TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

WebGL aplikace pro prohlížení živých 360° panoramat

Diplomová práce

Studijní program: N2612 – Elektrotechnika a informatika
Studijní obor: 1802T007 – Informační technologie

Autor práce: **Bc. Daniel Vích**
Vedoucí práce: Ing. Jiří Jeníček, Ph.D.

Liberec 2015

TECHNICAL UNIVERSITY OF LIBEREC
Faculty of Mechatronics, Informatics
and Interdisciplinary Studies

WebGL based application for live 360° panorama

Diploma thesis

Study programme: N2612 – Electrotechnics and informatics

Study branch: 1802T007 – Information technology

Author: **Bc. Daniel Vích**

Supervisor: Ing. Jiří Jeníček, Ph.D.

Liberec 2015

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky a mezioborových studií
Akademický rok: **2014/2015**

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Daniel Vích**
Osobní číslo:	**M12000239**
Studijní program:	**N2612 Elektrotechnika a informatika**
Studijní obor:	**Informační technologie**
Název tématu:	**WebGL aplikace pro prohlížení živých 360 panoramat**
Zadávající katedra:	**Ústav informačních technologií a elektroniky**

Zásady pro vypracování:

1. Seznamte se s problematikou streamování videa na internetu a 3D WebGL aplikací.
2. Navrhněte a realizujte WebGL prohlížeč živých panoramat.
3. Ověřte implementaci na různých OS a prohlížečích.
4. Otestujte hardwarovou náročnost.
5. Prověřte vliv kodeků na kvalitu a výpočetní nároky.

Rozsah grafických prací: **Dle potřeby dokumentace**

Rozsah pracovní zprávy: **cca 40 až 50 stran**

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

Seznam odborné literatury:

[1] **Makzan: Programujeme hry v HTML5, Computer Press, 2012, EAN:9788025137314**

[2] **Zakas, Z. N.: JavaScript pro webové vývojáře, Computer Press, 2009, EAN:9788025125090**

Vedoucí diplomové práce: **Ing. Jiří Jeníček, Ph.D.**
Ústav informačních technologií a elektroniky

Datum zadání diplomové práce: **12. září 2014**

Termín odevzdání diplomové práce: **15. května 2015**

prof. Ing. Václav Kopecký, CSc.
děkan

L.S.

prof. Ing. Zdeněk Plíva, Ph.D.
vedoucí ústavu

V Liberci dne 12. září 2014

Prohlášení

Byl jsem seznámen s tím, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé diplomové práce pro vnitřní potřebu TUL.

Užiji-li diplomovou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Diplomovou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím mé diplomové práce a konzultantem.

Současně čestně prohlašuji, že tištěná verze práce se shoduje s elektronickou verzí, vloženou do IS STAG.

Datum: 9 9 2015

Podpis:

Poděkování

Na tomto místě bych rád poděkoval vedoucímu mé diplomové práce, Ing. Jiřímu Jeníčkovi, Ph.D., za všechny rady, bohaté zkušenosti a poznatky, které mně pomohly k úspěšné realizaci této práce. Dále bych chtěl poděkovat své rodině za její podporu a možnost studovat na vysoké škole. Nakonec všem svým blízkým přátelům, kteří mně při studiu velice pomohli a motivovali.

Abstrakt

Tato práce se zabývá řešením streamování živých panoramat zaobírající úhel pohledu 360° použitím většího množství webkamer a jejich zobrazením pomocí aplikační rozhraní WEBGL ve webovém prohlížeči. Výsledkem této práce je aplikace s řešením typu server-klient, kde server zajišťuje zpracování a stream videa z webkamer a klient umožňuje zobrazení tohoto streamu na zařízeních disponujících webovým prohlížečem s podporou aplikačního rozhraní WEBGL.

Klíčová slova: panoráma, videopanoráma, stream videa, stream videa v reálném čase, WEBGL, aplikace server-klient

Abstract

This thesis deals with streaming 360 degree panoramas with multiple webcams. The final view is provided by WebGL API in a web browser. The final outcome of this thesis is to find a solution for a server-client type of application, where the server side provide processing and webcam streams and the client side can view such streams on devices that are supported by a web browser with (an application interface) WEBGL API.

Keywords: panorama, videopanorama, video sream, realtime video stream, WEBGL, server-client application

Obsah

Seznam obrázků

Seznam tabulek

Seznam zkratek

HTTP	Hypertext transfer protocol - internetový protokol pro výměnu hypertextových dokumentů
RTP	Real-time transport protocol - protokol standardizující paketové doručování zvukových a obrazových dat po internetu
RTMP	Real-time messaging protocol - protokol vyvinutý firmou Macromedia určený pro streamování audia a videa po internetu
MP4	Multimediální kontejner
HEVC	High efficiency video coding - video kodek
VP8	video kodek vyvinutý firmou Google
VP9	video kodek vyvinutý firmou Google
WMV	komprimovaný souborový videoformát vyvinutý firmou Microsoft
CMOS (fototechnika)	typ snímače fotoapárátu
FOV	Field of view - úhel viditelného pohledu
SSD	Solid-state drive - vysokokapacitní paměťové médium neobsahující pohyblivé součásti
USB	Universal serial bus - komunikační sběrnice

1 Úvod

Pokud se ohlédneme do nejmladších dob naší civilizace, tak se vždy lidé snažili uložit jisté výjevy a situace pomocí obrázků a maleb. Pravěcí lidé tyto výjevy vyrývali do skal, později malíři začali používat různé typy pláten, následně pak vynález fotoaparátu učinil obrovský pokrok v rychlém zachycení obrazu okamžiku. Ve všech těchto etapách vývoje se pořizovatelé snažili zachytit ty nejdůležitější části daného výjevu, a to buď v úzkém, nebo širokém záběru. Speciálním typem je zobrazení panoramatické, které umožní zobrazit celý prostor kolem pozorovatele.

Právě panoramatické zobrazení je problém, který se může řešit několika různými způsoby. V případě malíře je řešením široké plátno, na němž dokáže zachytit větší úhel pohledu, malíř je tedy limitován pouze šířkou plátna a nemá problém se zobrazením i celého úhlu pohledu (360°). Ale v případě použití fotoaparátu už takové triviální řešení neexistuje. U analogového fotoaparátu existuje pouze řešení pomocí speciálních optik. U digitálních fotoaparátů již existuje více variant stejně jako u analogových fotoaparátů můžeme zvolit použití speciální optiky, nebo také dokážeme se snímkem, nebo i více snímky dále pracovat pomocí tzv. postprocesingu (zpracování snímku po jeho pořízení).

Postprocesing pro panoramatické zobrazení pomocí digitálních zařízení můžeme aplikovat jak na statické, tak i dynamické snímky(video). Výhodou statických snímků je, že obraz je vždy stálý a v čase se nemění, tím se vytváření panoramatických

snímků zjednodušuje a existuje i mnoho nástrojů, které dokáží vytvořit z více snímků panoramatický obraz. Ale u dynamického zobrazení už nastávají právě komplikace se změnou obrazu snímků v čase, tyto komplikace jsou popsány také v této práci. Ale i pro toto dynamické zobrazení existují nástroje.

Samostatnou problematikou je zobrazení živých panoramat. Touto problematikou se zabývá tato práce, proto je jejím cílem vytvoření kompletího řešení pro pořízení těchto snímků a jejich zobrazení jako interaktivní aplikace v podporovaném webovém prohlížeči.

