ako na úrovni kódu, tak aj na úrovni stroja. Ďalšie vstavané nástroje zahŕňajú designer formulárov pre tvorbu aplikácií s GUI, designer webu, tried a databázových schém. Je možné pridávať rozšírenia, čo vylepšuje funkčnosť na takmer každej úrovni - od doplnenia podpory pre verzovacie systémy (ako Subversion a Microsoft Team Foundation Server) po nové nástroje ako editory a vizuálne dizajnéry pre doménové špecifické jazyky alebo nástroje na ďalšie aspekty návrhu programu (ako klient Team Foundation Serveru team Explorer).

Visual Studio podporuje jazyky prostredníctvom jazykových služieb, čo umožňuje, aby editor kódu a debugger podporoval akýkoľvek programovací jazyk. Medzi vstavané jazyky patrí C / C ++ (použitím Visual C ++), VB.NET (použitím Visual Basic .NET) a C# (použitím Visual C#). Podpora ďalších jazykov ako Oxygene, F#, Python a Ruby spolu s ostatnými môže byť pridaná jazykovými službami, ktoré musia byť nainštalované zvlášť. Tiež je podporované XML / XSLT, HTML / XHTML, JavaScript a CSS. Existujú aj verzie Visual Studia pre určitý jazyk, ktoré užívateľovi poskytujú obmedzenejšie jazykové služby. Tieto individuálne balíčky sú Microsoft Visual Basic, Visual J#, Visual C# a Visual C ++

1.7 Programovacie jazyky

Na riešenie zadania tejto Bakalárskej práce som sa rozhodol použiť objektovo orientovaný jazyk. Medzi objektovo orientované jazyky patria napríklad Java, Object Pascal alebo C#. Každý z týchto jazykov má svoje výhody a často záleží len na posúdení, schopnostiach a skúsenostiach programátora, ktorý jazyk si pre svoj projekt vyberie.

Pre túto Bakalársku prácu bol vybratý jazyk C#, kvôli jeho veľkej komunitnej podpore (vďaka čomu sa dá na internete nájsť veľa riešení rôznych problémov), má obrovskú podporu na platforme .NET Framework 4.5, ktorú som využíval, a mám s týmto jazykom najviac skúseností v porovnaní s inými.

1.7.1 C#

C# je elegantný, objektovo orientovaný jazyk, ktorý umožňuje vytvárať rôzne bezpečné a robustné aplikácie spustené v .NET Framework. C# umožňuje vytvoriť tradičnú Windows aplikáciu, webové služby XML, databázové aplikácie, v súčastnosti veľmi obľúbene MVC (Model, View, Controller) aplikácie a veľa ďalšieho.

Syntax jazyka C# je vysoko výrazová, ale je tiež celkom jednoduchá na pochopenie a naučenie. Užívatelia, ktorí niekedy pracovali v jazyku C/C++ alebo Java sú schopní byť produktívni v jazyku C# vo veľmi krátkej dobe. C# syntax zjednodušuje veľa zložitostí C++ a obsahuje výkonné funkcie, ako sú typy s možnou NULL hodnotou, výpočty, delegáti, lambda výrazy a priamy prístup do pamäte, ktoré nie sú k dispozícii v jazyku Java.

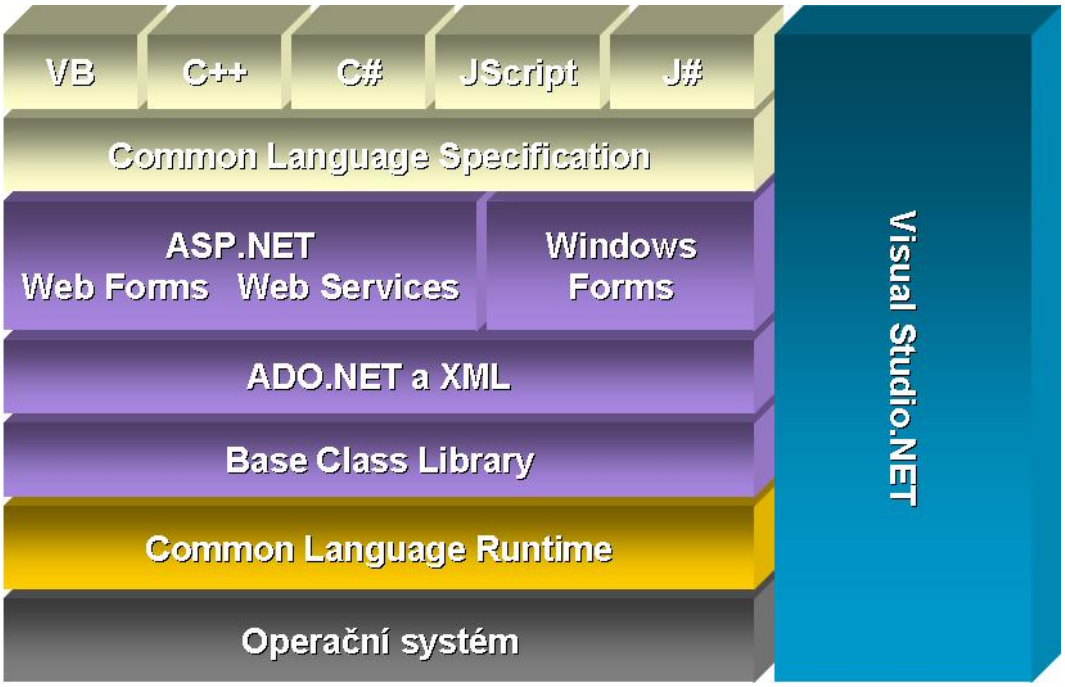
Ako objektovo orientovaný jazyk C# podporuje koncepty zapuzdrenie, dedičnosť a polymorfizmus. Všetky premenné a metódy (vrátane Main), sú zapuzdrené v rámci definície triedy. Triedy môžu dediť priamo od jednej nadradenej triedy, ale môžu implementovať ľubovoľný počet rozhraní. Prepisovanie metódy vyžaduje override (kľúčového slovo), aby sa zabránilo náhodnému prepisovaniu metódy nadradenej triedy.5

1.8 .NET Framework

.NET Framework poskytuje komplexný programovací model pre vytváranie všetkých typov aplikácií. Od mobilných cez webové, až po desktopové. Aplikácie pre platformu .net nie sú prekladané priamo do strojového kódu, miesto toho sú prekompilované do jazyka Microsoft Intermediate Language (MSIL). Keď dôjde k prvému spusteniu aplikácie .net, CLR aktivuje just-in-time prekladač, ktorý na požiadanie kompiluje MSIL kód do natívneho kódu, ktorý je potom vykonávaný. Prostredie CLR je tiež zodpovedné za realizáciu exekučných služieb nízkej úrovne, medzi ktoré patrí automatická správa pamäte, manipulácia s výnimkami, bezpečnostné služby a kontrola typovej bezpečnosti počas behu programov. Keďže úloha CLR je v rámci riadenia aplikácií veľmi dôležitá, cieľové aplikácie, ktoré bežia na platforme .Net Framework sú označované ako riadené aplikácie.

Pravdepodobne najcharakteristickejšou črtou .NET je podpora použitia viacerých programovacích jazykov, Microsoft ponúka štyri programovacie jazyky pre platformu .NET framework. Jedná sa o Visual Basic .NET, Visual C# .NET, riadené rozšírenia pre C++ (Managed Extensions for C++) a Visual J# .NET. Existujú aj ďalšie .NET kompatibilné jazyky, ako napríklad Perl, Python a COBOL.

Aby mohli programovacie jazyky .NET spoločne pracovať vo vývojovom prostredí .NET Framework, musia vyhovovať určitým štandardom. Tieto štandardy boli



Obr. 1.7: *Znázornenie architektúry .NET Framework*

vytvorené spoločnosťou Microsoft a sú sústredené v tzv. spoločnej jazykovej špecifikácii (Common Language Specification – CLS). CLS určuje akým kritériám musí jazyk vyhovovať, aby mohol byť použitý na platforme .NET Framework spoločným behovým prostredím, a aby mohol spolupracovať so softvérovými komponentami, ktoré boli vytvorené v iných programovacích jazykoch. Ak jazyk implementuje tieto štandardy, je označený ako .NET kompatibilný. Každý kompatibilný jazyk pracuje s rovnakými dátovými typmi, používa rovnaké triedy platformy .NET, jeho aplikácie sú skompilované do rovnakého MSIL kódu a sú riadené spoločným behovým prostredím (CLS). Preto je možné považovať všetky kompatibilné jazyky za rovnocenné, aspoň čo sa týka programovania pre platformu .NET. Tým pádom je možné, že komponenty napísane v jednom programovacom jazyku budú používané komponentami napisanými v inom programovacom jazyku. Napríklad trieda napísaná v C# môže dediť od triedy napísanej vo Visual Basicu.5, 6, 7

1.9 Emgu CV

Emgu CV je multiplatformný .NET wrapper knižníc **Open CV**. Jednoducho povedané, Emgu CV umožnuje využívanie knižníc Open CV pre .NET kompatibilné jazyky (C#,Visual Basic,IronPhyton atď.). Solution Emgu CV môže byť kompilované pomocou Visual Studia, Xamarin Studia alebo pomocou Unity.