2 Panoráma

Panoráma je široký pohled na krajinu nebo na jakýkoliv objekt. Jedná se o kompozici se širokým záběrem. Velikost úhlu záběru této kompozice není přesně specifikována. Pojem panoráma je znám už z dávných dob a s panoramatickými snímky se setkáme například v zámcích a hradech, kde najdeme většinou výjevy z bitev. Panoramatické fotografie se často i objevují jako snímky krajin, měst a různých prostor.

2.1 Technika pořízení panoramatických snímků

2.1.1 Statické panoráma

Statické panoráma určují snímky, které se v rámci času nemění. Zejména reprezentuje pořízení snímků fotaparátem s panoramatickým objektivem nebo pořízení několika snímků navazujících v kompozici na sebe a jejich následným spojením. Pořizování snímků pomocí speciálních objektivů je tou nejjednodušší variantou, ale ovšem vyžaduje speciální a často nákladné optiky. Jeden z těchto objektivů je na obrázku 2.1, tento objektiv má pozorovací úhly 60° nad a pod horizontem a 360° horizontálně. Levnější variantou je vytvoření panoramatického snímku pomocí soustavy více snímků, které na sebe navazují, a jejich následném spojení. Spojení těchto snímků je zajištěno pomocí programu. Programů pro vytvoření panoramatických snímků existuje hodně, ale vždy záleží na jejich schopnostech. Některé vyžadují snímky nafocené s přesnou návazností, kde software tyto snímky pouze spojí. Lepší nástroje dokáží sestavit

panoramatický snímek i z nepřesného napojení jednotlivých snímků. V těchto programech se využívá spojování a natáčení obrazu dle několika specifických bodů spojovaných snímků. Jako příklad takového softwaru můžu uvést bezplatné Easypano, ArcSoft Panorama Maker, Hugin, Picassa a mnoho dalších, mezi placené patří například PanoramPlus X4, Adobe Photoshop. Funkce vytvoření panoramat z více snímků je dnes i implementována ve většině digitálních fotaparátů. Také lze nalézt velké množství aplikací pro chytré telefony.

Obrázek 2.1: Objektiv PanoPro

2.1.2 Dynamické panoráma

Dynamické panoráma oproti statickému využívá snímky, které se v čase mění. V tomto případě lze opět využít dvou možností pro pořízení snímků, stejně jako u statických využitím speciálních panoramatických optik a také i pořízením snímků více zařízeními a jejich následné spojení. V rámci optik jsou vlastnosti stejné jako statické snímky. Druhá varianta - spojování snímků má v tomto případě rozdíl zda se kamery ve scéně pohybují či ne.

V případě nepohyblivých kamer je spojování snímků jednodušší z důvodu stat-

ické pozice pro výpočet spojovacích bodů. V tomto případě stačí výpočet spojovacích bodů a natočení obrazu pouze pro jednu sadu snímků v jednom časovém okamžiku. Ostatní sady snímků pak budou pouze aplikovat vytvořené rozložení a natočení.

2.1.3 Typy panoramat

Cylindrické panoráma

Cylindrické panoráma zajištuje rotace kolem horizontální osy. V podstatě si toto panoráma můžeme představit jako obraz vytvořený uvnitř horizontální stěny válce. Pořízení snímků pro toto panoráma můžeme realizovat speciálním objektivem, nebo pomocí pořízení více snímků a jejich vzájemného stitchingu.

První způsob využívá tzv. catadioptrický objektiv. Jedná se o objektiv, který je složen z několika čoček a zakřivených zrcadel, které odrážejí svůj obraz do zorného pole optiky objektivu. Výhodou této metody je okamžité pořízení celého panoramatu a kvalita obrazu. Nevýhodou samozřejmě je nákladnost a složitost výroby takového objektivu.

Druhý způsob vyžaduje pořízení více snímků tak, aby snímač byl na stejném místě a otáčel se po horizontální ose. V rámci tohoto otáčení se pořizují postupně snímky. Pořízení těchto snímků lze provést tak, že jednotlivé snímky budou na sebe těsně napojeny nebo se budou navzájem částečně překrývat. V případě těsného napojení může nastat problematika pořízení celého 360°, kde poslední snímek musí být oříznut tak, aby nenastal překryv se snímkem prvním. Varianta částečného překrytí využívá stitchingu. Podstatným parametrem je velikost překrytí a vzájemné natočení jednotlivých snímků.

Krychlové panoráma

Krychlové panorama se vytváří uvnitř krychle na jednotlivé stěny. Na každou stěnu této krychle je potřeba samostatný snímek, z toho vyplývá, že je nutné pro vytvoření takového panoramatu šest snímků. Aby snímky na sebe navazovaly musí se pořídit z jednoho bodu - středu krychle. Dále je nutné, aby jednotlivé snímky byly pořízeny optikou s úhlem záběru 90°. Ve výsledném složeném obrazu lze zpozorovat pravoúhlé napojení jednotlivých snímků. Výhoda tohoto řešení je v nízké náročnosti na výkon při zpracování obrazu.

Půlkulové panoráma

Využívá efektu rybího oka, to znamená, že se snímá obraz polokoule, kde v jejím kruhovém obvodu je zdrojový obraz hustší a s přibližováním ke středu se postupně stává řídkým. To má za důsledek velké množství detailů uprostřed a malé detaily při úhlech zobrazení větším jak 90°. Pořízení snímku lze zajistit optikou s efektem rybího oka nebo lze dopočítat z více snímků pomocí stitchingu a deformace obrazu.

Kulové panoráma

Kulové panoráma využívá stejnou techniku zpracování jako půlkulové, ale s rozdílem, že se využijí 2 poloukoule se společným středem a opačným směrem vypouklé strany.

Stereografické panoráma

Typ panoramatu využívající stereografické projekce. Nejčastěji se tato projekce využívá na kulové plochy. Výhodou této projekce je zachování úhlu zobrazených objektů v kruhovém obvodu, to má za následek lepší zobrazení detailů v těchto místech oproti klasickému půlkulovému panoramatu.

Panografie

Technika zpracování několika snímků tak, že výsledný obraz je sestaven ze vzájemně se překrývajících se snímků tvořících panoramatické zobrazení. V této technice se neřeší ořezání překryvu, ale pouze jen natočení a umístění jednotlivých snímků.

3 Streamování videa

Stream nebo česky proud je technologie pro přenos audia a videa mezi serverem (zdrojem) a klienty. Existují 2 typy streamu, a to v reálném čase nebo tzv. způsobem video on demand. Stream v reálném čase je způsob, kdy se uživatelům posílá aktuální přehrávané nebo vysílané video. V tomto streamu se nelze pohybovat dopředu po časové ose. Druhý způsob video on demand je způsob streamování videa na vyžádání uživatele. To znamená, že streamovací server má uložené video, o které si uživatel požádá a server poté uživateli vysílá stream videa po médiu. Tento způsob umožňuje uživateli přesouvání se po časové ose videa.

Nejdůležitějším parametrem pro streamování videa je volba způsobu přenosu a kódování dat. Přenos dat určuje použitý protokol. Protokol je standard, který určuje průběh komunikace mezi zařízeními. Mezi nejpoužívanější protokoly patří HTTP, RTP, RTMP, Adobe HTTP Dynamic Streaming, Microsoft Smooth Streaming a Shoutcast. Kódování dat zajišťují kodeky. Každý kodek má své specifické vlastnosti, jako jsou kompresní poměr, rozlišení, řízení datových toků, rychlost komprese a další. Pro streamování videa je nejdůležitější vlastností právě rychlost komprese. Čím rychleji dokáže kodek data zkomprimovat, tím i výsledné video bude mít menší zpoždění oproti reálnému obrazu. Mezi vhodné formáty kodeků pro streamování videa lze zařadit H.264(MP4), H.265(HEVC), RealVideo, Theora, VP8, VP9 a WMV.