Name	Emgu CV (Open Source)	Emgu CV (Commercial Optimized)	Emgu CV for iOS (Commercial)	Emgu CV for Android (Beta)
OS	Windows, Linux, Mac OSX	Windows	iOS (iPhone, IPad, IPod Touch)	Android
Supported CPU Architecture	i386, x64	i386, x64	armeabi, armeabi-v7, i386 (Simulator)	armeabi, armeabi-v7a, x86
GPU Processing	✓	✓	X	X
Machine Learning	✓	✓	✓	✓
Tesseract OCR	✓	✓	✓	✓
Intel TBB (multi-thread)	X	✓	X	X
Intel IPP (high performance)	X	✓	X	X
Intel C++ Compiler (fast code)	X	✓	X	X
Exception Handling	✓	✓	✓	✓
Debugger Visualizer	✓	✓	X	X
Emgu.CV.UI	✓	✓	X	X
License	GPL	Commercial License	Commercial License	Commercial License

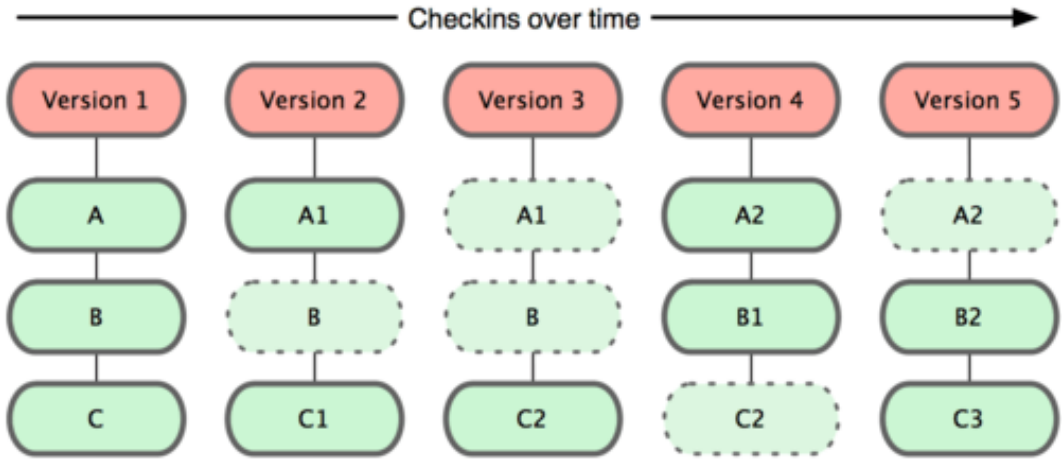
Obr. 1.8: *Prehľad verzií EmguCV. (Prevzaté od 17)*

1.10 Systém Git

Git je distribuované úložisko. Využíva tzv. repozitáre na ukladanie dát z projektu. Veľkou výhodou Gitu je možnosť verzovania projektu. Programátor si môže v Gite vytvoriť tzv. Branch (verziu projektu), vykonávať v nej zmeny, a v prípade potreby sa jednoducho vrátiť k predošlej nezmenenej verzií projektu. Hlavným rozdielom medzi systémom Git a všetkými ostatnými systémami VCS (vrátane Subversion a jemu podobných), je spôsob, akým Git spracováva dáta. Väčšina ostatných systémov ukladá informácie ako zoznamy zmien jednotlivých súborov. Tieto systémy (CVS, PERFORCE, Bazaar atď.) chápu uložené informácie ako sadu súborov a zoznamov zmien týchto súborov v čase.

Git spracováva dáta inak. Chápe ich skôr ako sadu snímok (snapshots) vlastného malého systému súborov. Zakaždým, keď v systéme zapíšeme (uložíme) stav projektu, Git v podstate „vyfotí“, ako vyzerajú všetky vaše súbory v danom okamihu, a uloží referencie na túto snímku. Ak v súboroch neboli vykonané žiadne zmeny, Git v záujme zefektívnenia práce neukladá znova celý súbor, ale iba odkaz na predchádzajúce identický súbor, ktorý už bol uložený.

V súčastnosti je možné repozitáre Gitu ukladať na špeciálne internetové úložiská pre vývojárov. Táto možnosť prináša veľké výhody nielen z pohľadu zálohovania projektu, ale aj sprístupnenia jedného zdrojového kódu viacerým vývojárom, ktorí v ňom môžu robiť zmeny. Tieto úložiská sú poskytované zadarmo s určitými ob-



Obr. 1.9: *Git ukladá dáta ako snímky projektu premenlivé v čase*

medzeniami (napr. počet vývojárov s prístupom k jednému projektu, počet verzií projektu a pod.).Najznámejším takýmto úložiskom je GitHub.

1.11 Asynchrónne programovanie pomocou modifikátoru Async a operátoru Await

Pomocou asynchrónneho programovania je možné zrýchliť beh programu, resp. zabrániť mylnému dojmu, že program neodpovedá, pretože je vykonávaná zdĺhavá operácia alebo výpočty. Vykonávanie časovo náročnejšie procesu (veľký počet výpočtov, čakanie na odpoveď zo serveru a pod.), je možné v asynchrónnych metódach, ktoré „bežia“ v inom ako hlavnom vlákne. Takéto riešenie umožňuje rýchlejšie reakcie na aktivitu užívateľa.

V jazyku C# na tvorbu asynchrónnej metódy slúži kľúčové slovo **async**. Takáto metóda byť implementovaná s návratovou hodnotu typu **Task<T>**. Následné pri volaní asynchrónnej metódy v kóde sa používa kľúčové slovo **await**. Riadky kódu, ktoré sa nachádzaju až za takýmto volaním sú vykonané až keď sú operácie v metóde ukončené resp. je vrátená nejaká výsledná hodnota.

- Modifikátor **async:** určuje, že metóda (alebo lambda výraz) je asynchrónny. Nevytvára automaticky asynchrónnosť metódy, ale označuje, že daná metóda obsahuje volanie minimálne jednej – inej asynchrónnej metódy, pomocou operátoru await. Takáto metóda je spracovávaná v hlavnom vlákne tak dlho, ako je to len možné.
- Operátor **await:** vracia kontrolu, kým nie je úloha označená operátorom Await (čakajúca úloha) dokončená. Neblokuje však hlavné vlákno (synchrónne ope-

rácie). Kód, ktorý sa vo vnútri metódy nachádza až za operátorom await, je vykonaný po ukončení (vrátenie hodnoty, timeout a pod.) tejto úlohy. Operátor await určuje, že od tohto miesta môže byť metóda spracovávaná v inom ako hlavnom vlákne. Prakticky nahrádza písanie pokračujúcich úloh, čo umožňuje TPL. Pokračovanie teda vytvára kompilátor, nie programátor.

Je nutné podotknúť, ze kľúčové slova async a await sú súčasťou jazyka C# od verzie 5.0.