Pro streamování videa pro více klientů je duležité použití streamovacího serveru. Tento server zajišťuje distribuci zdrojů mezi připojenými klienty. Kromě této distribuce vytváří mezipaměť mezi jednotlivými zdroji a klienty. Tato mezipaměť oproti přímému vysílání videa umožňuje jednotlivým klientům plynulé přehrávání. Může nastat situace, kdy například dva klienti A a B se připojí k serveru a požadují stream videa. Server jim tento stream poskytne. U klienta B ale například nastane zpoždění příjmu o několik ms, aby klient B nepřišel o snímky, které v rámci tohoto zpoždění klient A přijmul, tak server v rámci použití mezipaměti klientovi B tyto snímky zašle a všechny následující snímky bude zasílat posunuté právě o toto zpoždění. Právě velikost mezipaměti je jedním z parametrů nastavení serveru. Dalšími vlastnosmi stream serverů je kontrola a řízení přístupu k médiím, řešení špatných připojení, statistika, řízení zdrojů. Mezi nejznámější servery pro streamování videa patří ze skupiny s volnou licencí například Icecast, FFServer, Red5 a Darwin Streaming server. Mezi servery s placenou licencí se řadí QuickTime Broadcaster, WOWZA Streaming engine, Adobe Media server, Evostream.

4 WebGL

Je Javascriptové aplikační rozhraní pro zobrazování 3D grafiky ve webových prohlížečích. Vývoj WebGL zajištuje nezisková organizace Khronos Group. WebGL pro svoji funkcí využívá shaderů realizovaných v jazyce GLSL (OpenGL Shading Language) postavených na aplikačním rozhraní OpenGL ES 2.0. Implementaci tohoto rozhraní najdeme ve všech dnes nejpoužívanějších prohlížečích. Pro zobrazení grafické scény se využívá HTML elementu canvas. Mezi zajímavé projekty realizované pomocí WebGL lze zařadit Google mapy, Zygote Body (model lidské anatomie), populární hru Angry birds, Autocad 360 WEB.

5 Existující řešení

Existuje několik řešení pro snímání 360° videa, ale všechna tato řešení vyžadují konkrétní hardware anebo zpracovávají 360° video až po pořízení jednotlivých videosekvencí.

V rámci rešerše kompletního řešení jsem vybral zařízení Allie od firmy IC Realtech, které bylo jako první veřejně prezentováno jako plně funkční řešení s podporou streamu v reálném čase. Toto zařízení se vyrábí v několika verzích, rozdíl mezi nimi je zejména v rámci rozlišení použitého fotosnímače. Jedná se o konstrukci dvou fotosnímačů umístěných zády k sobě s velmi širokoúhlou optikou. Výsledný panoramatický obraz je pak pomocí softwaru spojen a distribuován do aplikace, která tento obraz zobrazí. Ovládání je zajištěno orbitálním posuvem po obrazu anebo, pokud se jedná o zařízení s gyroskopem, ovládáno pomocí naklápění a rozpoznávání dat pávě z tohoto gyroskopu. Zařízení dle výrobce umožňuje streamování videa v reálném čase. Toto streamování probíhá opět do speciální aplikace od výrobce. V rámci produktového videa na veletrhu CES 2015 je tento stream prezentován, ale nenabízí dostatečný počet snímků za sekundu aby byl výsledný obraz plynulý.

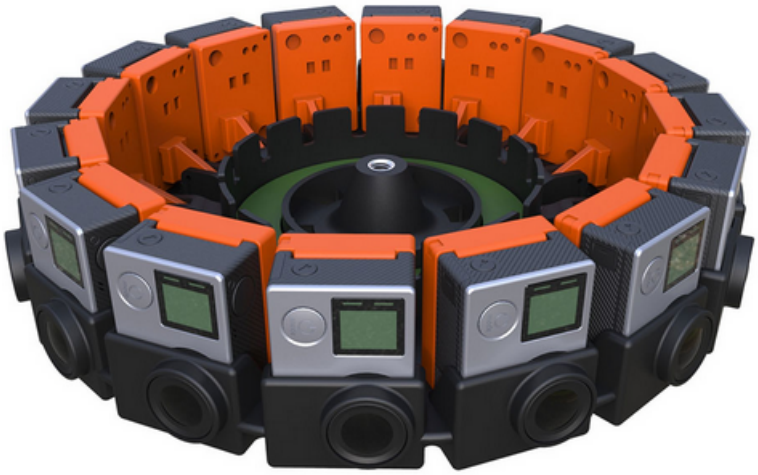
Obrázek 5.1: Verze řešení kamer Aliie od IC Realtech

Další ze zajímavých řešení nabízí firma Nokia se zařízením OZO. Jedná se o začínající projekt s vyrobeným prototypem. Zařízení dle oficiální zprávy výrobce umí nahrávat a i pomocí specializovaného softwaru zobrazovat stream tohoto videa v reálném čase. Zařízení se skládá z osmi kamer umístěných v kouli. Na obrázku 5.2 je ukázka tohoto zařízení.

Obrázek 5.2: Nokia OZO

Společnost Google pro 360° videa přizpůsobila svoji službu YouTube. Podpora těchto videí ale zatím není pro videa v reálném čase. Pro nahrání takového videa na servery YouTube musí být video natočené některou z doporučených nebo podporovaných kamer. Nahrávaní těchto videí lze také pomocí obyčejných kamer a pomocí specializovaného softwaru vytvořit podporované video. Příkladem tohoto softwaru může být například Kolor Autopano nebo VideoStitch. V tomto roce společnost Google na konferenci Google I/O představila řešení, které by v budoucnu mělo umožňovat zobrazení videa v reálném čase. Toto řešení nese název Google JUMP a přístup k němu je zatím jen pro vybrané úzké testovací skupiny. Jednou ze součástí

je YouTube360. Na obrázku 5.3 je rámeček JUMP camera rig osazený kamerami GoPRO.



Obrázek 5.3: Google JUMP camera rig

6 Návrh vlastního řešení

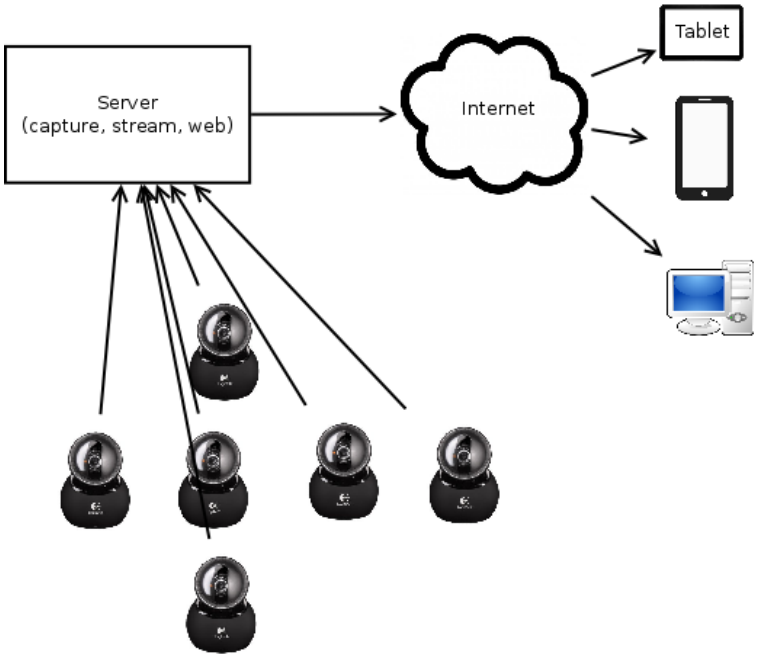
Cílem práce bylo seznámení se s problematikou streamování videa na internetu a 3D WebGL aplikací. Následně dané znalosti uplatnit pro návrh a realizaci WebGL aplikace pro prohlížení živých 360° panoramat.

6.1 Seznámení se s problematikou

Současná řešení nejčastěji zpracovávají tzv. offline video panoramata. To znamená, že zaznamenaná videa se nahrají a pak teprve zpracují, aby se mohla zobrazit. Software, který tyto videa zpracovává, ovšem nelze použít pro videa v reálném čase. Dalším problémem je, že i tato řešení často vyžadují specializovaný hardware pro své použití (viz kapitola Existující řešení). Streamování videa na webu je dnes již běžnou praxí, a to i v případě živých videí. V letošním roce spustil portál YouTube streamování 360° videí v rámci svého standardu, jedná se ale pouze opět o post-procesingem zpracované video. V případě streamování 360° videí se přenáší velké množství dat, to už ale také nebývá problém z důvodu neustálého zkvalitňování rychlosti připojení k internetu. Tato práce tedy přináší řešení pro spojení 360° videí a jejich živého přenosu.

6.2 Návrh vlastního řešení

Pro návrh řešení jsem zvolil krychlové panorama. Pro záznam snímků bude tedy potřeba 6 kamer umístěných ve společném středu. Celé řešení bude typu server-klient. Server bude zajišťovat zpracování obrazu z jednotlivých kamer tak, aby šly co nejlépe umístit na virtuální krychli. Dále bude mít na starosti distribuci těchto upravených snímků přes internet ke klientské části aplikace a své grafické uživatelské rozhraní. Server bude sestavený pro operační systém Linux. Klienstská část aplikace bude postavena na technologiích HTML páté verze, Javascriptu a aplikačního rozhraní WebGL. Díky těmto technologiím může být klientská aplikace co nejvíce multiplatformní. Klient řeší příjem streamu videa ze serveru a vytvoří právě onu virtuální krychli, na kterou se jako textura přilepí jednotlivé snímky z kamer. Uživateli bude poskytnuto tzv. orbitální ovládání ve středu této krychle. Ovládání klientské aplikace bude přizpůsobeno také pro uživatele s dotykovými zařízeními jako telefony a tablety. Návrh řešení popisuje obrázek 6.1.



Obrázek 6.1: Návrh řešení

7 Vlastní řešení

V následující části práce se budu zabývat popisem vytvořeného řešení. V Hardwarové části bude popsána testovací konfigurace, zejména použité kamery, jejich připojení, použité rozhraní a problematika připojení většího množství kamer k počítači. Na to naváže softwarová část, kde popíši vytvořenou aplikaci, její implementaci a nastavení. Zároveň v této části práce bude kapitola s návrhem řešení využívající stitchingu v reálném čase, které umožňuje volnější rozmístění kamer v prostoru. V závěrečné kapitole této práce jsou výsledky testování vlivů kodeků a náročnosti na přenosové médium v rámci testovací hardwarové konfigurace.

7.1 Hardwarová část

Už z návrhu řešení je jako výsledný typ panoramatu zvolené krychlové panorama a koncepce server-klient. Pro tento případ je tedy nutné použití šesti kamer snímajících daný prostor ze společného středu. Z důvodů dostupnosti řešení širokému množství uživatelů byly zvoleny webové kamery připojené pomocí rozhraní USB k počítači (serveru). Jednotlivé požadavky na webkamery a serverový počítač jsou popsány v následujících podkapitolách. V rámci klientského hardwaru nejsou kladeny žádné speciální požadavky, pouze zařízení disponující webovým prohlížečem podporujícím standard HTML5, Javascript a WebGL. V dnešní době existuje obrovské množství zařízení s touto podporou, od mobilních telefonů po chytré televize.

7.1.1 Požadavky na webkamery

Jako hlavní požadavek na webkameru se v případě krychlového panoramatu řadí zobrazovací úhel. V nejideálnějším případě je nejlepší zobrazovací úhel (FOV) 90°, právě z důvodu jednoduchého složení výsledného panoramatického obrazu. Výsledný obraz z jednotlivých kamer je poté pomocí softwaru složen do tzv. krychlové mapy. Další vlastnosti kamer jako rozlišení snímače, typ snímače, světelnost už záleží na individuálních potřebách. Výsledná aplikace dokáže pracovat s webovými kamerami připojenými přes libovolná rozhraní, ale s podmínkou, že v operačním systému se k nim bude přistupovat jako ke standardnímu video zařízení. Tento parametr v podstatě splňují veškeré USB webkamery. Jelikož pro výsledné panorama potřebujeme šest kamer, tak je vhodné, aby tyto kamery měly stejné vlastnosti, nejlépe byly totožné.

7.1.2 Použité webkamery

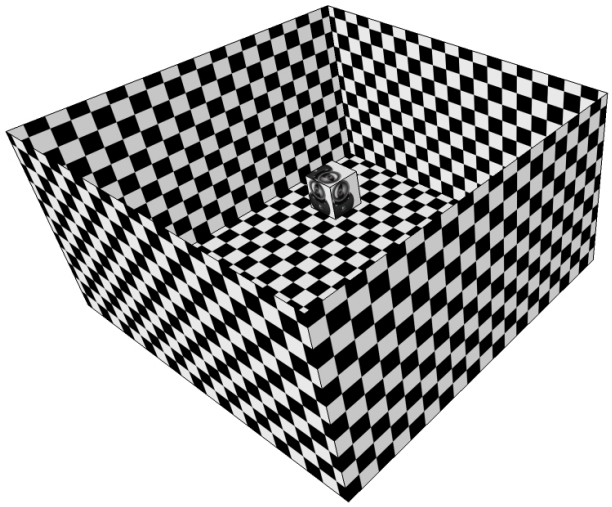
Pro testování aplikace bylo zapůjčeno šest webových kamer Logitech QuickCam PTZ z Ústavu informačních technologií a elektroniky Technické univerzity v Liberci. Parametry těchto kamer:

- Maximální rozlišení: 1600x1200 (2MP)
- Typ senzoru: CMOS
- Bitů na pixel: 12
- Zobrazovací úhel (FOV): 75° diagonálně
- Komprese: MJPEG
- Max. snímkovací rychlost:
  - 30 snímků/s při rozlišení 320x240, 640x480, 800x600
  - 15 snímků/s při rozlišení 960x720, 1280x960, 1600x1200
- Rozhraní: High Speed USB 2.0

Obrázek 7.1: Webkamera Logitech QuickCam PTZ

Rozmístění kamer

Pro usnadnění přesného rozmístění, tak aby byl snímán prostor kopírující pohled ze středu krychle na její jednotlivé strany, je vhodné kamery kalibrovat. Jednou z možností kalibrace je šachovnicové schéma. Použití šachovnice pro krychlové panorama ukazuje obrázek 7.2. Samotná kalibrace se provádí natočením a umísťováním kamer tak, aby jednotlivě všechna krajní pole šachovnice na sebe v náhledu přímo navazovala a jejich natočení bylo shodné.



Obrázek 7.2: Kalibrace pomocí šachovnice pro krychlové panorama

7.1.3 Požadavky na server

Pro variantu krychlového panoramatu s přesným rozmístěním kamer popsaných výše není minimální hardwarová konfigurace příliš náročná. Pro spuštění výsledné aplikace postačí současná konfigurace běžné kancelářské sestavy. Jedinou podmínkou je dostatek rozhraní pro připojení daného množství kamer tak, aby všechny mohly pracovat najednou. Pro návrh řešení využívající stitching v reálném čase je výkon sestavy přímo zavislý na kvalitě a plynulosti výsledného obrazu.