1.12 Nástroje pre splnenie zadania Bakalárskej práce

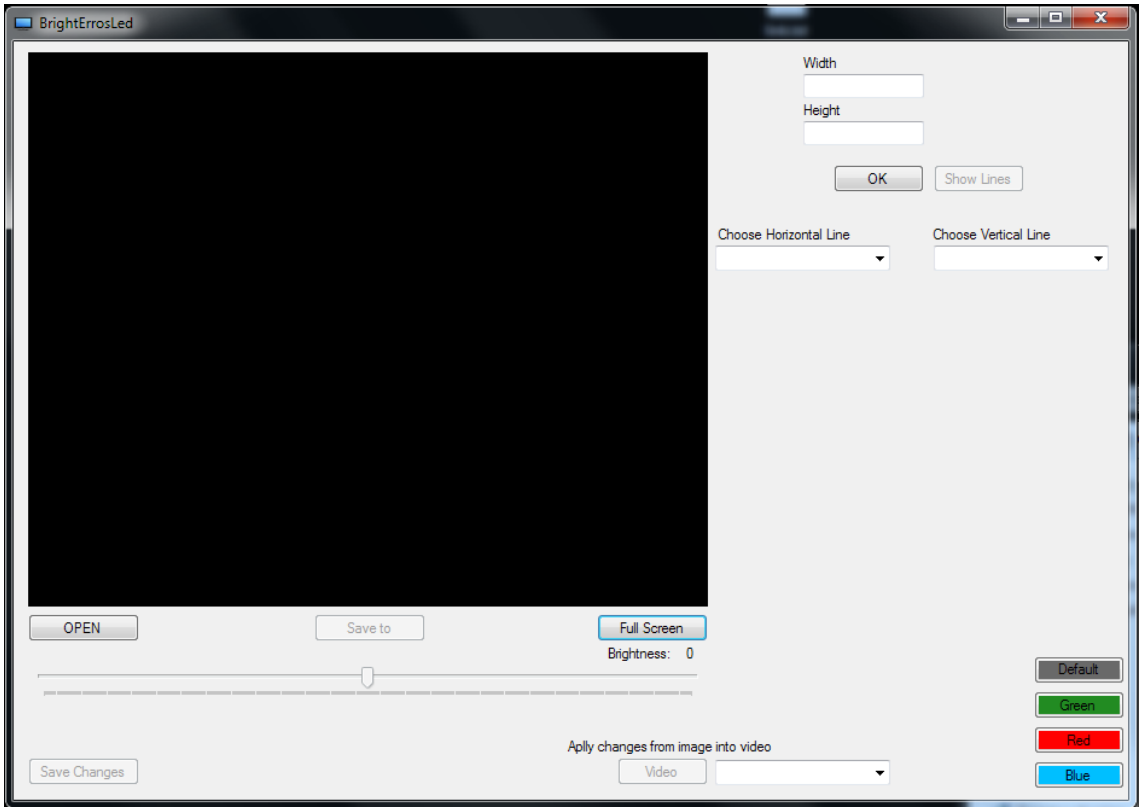
Na základe predošlých skúseností som si pre riešenie zadania bakalárskej práce, t.j. vytvorenie softvérovej aplikácie (ako som už spomenul aj v teoretickej časti práce) vybral programovací jazyk **C#** a využíval som možnosti platformy **.NET framework v4.5**.Vytvoril som aplikáciu pre OS Windows (Windows Form Application) vo vývojovom prostredí **MS Visual Studio 2012** a **MS Visual Studio 2015**.

2 PRAKTICKÉ RIEŠENIE

V tejto kapitole bakalárskej práce popisujem fungovanie a obsluhu aplikácie, ktorú som navrhol a vytvoril podľa zadania.

2.1 Vlastná aplikácia

Výsledkom mojej práce je aplikácia, pomocou ktorej je možné zobrazovať statický obrázok, a v prípade potreby kalibrovať jeho jas na miestach, kde hrozí vznik jasových chýb na LED obrazovke. Tieto chyby sa nachádzajú spravidla na mieste, kde sa stretávajú kabinety, z ktorých LED obrazovka pozostáva, alebo tiež na miestach stretu klástrov vo vnútri kabinetov. Program tieto miesta vyznačí na obrazovku po zadaní rozlíšenia kabinetu obrazovky v jednotkách pixel. Po úprave jasu v statickom obrázku, umožňuje program túto zmenu premietnuť aj do videa. Program renderuje nové video, v ktorom je aplikovaná zmena jasu v každom obrázku (*angl. frame*), z ktorého sa video skladá. Toto nové video je uložené na miesto na disku, ktoré užívateľ aplikácie vyberie.



Obr. 2.1: *Ukážka programu po spustení*

Celá práca programu by sa dalo rozdeliť do týchto krokov:

1. Načítanie obrázku z disku PC
2. Prispôsobenie rozlíšenia načítaného obrázku podľa rozlíšenia pripojenej obrazovky
3. Načítanie hodnôt rozlíšenia kabinetov (do programu ich musí zadať užívateľ)
4. Vykreslenie hraníc kabinetov na obrazovku pomocou čiar
5. Uloženie súradníc chybových miest do objektu „MyLine“
6. Výber vykreslenej čiary a úprava jasu v obrázku užívateľom
7. Uloženie vykonanej úpravy
8. Výber videa, do ktorého sa má zmena premietnuť
9. Vyhľadanie zmien v obrázku
10. Výber miesta na disku, kam sa má upravené video uložiť
11. Vytvorenie nového upraveného videa a jeho uloženie na disk

2.2 Popis implementácie kódu

```
private async Task<VideoWriter> RemakeVideoAsync(Capture Video, PixelChanges?[] Changes, Image<Hls, Byte> OrgHls, Image<Hls, Byte> ChHls, VideoWriter VW, IProgress<int> progress)
{
    BrightErrosLed.Cancel = false;
    await Task.Run(async () =>
    {
        for (int i = 1; i < (Video.GetCaptureProperty(Emgu.CV.CvEnum.CapProp.FrameCount) - 1); i++)
        {
            Image<Bgr, Byte> NewFrame = (Video.QueryFrame().ToImage<Bgr, Byte>());
            Image<Hls, Byte> FrameHls = NewFrame.Convert<Hls, Byte>();
            FrameHls = FrameHls.Resize(resolution.Width, resolution.Height, Emgu.CV.CvEnum.Inter.Nearest);

            if (Changes != null)
                FrameHls = await FindBitmapChanges(OrgHls, ChHls, FrameHls, Changes);
            NewFrame = FrameHls.Convert<Bgr, Byte>();
            VW.Write(NewFrame.Mat);
            GC.Collect();
            GC.WaitForPendingFinalizers();
            progress.Report(Convert.ToInt32((i / (Video.GetCaptureProperty(Emgu.CV.CvEnum.CapProp.FrameCount) - 1)) * 100));
            if (BrightErrosLed.Cancel)
            {
                break;
            }
        }
        return VW;
    });
    return VW;
}
```

Obr. 2.2: *Ukážka metódy, ktorá vytvára upravené video. Metóda sa nachádza v triede Controller*

Aplikácia pre manuálnu korekciu jasových chýb pozostáva z jedného projektu typu Windows Form Application, ktorého implementácia je rozdelená do niekoľkých tried:

1. BrightErrorsLed.cs
2. Controller.cs
3. Myline.cs
4. PixelChanges.cs

Trieda **BrightErrorsLed** obsahuje všetky metódy, ktoré priamo reagujú na činnosť užívateľa programu. Tieto metódy sú naviazané na jednotlivé komponenty (napr. tlačítko, trackbar a pod.).

Trieda **Controller** obsahuje metódy, ktoré vykonávajú výpočtovo náročnejšie akcie. Väčšina metód v tejto triede je verejná a sú asynchronné. Objekt Controller má prázdny konstruktor a slúži len ako sprostredkovateľ týchto metód. Takáto implementácia nie je nutná, ale kód je vďaka nej prehľadnejší.

Trieda **Myline** obsahuje implementáciu štruktúry Myline. Myline je objekt, ktorý zapúzdruje pole bodov (Point). Obsahuje metódu, ktorá vracia celé pole bo-

dov, na ktorých leží čiara, a metódu, ktorá vráti určitý bod (Point) podľa predaného parametru *x*.

Trieda **PixelChanges** obsahuje iba jednoduchú štruktúru, ktorá má vlastnosti *int X* a *int Y*, a konstruktor, ktorý tieto vlastnosti pri vytvorení nového objektu inicializuje. V kóde tento objekt využívaný ako pole objektov, ktorých súradnice X a Y predstavujú súradnice pixelov obrazu, v ktorých bola vykonaná zmena jasu.

Ako už bolo spomenuté vyššie, vytvorená aplikácia je typu Windows Forms a bola implementovaná v jazyku Visual C#. Základom celého programu je framework EmguCV, ktorý sprostredkuje možnosti knižníc OpenCV pre .NET a komponenta PictureBox, prostredníctvom ktorej sa zobrazuje obrázok a vykresľujú sa hranice kabinetov obrazovky.

Počas práce na tomto projekte bol použitý Git a internetové úložisko pre vývojárov od MSDN. Okrem možnosti prístupu k projektu z rôznych PC, Git umožňoval aj prehľad o histórií vykonaných zmien.