7.1.4 Použitá konfigurace serveru

Testovací konfigurace serveru má tyto parametry:

- Procesor: Intel Core i3 4010U Hasswell (1.7 GHz)
- Operační paměť: 4GB DDR3
- Pevný disk: 500GB (5400rpm) + 16GB SSD Cache
- Použité rozhraní pro připojení kamer: 2 xUSB 3.0
- Další zařízení: aktivní USB3.0 rozbočovač

7.1.5 Připojení více kamer na rozhraní USB

Z důvodu, že pro testovací konfiguraci byly zvoleny webkamery na rozhraní USB, jsem se setkal právě s problematikou připojení více webkamer. Problém nastává v šířce komunikačního pásma sběrnice USB, jelikož se standardně snímky z webkamer odesílají do počítače v nekomprimovaném formátu, tak pro použití většího rozlišení snímků a více kamer tato šířka pásma je nedostačující. Částečným řešením je využití novějšího standardu USB3.0, který má přenosové pásmo širší. Ovšem pokud chceme využívat nekompresní přenos snímků, nastane ještě problém s nemožností vytvoření více komunikačních kanálů na jednom řadiči posílající tato surová data. Optimálním řešením by tedy bylo mít na každou připojenou webkameru samostatný řadič, ovšem tuto možnost nelze aplikovat na všech zařízeních z důvodu absence adekvátních připojení. Řešení se tedy nachází již ve výběru kamer, a to takových, aby měly hardwarovou podporu kompresního přenosu. Nejčastější kompresní přenos zajišťuje komprese typu MJPG (motion JPEG). Výhoda této komprese je v její rychlosti, která je zajištěna pomocí intra-frame komprese. Nejlepší kompresní poměr, kterého lze docílit, je 1:20. Tato komprese, pokud není nastavena jako výchozí typ

přenosu na webkameře, se musí aktivovat softwarově před vytvořením datového komunikačního kanálu. V tabulce 7.1 je přehled zatížení jednou testovací kamerou přenosového pásma pro vybraná rozlišení bez a s kompresí MJPG. Výpočet je dán následujícím vzorcem:

Zatížení pásma = snímků za sekundu (fps)∗šířka snímku∗výška snímku∗bitů na pixel

Komprese MJPEG je ve výpočtu v tabulce 7.1 brána jako maximální možná. Na reálných snímcích bude komprese nižší.

Rozlišení	Snímků za sekundu			
	15		30	
	Bez komprese	MJPEG	Bez komprese	MJPEG
320x240	14 Mbit/s	0,69 Mbit/s	28 Mbit/s	1,38 Mbit/s
640x480	55 Mbit/s	2,76 Mbit/s	111 Mbit/s	5,53 Mbit/s
800x600	86 Mbit/s	4,32 Mbit/s	173 Mbit/s	8,64 Mbit/s
960x720	124 Mbit/s	6,22 Mbit/s	249 Mbit/s	12,44 Mbit/s
1280x960	221 Mbit/s	11,06 Mbit/s	442 Mbit/s	22,12 Mbit/s
1600x1200	346 Mbit/s	17,28 Mbit/s	691 Mbit/s	34,56 Mbit/s

Tabulka 7.1: Zatížení přenosového pásma

Aby bylo možné připojit všech šest webkamer k testovací konfiguraci serveru, byl zvolen pro zvýšení počtu portů aktivní USB3.0 rozbovač. Volba právě rozbočovače standardu USB3.0 byla důležitá, aby byla zaručena požadovaná datová propustnost.

7.1.6 Požadavky klienta

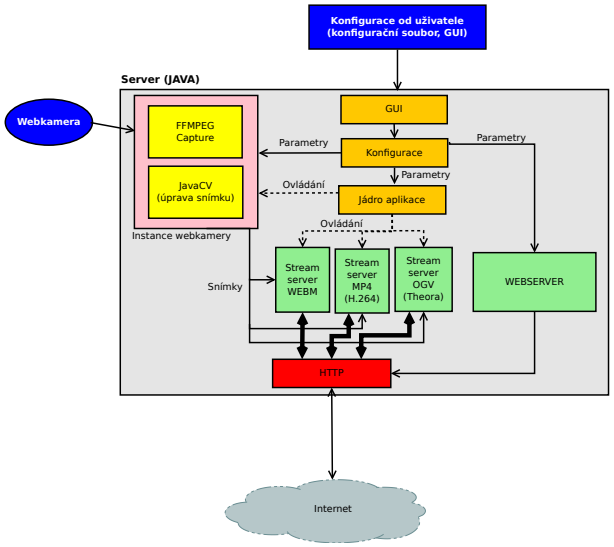
Klienstká aplikace vzhledem ke svému univerzálnímu použití na zařízeních disponujících webovým prohlížečem s podporou HTML5, Javascriptu a WebGL příliš omezujících parametrů nemají. Omezení se ovšem nalézá v nutnosti podpory grafického adaptéru pro OpenGL ES 2.0. Tuto podporu ale najdeme ve většině cílových zařízení zmíněných výše.

7.2 Softwarová část

Z návrhu řešení je použita koncepce server-klient. Jak už bylo výše zmiňováno v návrhu, server bude zařizovat zpracování obrazu z kamer a jeho streamování. Klient přijme stream a přidá grafické uživatelské rozhraní pro pohyb po výsledném panoramatu.

7.2.1 Server

Serverová aplikace je realizována pro operační systém Linux. Jako programovací jazyk byla zvolena Java, z důvodu jednoduché přenositelnosti kódu pro případného budoucí rozšíření na jiné platformy. Pro spuštění aplikace je nutné tedy Java JDK minimálně verze 1.8, FFmpeg minimálně verze 2.2.x se všemi zavislými knihovnami (zejména jsou nutné knihovny libavutil, libavcodec, libavdevice) a VLC(konkrétně postačuje konzolová verze CVLC) . Blokové zobrazení funkčních částí aplikace popisuje obrázek 7.3. Jednotlivé bloky a samotná funkce aplikace budou popsány v následujících podkapitolách. Samotná aplikace a zdrojové kódy jsou zaznamenány na přiloženém CD této práce.



Obrázek 7.3: Funkční části serveru

Funkce aplikace

Dle obrázku 7.3 aplikace umožňuje dvě možnosti nastavení parametrů. Za prvé pomocí konfiguračního souboru, tato varianta je vhodná pro nasazení na terminálovém (negrafickém) serveru. A za druhé pomocí grafického uživatelského rozhraní. Obě možnosti budou popsány níže. Po nastavavení konfigurace se vytvoří instance jednotlivých připojených webkamer. Každá tato instance má pro sebe vytvořený samostatný proces přijmu snímků pomocí FFmpeg nebo CVLC. Pro budoucí rozšíření je v každé instanci webkamery implementován blok úpravy snímků pomocí JavaCV, o tomto rozšíření se budu zmiňovat v kapitole Návrh řešení využívající stitching. Snímky z každé webkamery poté aplikace dle zadaných parametrů odesílá vhodnému streamovacímu serveru. Tento server je vždy vybrán dle zvoleného výstupního kodeku. Pro formáty WEBM a MP4 je použit upravený open source stream server Stream-M. Pro OGV se využívá řešení Icecast 2 distribuované pod licencí GNU GPL v2. Veškeré výstupní streamy využívají přenosového protokolu HTTP. Aplikace zároveň obsahuje integrovaný webserver, který hostí klientskou část aplikace a zároveň poskytuje konfiguraci této části.

Instance webkamery

V této části se řeší veškerá obsluha webkamer. V programu je každá webkamera reprezentací instance jednoduché třídy *webcam*, která má dva atributy - název a označení zařízení v operačním systému. Při spuštění aplikace se vytvoří právě tyto instance dle aktuálně připojeného hardwaru. Pro vyhledání kamer připojených k počítači slouží metoda *getListWebcams()*, pro svoji funkci využívá programu *v4l2-ctl*. Samotné získání snímků z kamer a jejich nastavení poté obstarává výkonná část v jádru aplikace reprezentována třídami *commandMaker* a *streamProcess*, tyto třídy budou popsány dále.

Stream server

Aplikace v sobě integruje dvě možnosti stream serveru. První možnost pomocí open source nástroje StreamM, který zajišťuje stream pomocí kodeků WebM a MP4(h.264). A druhá možnost slouží pro kodek OGV(Theora), kde se využívá nástroje Icecast. Aplikace vytvoří dle nastavené konfigurace parametry k těmto nástrojům a spustí jejich instanci za pomoci třídě *serverProcess*. Samotné nastavení a předání vstupních snímků poté zajištuje opět příslušná instance třídy *commandMaker*. Dle typu zvoleného kodeku, a tudíž i streamovacího nástroje aplikace vybere příslušný nástroj pro získání snímků. Pro kodeky WebM a MP4 se jako zdrojový nástroj vybere *ffmpeg*, pro kodek OGV je využito *cvlc*. V těchto obou případech aplikace nastaví vhodné parametry kamerám a nástrojům předá jejich konfiguraci. V obou případech stream probíhá po protokolu HTTP na zvoleném portu v konfiguraci aplikace. Mezi nejdlůležitější parametry patří rozlišení, snímkovací rychlost, nastavení kompresního přenosu a synchronizace klíčových snímků.

Jádro aplikace

Jádro aplikace zpostředkovává řízení všech jednotlivých součástí. Zajišťuje řízení a sledování stavu jednotlivých součástí dle blokového diagramu na obrázku 7.3. Samotné metody jsou obsaženy v hlavní třídě celé aplikace. Každá jednotlivá součást aplikace obsahuje metodu pro zjištění jejího aktuálního stavu a metodu pro kompletní zastavení. Třídy, které aplikace obsluhuje, jsou popsány zde:

- *commandMaker* - určuje formát a parametry daným příkazům dle nastavené konfigurace
- *serverHTTP* - integrovaný jednoduchý HTTP server
- *serverProcess* - obsluha a řízení streamovacích serverů
- *settingsClient* - nastavení klienstké části del parametrů serveru

- *streamProcess* - obsluha a řízení samotného streamu
- *webcam* - třída určující vlastnosti kamer

Každá instance kamery a aktuálně používaný stream server pro svou práci vytvoří samostatný proces. Výhodou tohoto rozvržení je, že v případě chyby jednoho z těchto klíčových prvků nenastane pád celé aplikace, ale pouze jednotlivé části. Aplikace právě díky kontrolním metodám nad jednotlivými procesy graficky oznámí uživateli stavy daných funkcí. Uživatel poté může zkusit znovu spustit právě jenom určitou část aplikace. Výměna dat(snímků) mezi streamovacím serverem a streamem kamer se provádí vždy pomocí rour mezi jednotlivými vytvořenými procesy.

Webserver

Funkci zajišťuje třída serverHTTP. Jedná se o nenáročný server, který slouží pro účely hostování klientské aplikace. Výchozí adresář var/www se nachází ve zdrojové složce aplikace. V této složce se i nachází zdrojové soubory klientské části aplikace. Maximální počet spojení je omezen na 201 a určuje ho proměnná *maxConections*, aktuální počet spojení je dán proměnou *actualConections*.

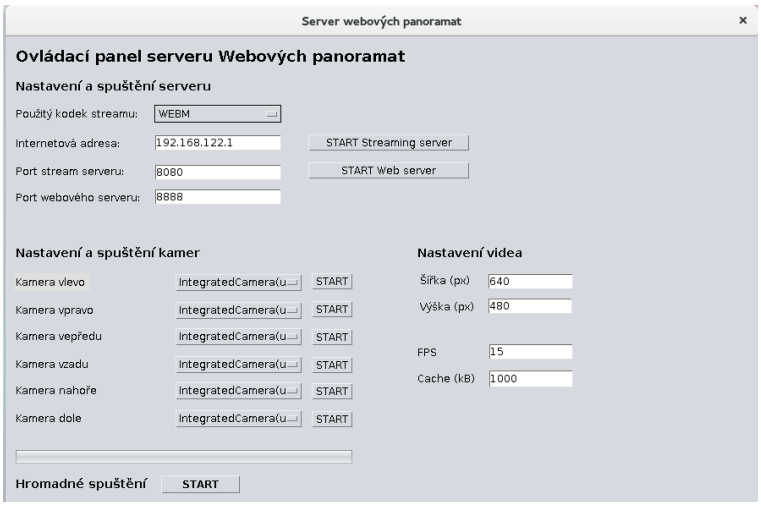
Konfigurační soubor

Jak už bylo výše zmíněno aplikace lze spouštět i přímo z konzole. Tato funkce je vhodná pro terminálové servery. Na to, aby šla aplikace tímto způsobem spustit, musí se jí jako parametr předat konfigurační soubor. Vzorový konfigurační soubor je obsažen na přiloženém CD této práce. Pokud aplikace bude mít právě tento parametr, automaticky se nezobrazí grafické uživatelské rozhraní ale začnou se postupně s několika sekundovým zpožděním spouštět jednotlivé funkce programu s parametry nastavenými právě v konfiguračním souboru.

Grafické uživatelské rozhraní

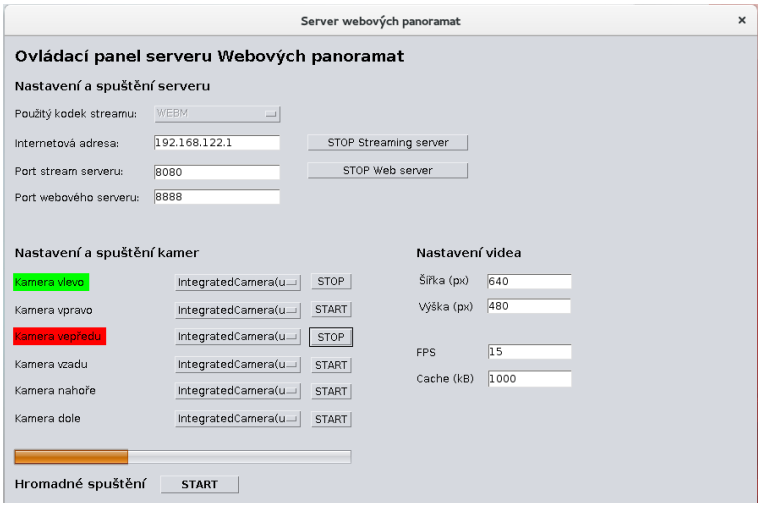
Grafické uživatelské rozhraní (GUI) je koncipováno tak, aby po uživateli nevyžadovalo žádné velké možnosti nastavení. Obrázek 7.4 znázorňuje obrazovku aplikace při spuštění. GUI se skládá ze tří částí možných nastavení, konkrétně jde o nastavení serverů, nastavení kamer a nastavení videa. Nastavení serverů určuje volba použitého kodeku, internetové adresy, port stream a webového serveru. Internetová adresa je důležitý parametr pro nastavení klientské aplikace, pokud by totiž server neměl přímou adresu do internetu a byl například umístěn za NATem, tak adresa zadaná v tomto poli je právě adresou viditelnou z internetu. Standardně při spuštění programu se volí aktuální adresa výchozího adaptéru. Adresa v tomto poli může být zadána buď v klasickém numerickém formátu, nebo i pomocí doménového názvu. Nastavení portů lze nastavit jak pro stream server, tak i pro webový server. Standardně jsou voleny porty pro stream server 8080 a pro web server 8888. Nastavení videa určuje základní parametry videa, jako jsou rozměry videa (šířka a výška), tyto rozměry ovlivňují jak samotný příjem snímků z webkamer, tak i velikost zobrazovaného videa v klentské části. Parametr FPS určuje jaká bude nastavená snímkovací rychlost webkamer a samotného přenosu. Hodnoty snímkovací rychlosti přímo ovlivňují požadovanou rychlost přenosu, výsledky testování přenosových rychlostí budou uvedeny v závěru této práce. Nastavení cache určuje velikost vyrovnácí paměti streamovacího serveru. V nastavení kamer se nastavuje přidělení jednotlivých kamer připojených k počítači ke stranám výše zmíněné virtuální krychle. Jednotlivé části aplikace se spouštějí buď samostatně pomocí příslušných tlačítek, nebo pomocí takzvaného hromadného spuštění v dolní části GUI. V případě ručního spouštění je vhodné, aby se jednotlivé kamery spouštěly s časovým zpožděním alespoň pět sekund. V případě automatického spuštění aplikace automaticky zpožděně spouští jednotlivé kamery.