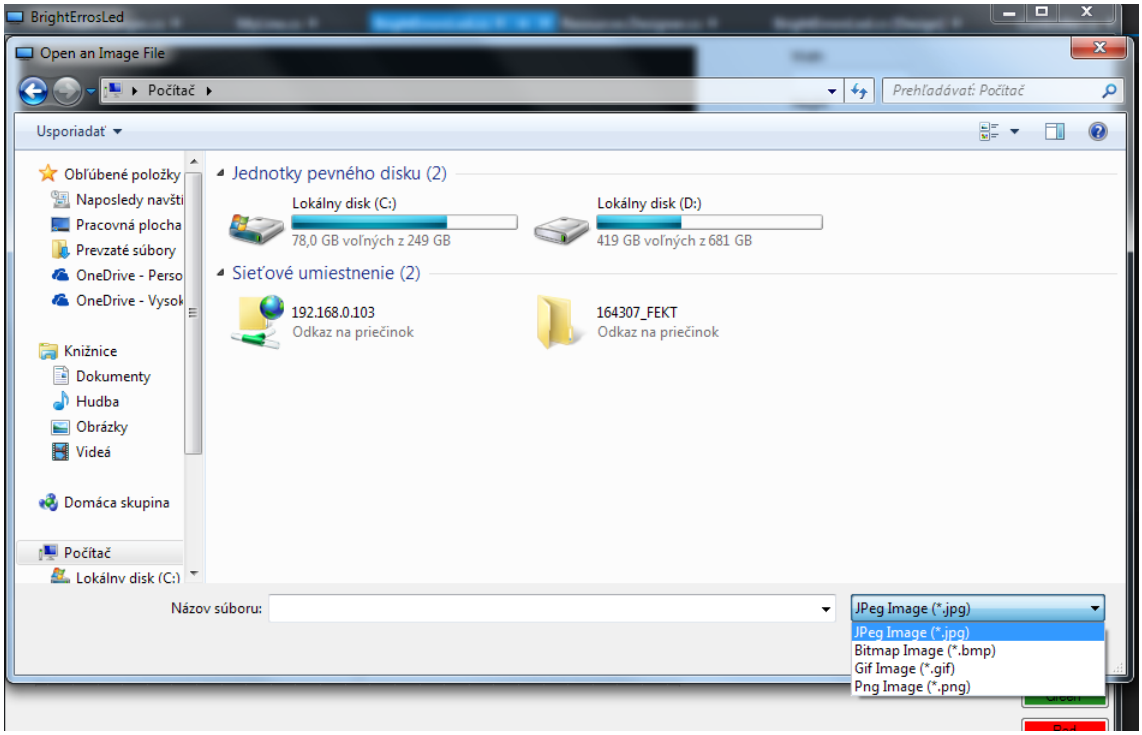
2.3 Popis obsluhy a fungovania aplikácie

Táto kapitola si kladie za cieľ popísať spôsob práce s aplikáciou pomocou grafického užívateľského rozhrania a možnosti, ktoré aplikácia ponúka.

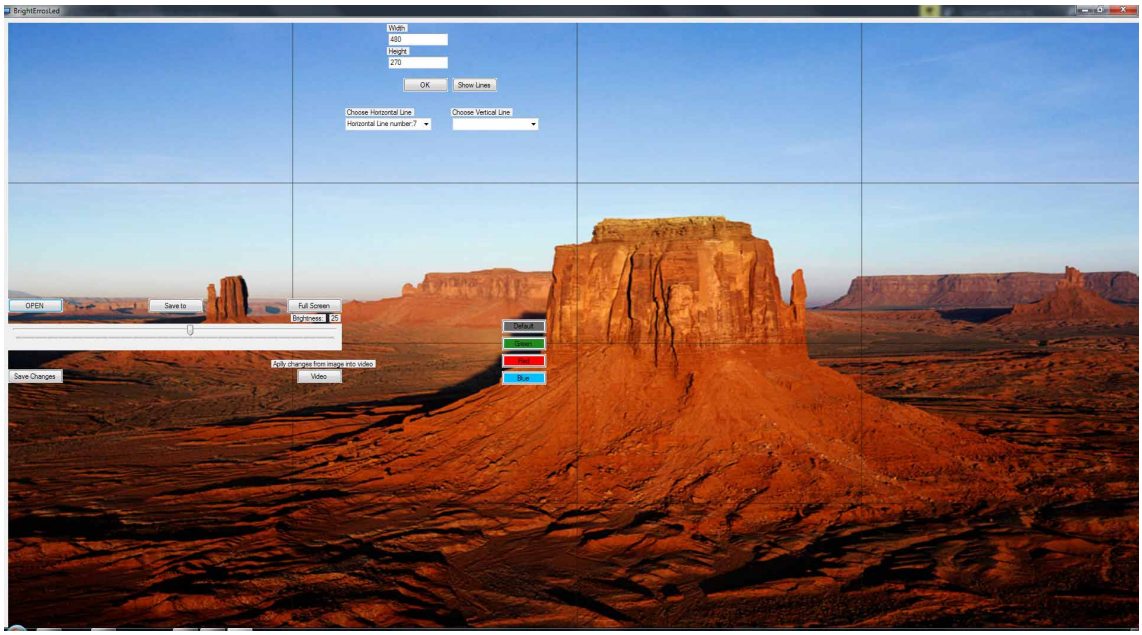
2.3.1 Úprava statického obrazu

Po spustení aplikácie je potrebné najprv vybrať obrázok, v ktorom bude kalibrovaný jas. Na tento úkon slúži tlačítko „*OPEN*“. Po kliknutí na toto tlačítko sa otvorí okno, pomocou ktorého je možné vložiť do programu ľubovolný obrázok, ktorý spĺňa jeden z formátov *.jpg, .png, .bmp a .gif*. Následne po potvrdení sa toto okno zatvorí, program vybraný obrázok spracuje, zmení jeho rozlíšenie na rozlíšenie obrazovky (resp. rozlíšenie, ktoré je momentálne nastavené v OS), a obrázok sa zobrazí v aplikácií. Ak niečo nieje s obrázkom v poriadku, aplikácia vypíše chybovú hlášku v okne „MessageBox“.

Ďalším krokom je zadanie rozmerov, resp. rozlíšenia kabinetov HD LED obrazovky. Tento údaj je potrebné zadať v jednotkách pixel do polí označených nápisom „Width“ a „Height“ (šírka a výška). Zadané rozlíšenie užívateľ potvrdí tlačítkom „OK“. V programe prebehne validácia zadaných rozmerov, zadaný string sa musí dať skonvertovať na celočíselnú hodnotu, touto hodnotou musí byť rozmer obrazovky deliteľný bez zvyšku a pod. Ak užívateľ zadá rozmery, ktorými by sa obrazovka nedala rozdeliť na rovnako veľké kabinety, alebo z iného dôvodu neprejdú validáciou, program vypíše chybovú hlášku v okne „MessageBox“.

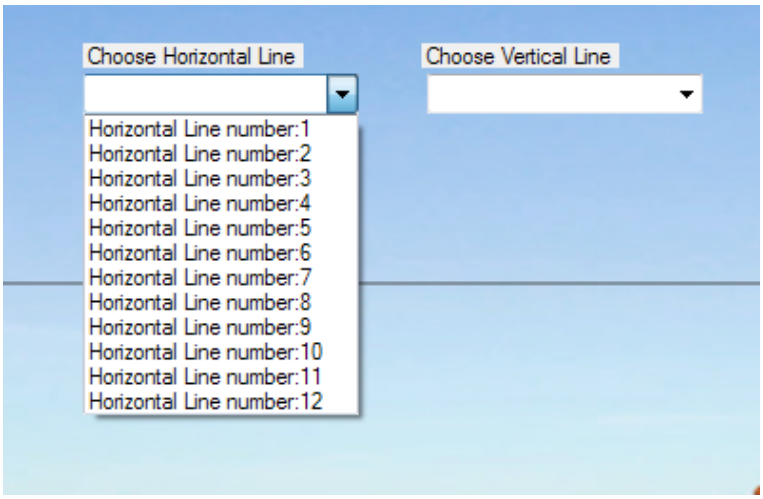


Obr. 2.3: *Okno pre výber obrázku.*



Obr. 2.4: *Okno aplikácie po úspešnom vykreslení hraníc kabinetov obrazovky.*

Po vykreslení hraníc kabinetov sú súradnice zobrazených čiar uložené v programe. Pre úpravu jasu na mieste, kde sa nachádza jedna z čiar, musí užívateľ požadovanú čiaru vybrať pomocou komponenty „DropDownList“. Tieto komponenty, obsahujúce zoznamom čiar, sa v programe nachádzajú dve. V jednej je zoznam vertikálnych a v ďalšej horizontálnych čiar. Čiary su v týchto zoznamoch indexované od 1 až po x (x je počet vertikálnych alebo horizontálnych čiar), začínajúc v ľavom hornom rohu okna.