GUI umožňuje i kontrolu funkčnosti jednotlivých streamů. Jak už bylo psáno výše v popisu aplikace, zajišťuje to vizuální signalizace. Pokud daný stream kamery byl spuštěn a funguje, je pod jeho názvem zbarveno pozadí zeleně, pokud je stream



Obrázek 7.4: Grafické uživatelské rozhraní

spuštěn, ale nefunguje, zbarvení má červenou barvu, a pokud stream nebyl spuštěn pozadí, splývá s pozadím GUI. Obrázek 7.5 ukazuje možné situace.

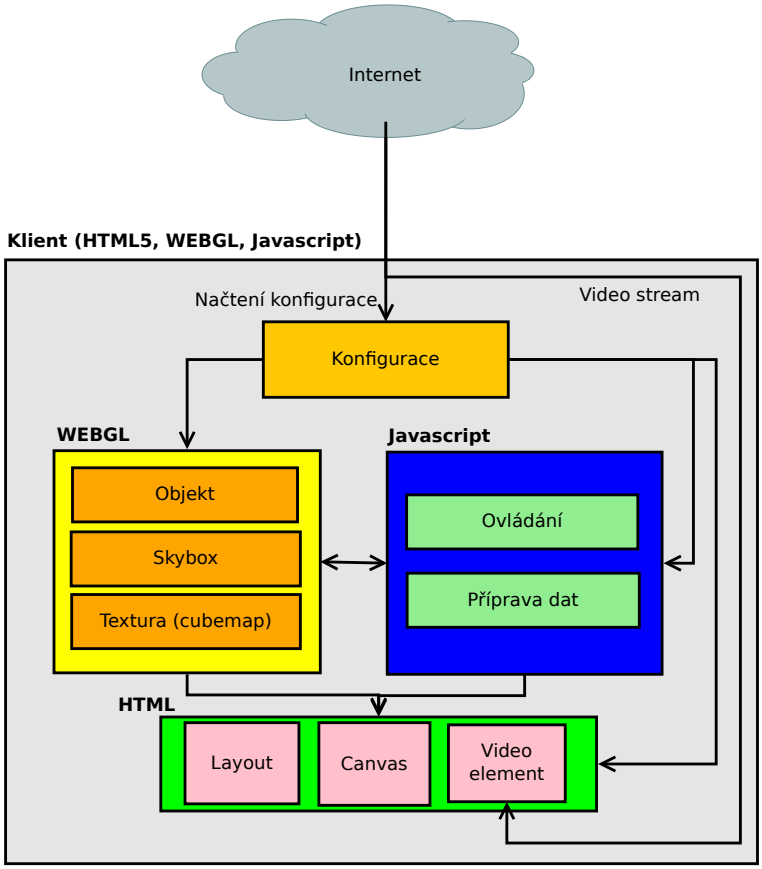


Obrázek 7.5: Grafické uživatelské rozhraní - spuštěný stream

7.2.2 Klient

Klientská aplikace využívá jazyky HTML5, CSS3, JavaScriptu, knihovny THREE.js a aplikačního rozhraní WebGL. Z tohoto důvodu je možné ji spustit na širokém množství platforem, které mají webový prohlížeč podporující tyto jazyky a aplikační

rozhraní WebGL. Blokové zobrazení funkčních částí aplikace popisuje obrázek 7.6. Jednotlivé bloky a samotná funkce aplikace budou popsány v následujících podkapitolách. Samotná aplikace a zdrojové kódy jsou obsažené na přiloženém CD této práce. Klientská aplikace vytvoří z dodaných streamů ve WebGL objekt krychle, na kterou se jednotlivé streamy budou zobrazovat dle určených stran. Uživatel se pohybuje pomocí ovládacího rozhraní v prostoru krychle (uvnitř krychle).



Obrázek 7.6: Funkční části klienta

Implementace klienta

Klientská aplikace je hostována na straně serveru v integrovaném webserveru. Z toho důvodu není nutné na klientské zařízení instalovat žádnou aplikaci, ale pouze přistupovat k této aplikaci pomocí webového prohlížeče s výše zmíněnou podporou. Adresa přístupu k aplikaci je dána nastavením serveru, konkrétně v poli *Internetová adresa*

a také portem webového serveru zadaným v poli *Port webového serveru*.

Konfigurace

Jak už bylo zmýněno výše, konfigurace klientské aplikace se vytvoří společně s nastavením serveru. Tudíž není nutné, aby uživatel klientské aplikace musel nastavovat některé z parametrů. Jelikož se konfigurace vyměňuje mezi dvěma různými aplikacemi, zvolil jsem jako formát výměny dat jazyk JSON. Základní struktura konfigurace může vypadat například takto:

```
{"defaultPath":"video/video.webm","globalCodec":"video/webm",
"globalWidth":320,"globalHeight":240,
"pathTextureRight":"http://192.168.0.30:8080/consume/videoRight",
"pathTextureLeft":"http://192.168.0.30:8080/consume/videoLeft",
"pathTextureFront":"http://192.168.0.30:8080/consume/videoFront",
"pathTextureBack":"http://192.168.0.30:8080/consume/videoBack",
"pathTextureTop":"http://192.168.0.30:8080/consume/videoTop",
"pathTextureBottom":"http://192.168.0.30:8080/consume/videoBottom"}
```

Parametr `defaultPath` určuje, jaké video se bude přehrávat v případě neaktivního streamu pro jednotlivé strany. Další parametry jako `globalCodec`,`globalWidth` a `globalHeight` udávají použitý kodek a rozměry videa na jednotlivých stranách virtuální krychle. Posledními parametry jsou adresy k jednotlivým streamům označované jako `pathTexture` a požadovaná strana.

Knihovna THREE.js

Knihovna THREE.js je volně šiřitelná knihovna implementující aplikační rozhraní WebGL. Tato knihovna obsahuje nejčastěji používané metody a objekty používané právě ve WebGL, vytvořené tak, aby se k nim přistupovalo co nejjednodušeji.