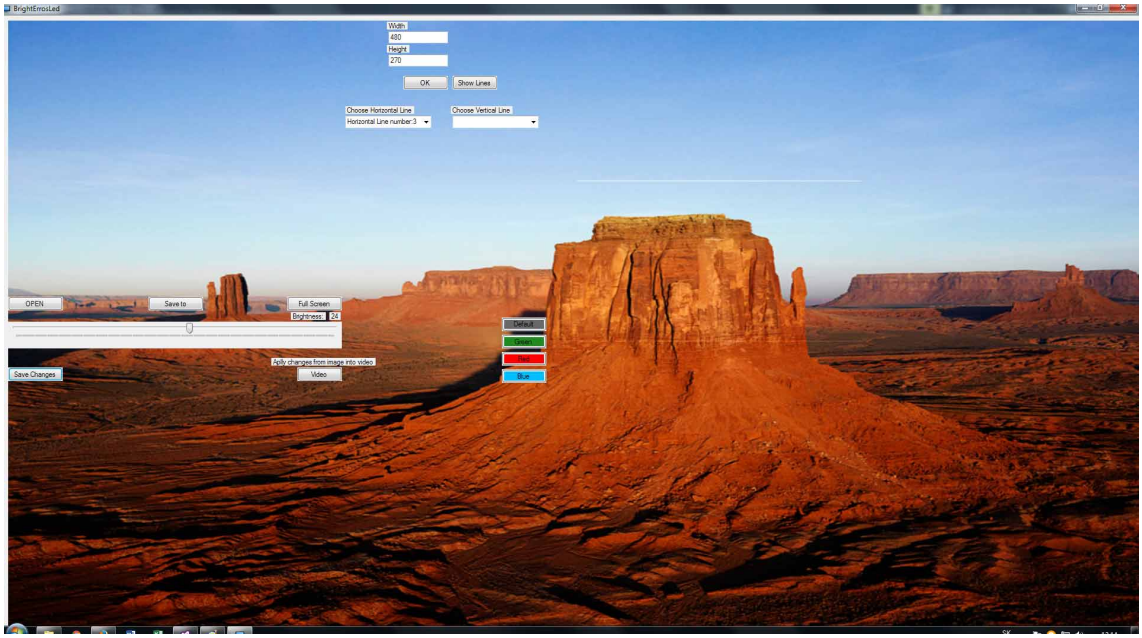
Obr. 2.5: *Zoznam čiar.*

Ďalej môže užívateľ pristúpiť k samotnej úprave jasu. Najprv si musí vybrať čiaru, na ktorej mieste chce úpravu vykonať, a následne pomocou komponenty „TrackBar“ pridať alebo naopak odobrať množstvo jasu na tomto mieste v obrázku. Pri každej zmene hodnoty na tejto komponente, je v programe volaná metóda, do ktorej sa v parametroch predáva táto nová hodnota, a súradnice pixelov v ktorých ma byť zmenený jas. Samotný prepočet jasu prebieha tak, že pixely sú prepočítané do modelu HSL a hodnota parametru Lightness je znížená alebo naopak zvýšená (viď. 2.6). Ak je užívateľ s vykonanou zmenou spokojný a chce ju v obrázku ponechať, je potrebné kliknúť na „Save changes“. Kliknutím na toto tlačítko dá uživateľ programu pokyn uložiť vykonané zmeny jasu. Program po úspešnom uložení vypíše hlášku „Changes have been saved“ ,teda zmeny boli uložené. Týmto krokom sa však neprepíše pôvodný súbor na disku, ale iba objekt v operačnej pamäti.

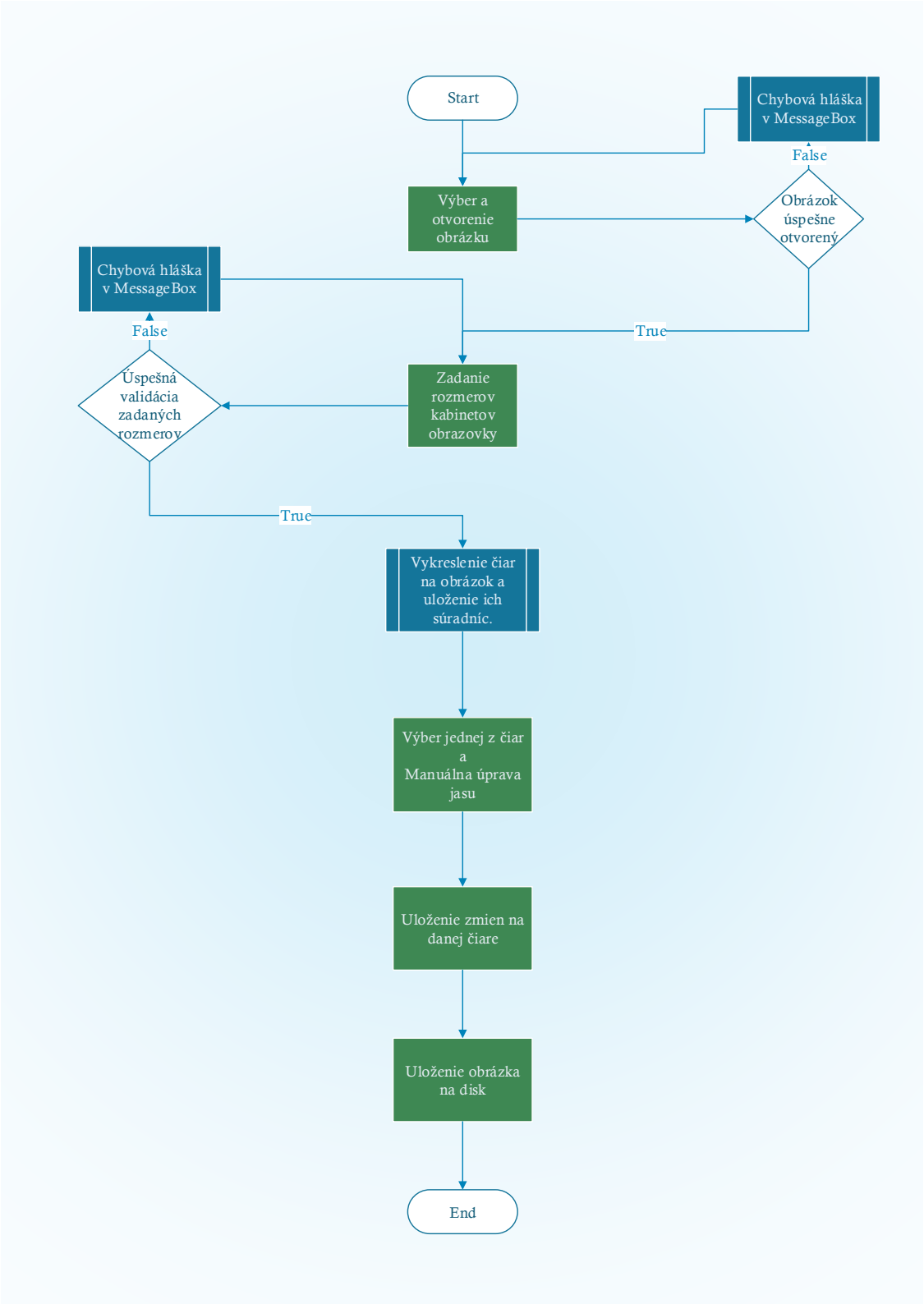
Ak chce užívateľ uložiť nový obrázok na miesto na disku, musí použiť tlačítko „Save to“. V novom okne vyberie adresár, do ktorého chce obrázok uložiť, formát uloženého obrázku a jeho názov.

```
public async Task<Bitmap> SetLineBrightness(int lightness, Bitmap bmap, Point[] Body)
{
    Image<Hls, Byte> newBmap = new Image<Hls, byte>(bmap);
    await Task.Run(() =>
    {
        for (int i = 0; i < Body.Length; i++)
        {
            var c = newBmap[Body[i].Y, Body[i].X];
            c.Lightness += lightness;
            newBmap[Body[i].Y, Body[i].X] = c;
        }
    });
    return newBmap.Bitmap;
```

Obr. 2.6: *Metóda vykonávajúca zmenu parametru Lightness v pixeloch obrázku*



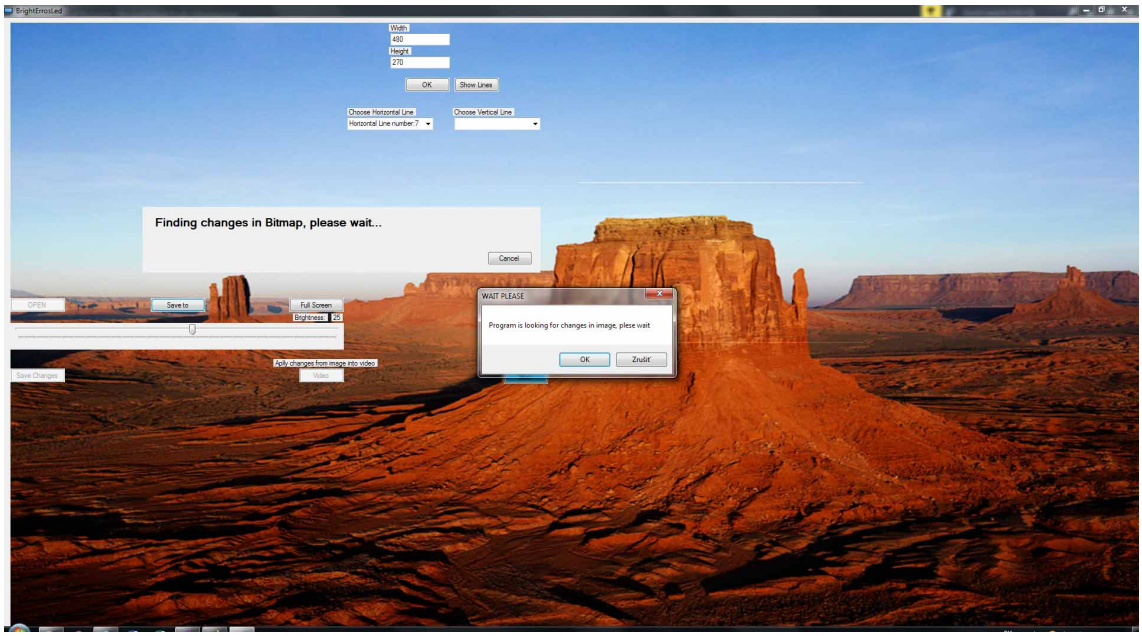
Obr. 2.7: *Úprava jasu na čiarách 3 a 7 pri použití kabinetov s rozlíšením 480 x 270px na obrazovke s rozlíšením 1920 x 1080px.*



Obr. 2.8: *Vývojový diagram práce s programom pri úprave statického obrazu (zelené bloky –> kroky užívateľa)*

2.3.2 Úprava videa

Program umožňuje aplikovať zmenu jasu vykonanú v statickom obraze do videa. K tejto možnosti sa užívateľ dostane po stlačení tlačítka „Video“. Program najprv porovná upravený obrázok s pôvodným a súradnice každého pixelu v ktorom je nájdený rozdiel si uloží do objektu PixelChanges. Následne aplikácia otvorí okno pre výber videa na spracovanie z ľubovolného adresáru. Hneď za týmto krokom, sa zobrazí MessageBox so žiadosťou, aby užívateľ vybral adresár do ktorého sa má uložiť nové video. Po kliknutí na tlačítko OK sa otvorí okno pre výber miesta, kde sa má uložiť nové video.

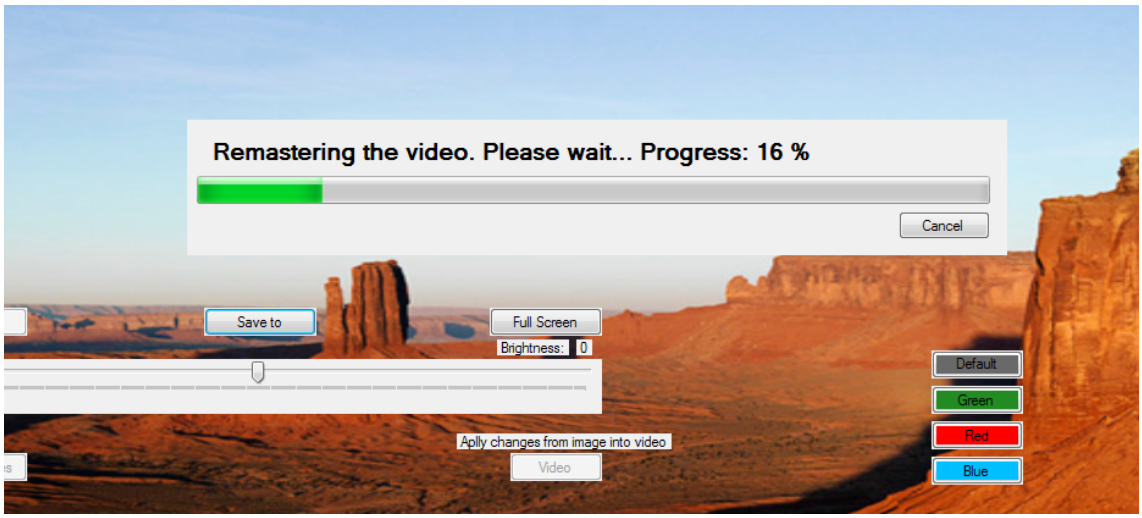


Obr. 2.9: *Vyhladávanie úprav v obrázku po kliknutí na tlačítko „Video“.*

Proces vytvárania nového videa, v ktorom je aplikovaná zmena jasu, je časovo veľmi náročný, keďže rozlíšenie každého framu videa je najprv zmenené na rozlíšenie obrazovky. V takto upravenom obrázku je vykonaná aj zmena jasu a to tak, že sa mení hodnota Lightness (model HSL) každého pixelu obrázku, ktorého súradnice sú uložené v objekte PixelChanges. Z dôvodu zdĺhavosti celého procesu, program oboznamuje užívateľa o progrese pomocou komponenty „ProgressBar“ a tiež textom ktorý zobrazuje percentuálne množstvo úspešne spracovaných dát (viď. 2.10).

2.3.3 Iné funkcie programu

Okrem úpravy statického obrazu alebo videa, je aplikácia schopná zobrazovať vybraný obrázok, či už s úpravami alebo bez nich. Na zobrazenie obrázku na celú obrazovku a skrytie všetkých ovládacích komponentov slúži tlačítko „Full Screen“



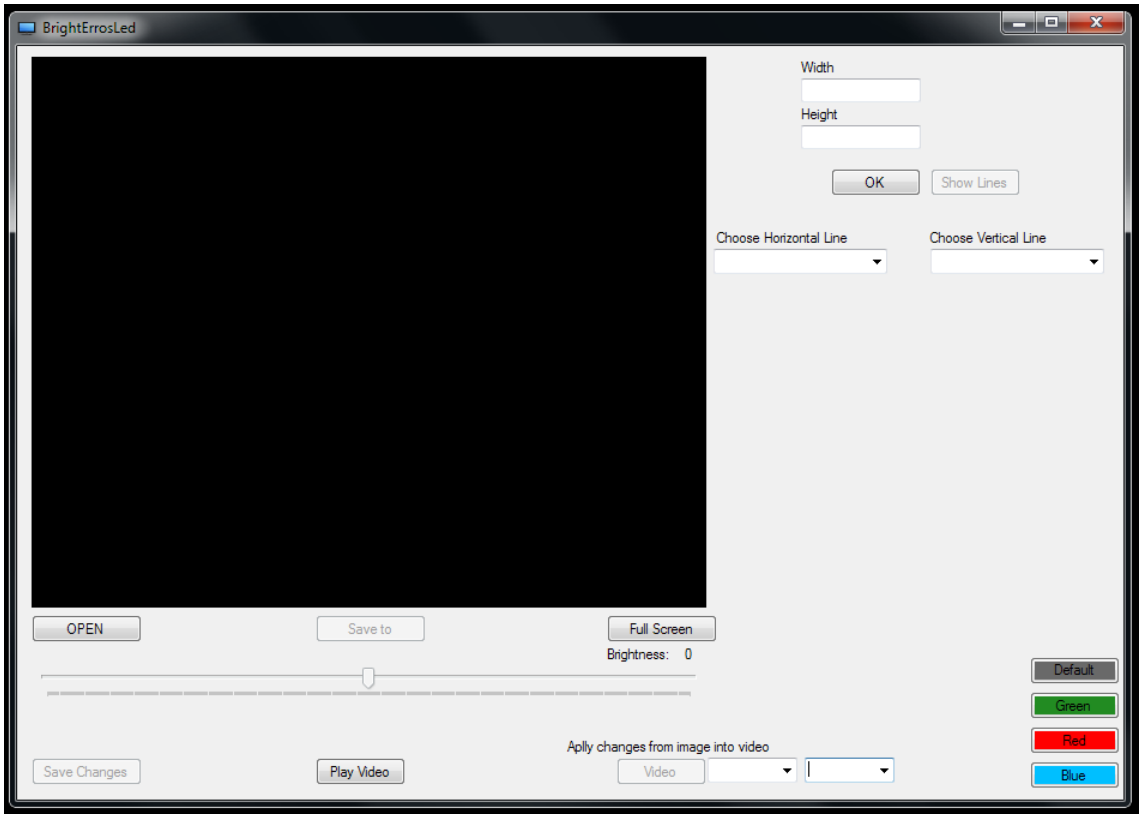
Obr. 2.10: *Zobrazovaná informácia o progrese, počas vytvárania upraveného videa*

alebo dvoj-kliknutie myšou. Späť do predošlého režimu sa užívateľ dostane rovnako dvoj-klikom na ľubovoľné miesto na obrázku.

Ďalej grafické užívateľské rozhranie aplikácie ponúka tri tlačítka na zmenu farby čiar, ktoré vyznačujú hranice kabinetov obrazovky, a jedno tlačítko na návrat k pôvodnej farbe čiar. Tieto tlačítka sú umiestnené v pravom dolnom rohu rozhrania.

Keďže po začatí úprav jasu pomocou trackbaru vykreslené hranice kabinetov zmiznú, aby užívateľovi nezakrývali miesto, kde upravuje jas, je potrebné aby aplikácia ponúkala možnosť tieto hranice opäť vykresliť. Na tento úkon slúži tlačitko „Show Lines“.

2.4 Verzia aplikácie `1.1.0.2` beta



Obr. 2.11: *Úvodné okno verzie `1.1.0.2` beta*

Súčasťou prílohy k Bakalárskej práci je ešte jedna verzia aplikácie, ktorú som popisoval v predošlých kapitolách. Verzia `1.1.0.2` beta naviac ponúka možnosť kalibrácie jasu počas prehrávania videa. I keď verzia `1.0.0.0` umožňuje kalibrovať aj dynamický obraz, v praxi by bolo priaznivejšie aby aplikácia umožňovala kalibrovať a prehrávať video v reálnom čase.

Túto úlohu novšia verzia aplikácie zvláda, avšak iba z časti. Video je možné prehrávať a počas prehrávania upravovať, no celý proces úpravy každého obrázku videa je o niekoľko milisekúnd pomalší, ako by bolo potrebné. Toto vo výsledku spôsobuje jemné „sekanie“, resp. video je prehrávané spomalene. Na obrázku 2.12 je vidieť čas v milisekundách, potrebný výber obrázku z videa, zmenu jeho veľkosti a pripočítanie masky, ktorá reprezentuje úpravu jasu v obrázku. Jednalo sa o video s 30 FPS teda čas na zmenu obrázku bol 33,33 milisekundy. Na obrázku 2.15 je vidieť celý priebeh metódy, ktorá sa volá po stlačení tlačítka „Play Video“

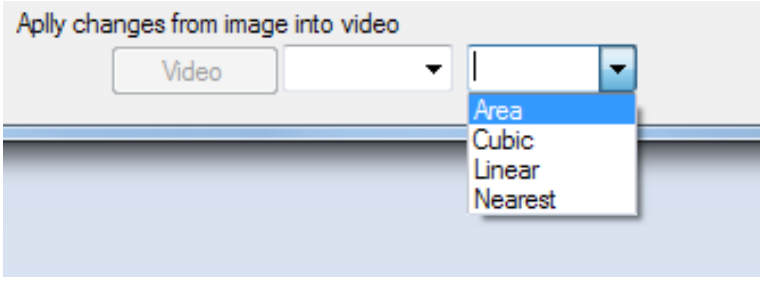
Ďalším problémom, ktorý sa pre túto verziu nepodarilo vyriešiť je, že úprava jasu, ktorá sa premietne do videa, môže byť buď kladná alebo záporná. Čiže v celom obraze je možne jas buď iba pridať alebo iba ubrať. Nieje možné na jednej čiare

jas zvýšiť a na inej znížiť. Respektíve, nieje možné to takto premietnuť do videa. Dôvodom je iný spôsob implementácie a hľadania zmeny v obrázku, než v prvej verzií.Vo verzií 1.0.0.0, program mení hodnotu jasu v každom pixeli obrázku, ktorý bol upravený. Vo verzií 1.1.0.2 každá zmena jasu iba vytvára nový obrázok, ktorý je absolútnym rozdielom medzi pôvodným a upraveným obrázkom. Pri prehrávaní videa sa potom nemusí každý obrázok komplikovane prepisovať ale je k nemu iba pripočítaná takto vytvorená maska. Vďaka takémuto riešeniu je vôbec možné video prehrávať, pretože je oveľa rýchlejšie ako riešenie pre prvú verziu.

```
if (Stop)
    break;
var frame = await GetNextRemastredFrameAsync(video);
stopWatch.Stop();
if (stopWatch.ElapsedMilliseconds >= imageTime)   ≤ 53ms elapsed
    MainScreen.Image = frame;   stopWatch.ElapsedMilliseconds 37
else
{
    Thread.Sleep((int)(imageTime - stopWatch.ElapsedMilliseconds));
    MainScreen.Image = frame;
}
```

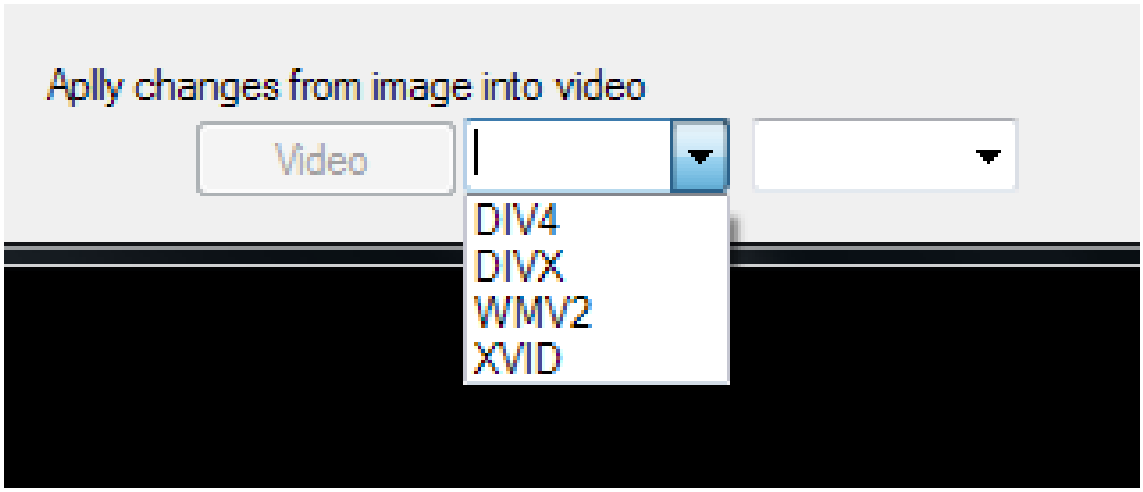
Obr. 2.12: *Čas potrebný na načítanie, zmenu rozlíšenia, súčet s maskou rozdielov a zobrazenie každého obrázku z videa. Tento čas kolíše na hodnotách od 33 až po 45ms*

V tejto verzií je pridaná aj možnosť výberu kodeku, ktorý sa použije pri vytváraní nového videa, a možnosť výberu metódy, ktorá sa má použiť, pri zmene veľkosti/rozlíšenia obrázku.

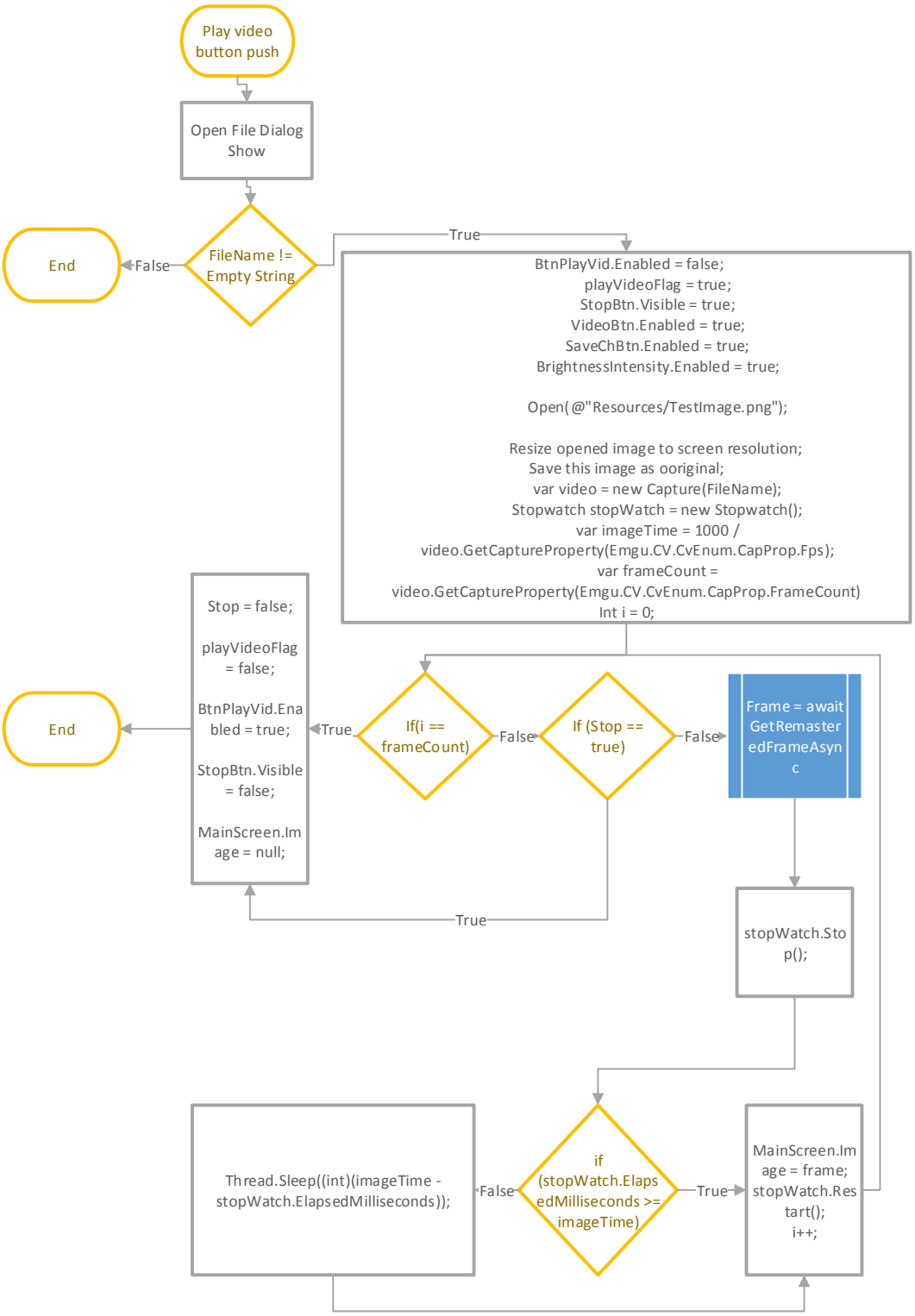


Obr. 2.13: *Ponuka metód zmeny rozlíšenia obrázku*

Avšak celkovo nie je verzia 1.1.0.2 aplikácie veľmi stabilná, a to z jedného dôvodu. Video je prehrávané v komponente PictureBox (resp. PictureBox je použitý na zobrazovanie snímkov videa), ktorá na to nie je určená a niekedy spôsobí chybu. Táto chyba sa prejaví ako dve červené čiary na bielom pozadí, idúce po uhlopriečkach komponenty.



Obr. 2.14: *Ponuka kodekov*



Obr. 2.15: *Vývojový diagram metódy po stlačení tlačítka Play Video vo verzií 1.1.0.2*

3 ZÁVER

Cieľom tejto Bakalárskej práce bolo navrhnúť a vytvoriť softvérovú aplikáciu, pomocou ktorej je možné zobrazovať statický obraz na HD LED displeji, a zároveň upravovať hodnoty veľkosti jasu obrázku na miestach, kde je to potrebné k odstráneniu jasových chýb, ktoré sú spôsobené teplotnou rozťažnosťou blokov, z ktorých sa obrazovka skladá.

Práca pozostáva z teoretickej a praktickej časti. Obsah teoretickej časti oboznamuje čitateľa s nástrojmi a princípmi, ktoré boli využité počas vypracovávania zadania. Praktická časť popisuje spôsob implementácie, funkciu a možnosti vytvorenej aplikácie.

Výsledný softvér, spustiteľný na OS Windows, je súčasťou priloženého CD, na ktorom sa nachádza v zložke „/Bakalárskej práca – aplikácia/Release“ pod názvom „BP_v1.0.0.0.exe“ a „BP_v1.1.0.2_beta.exe“. Aplikáciu som vytvoril a testoval vo vývojom prostredí MS Visual Studio 2012 a MS Visual Studio 2015, na PC s OS Windows 7 Professional a nasledujúcimi HW parametrami:

- **CPU**: AMD FX-8320E Black Edition, 3,2 GHz
- **RAM**: Patriot DIMM 8GB DDR3, 1600MHz
- **GPU**: MSI nVidia Geforce GTX N750Ti Gaming 2GD5/OC, 2GB

Pri testovaní bola použitá obrazovka Philips s rozlíšením 1920x1080px. Verzia 1.0.0.0 je stabilná a sú v nej ošetrené všetky neočakávané udalosti, na ktoré som počas vývoja aplikácie narazil. Verzia 1.1.0.2 ponúka naviac úpravu dynamického obrazu, avšak má nedostatky, ktoré sú popísané v praktickej časti tejto práce.

LITERATÚRA

[1] DVOŘÁČEK V. *Světelné zdroje - svételné diody* [online].:4[cit. 2015-12-04]. Dostupné z URL: <http://www.odbornecasopisy.cz/res/pdf/39810.pdf>.

[2] PIHAL R. *Vše o světle - 3.Intenzita (jas) světla* [online]. [cit. 2015-12-04]. Dostupné z URL: <http://www.fotoroman.cz/techniques3/svetlo03jas.htm>.

[3] KOTEK J. *Velkoplošné obrazovky na bázi diod LED* [online]. [cit. 2015-12-05]. Dostupné z URL: <http://www.odbornecasopisy.cz/svetlo/casopis/tema/velkoplosne-obrazovky-na-bazi-diod-led--16562>.

[4] *RGB farebný model* [online]. [cit. 2015-12-05]. Dostupné z URL: <http://www.cookie.sk/slovnik-tvorba-webstranky/r/rgb-farebny-model.htm>.

[5] MSDN *Introduction to the C# Language and the .NET Framework* [online]. [cit. 2015-12-06]. Dostupné z URL: <https://msdn.microsoft.com/cs-cz/library/z1zx9t92(v=vs.110).aspx>.

[6] *.Net Framework* [online]. [cit. 2015-12-06]. Dostupné z URL: <http://ics.upjs.sk/~rkb/web/s.ics.upjs.sk/_mamjur/net/dotnet.html>.

[7] BĚHÁLEK *Architektura .NET Framework* [online]. [cit. 2015-12-06]. Dostupné z URL: <http://www.cs.vsb.cz/behalek/vyuka/pcsharp/text/ch01s01.html>.

[8] *SCHÉMA FUNGOVÁNÍ LED OBRAZOVKY* [online]. [cit. 2015-12-09]. Dostupné z URL: <http://www.internationaltrucker.cz/led-obrazovky/schema-fungovani-led.html>.

[9] *Krátky popis obrazovky* [online]. [cit. 2015-12-09]. Dostupné z URL: <http://www.goldoffice.cz/prodej-obrazovek-kategorie.php?kat=HD>.

[10] LEYARD *LED Display TV1.9 (P1.9)* [online]. [cit. 2015-12-09]. Dostupné z URL: <http://www.leyardeurope.eu/product/12/LED-Display-TV1-9-LED-TV-P1-9>.

[11] MSDN *Visual Studio 2012 and .NET 4.5 Complete!* [online]. [cit. 2015-12-09]. Dostupné z URL: <http://blogs.msdn.com/b/somasegar/archive/2012/08/01/visual-studio-2012-and-net-4-5-complete.aspx>.

[12] MS *Description of Visual Studio 2012 Update 3* [online]. [cit. 2015-12-09]. Dostupné z URL: <https://support.microsoft.com/en-us/kb/2835600>.

[13] DAITE *Velkoplošné LED obrazovky a LED technologie* [online]. [cit. 2015-13-09]. Dostupné z URL: <http://daite.cz/led-obrazovky-led-technologie/>.

[14] MSDN *Color* [online]. [cit. 2016-05-22]. Dostupné z URL: <https://msdn.microsoft.com/en-us/library/windows/desktop/dn742482(v=vs.85).aspx/>.

[15] Carmen Alonso Montes *Practical Computer Vision: Theory & Applications* [online]. [cit. 2016-05-22]. Dostupné z URL: <http://www.bcamath.org/documentos_public/courses/course_day1.pdf/>.

[16] Jose Vargas *HSL, HSB and HSV color: differences and conversion* [online]. [cit. 2016-05-22]. Dostupné z URL: <http://codeitdown.com/hsl-hsb-hsv-color//>.

[17] *EMGU CV Tutorial* [online]. [cit. 2016-05-22]. Dostupné z URL: <http://www.didehbonyan.com/rz/Portals/0/pdf/Emgu%20CV%20Tutorial%20Skander.pdf/>.

ZOZNAM SYMBOLOV, VELIČÍN A SKRATIEK

V ľavom stĺpci je skratka a v pravom je jej krátky slovný popis

CD Compact Disc

CLR Common Language Runtime

CLS Common Language Specification

HD high definition

LED Luminiscenčná dióda - Light-Emitting Diode

MSIL Microsoft Intermediate Language

MS Microsoft

OS Operačný systém

PC Osobný počítač (z angl. personal computer)

RGB skratka pre farby: červená,zelená,modrá - red, green, blue

FPS počet obrázkov zobrazených za sekundu.

ZOZNAM PRÍLOH

A CD S APLIKÁCIOU PRE ÚPRAVU JASOVÝCH CHÝB

Na priloženom CD sú uložené dve verzie (1.0.0.0, 1.1.0.2) softvéru ktorý, je popisovaný v praktickej časti tejto práce a Bakalárska práca v elektronickej forme.