Struktura a funkce klientské aplikace

Jednotlivé moduly v rámci klientské aplikace na sebe navazují. Řídící část obstarává Javascript. Při spuštění aplikace se načte nejprve defaultní tmavý vzhled pozadí, zjistí se velikost zařízení a nastaví se zobrazení. Zjištění velikosti a zároveň nastavení zobrazení tak, aby se zabránilo změnám velikosti v průběhu zobrazování streamu, je zajištěno pomocí meta tagu `viewport`. Nastavení tohoto tagu je následují:

```
<meta name="viewport" content="width=device-width, user-scalable=no,
minimum-scale=1.0, maximum-scale=1.0">
```

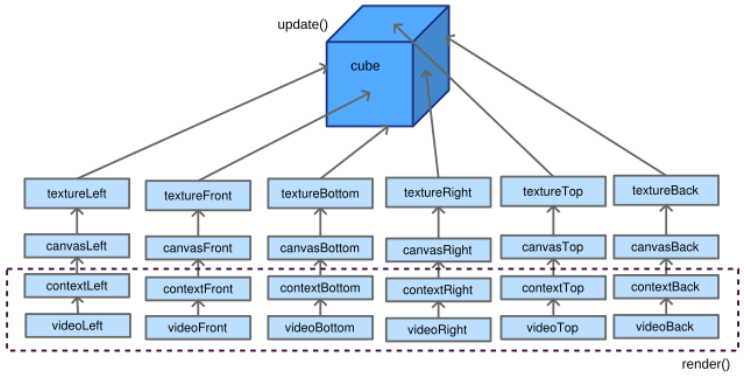
V těle stránky poté následuje element `div` s identifikátorem `container`, do kterého se budou umísťovat vygenerované elementy pro vykreslování grafiky. Ještě před spuštěním samotného zpracování panoramatu je provedena kontrola přítomnosti a správného spuštění WebGL, bez kterého se aplikace neobejde. Po této kontrole následuje načtení parametrů aplikace z konfiguračního souboru uloženém na serveru. Pro toto načtení se využívá `XMLHttpRequest()`pro načtení souboru JSON popisovaného výše.

Po této základní inicializaci se zpracuje konfigurace. Prvním krokem je příprava textur jednotlivých stran virtuální krychle. Tuto přípravu zajišťuje funkce `getTexture()` pro každou stranu samostatně. Funkce vytvoří HTML5 element `video` a přidělí mu nastavení, zároveň načte zdroj tohoto videa a spustí jeho přehrávání. Zároveň se vytvoří další element `canvas`, který slouží jako plocha pro kreslení grafiky do webové stránky. Velikost této plochy je stejná jako velikost videa. Abychom mohli použít tento canvas pro zobrazení videa, musíme získat přístup k jeho obsahu, pro získání přístupu k tomuto obsahu slouží v aplikaci proměnná `context` opět vytvořená pro každou stranu samostatně. V tomto případě už je vše připravené na vytvoření textury pro krychli, která se vytvoří ve funkci `init()`. Samotné vytvoření textury pak

má jako parametr zdroje vždy příslušný vytvořený `canvas`. V následující ukázce kódu je zobrazena funkce získání textury pro levou stranu krychle.

```
function getTextureLeft(path){
     videoLeft = document.createElement( 'video' );
     videoLeft.id = 'videoLeft';
     videoLeft.type = globalCodec;
     videoLeft.src = path;
     videoLeft.load();
     videoLeft.play();

     canvasLeft = document.createElement( 'canvas' );
     canvasLeft.id = "canvasLeft";
     canvasLeft.width = globalWidth;
     canvasLeft.height = globalHeight;

     contextLeft = canvasLeft.getContext( '2d' );
     contextLeft.fillStyle = '#000000';
     contextLeft.fillRect( 0, 0, canvasLeft.width, canvasLeft.height );

     textureLeft = new THREE.Texture( canvasLeft );
     textureLeft.minFilter = THREE.LinearFilter;
     textureLeft.magFilter = THREE.LinearFilter;

     return textureLeft;}
```

Celou tuto srukturu skládání textury pro jednotlivé strany vystihuje diagram na obrázku 7.7.

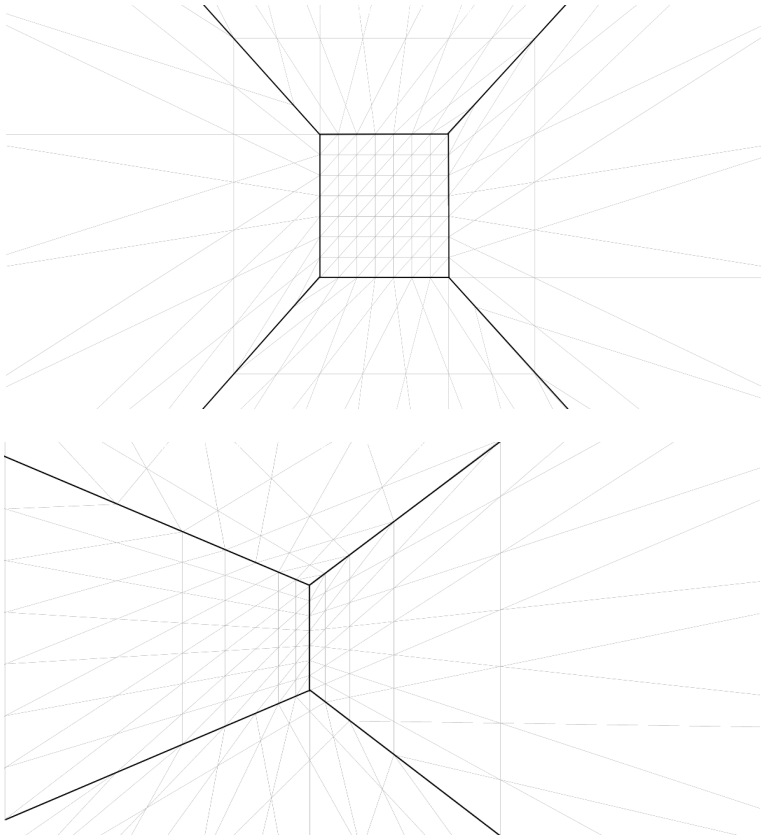


Obrázek 7.7: Diagram otexturování stran krychle

Inicializaci scény, kamery a samotné krychle, na kterou se budou vkládat vytvořené textury, obstarává funkce `init()`. V této metodě se vytvoří scéna. To je v podstatě jen prostor, do kterého se vkládají objekty. Do této scény se poté přidá kamera, neboli pohled do scény. Úhel pohledu kamery je zvolen na 75°. Šířka a výška tohoto pohledu se kvůli možnosti zobrazení na více typech zařízení vypočítává z aktuální velikosti okna, to má za důsledek stále optimální zobrazení pohledu. Kamera je po vytvoření objektu krychle přesunuta dovnitř do středu krychle. Před vytvořením objektu krychle je nutné aplikovat vytvořené textury uvedené výše jako samostatné plochy, které budou pokrývat strany krychle. Toho je docíleno zmapováním výše uvedeného vytvoření textur na tyto plochy, opět pro jednotlivé strany samostatně. Poté se vytvoří samotný objekt krychle a umístí se na něj jako jednotlivé textury právě tyto plochy.

Pro renderování (znovunačtení) scény se využívá `CanvasRenderer()`, který volá funkci `render()`. Ve funkci render se obnovují jednotlivé textury a pohyb krychle. V případě pohybu po tomto panoramatu se jednotlivé textury stran deformují tak, aby vytvářely dojem pohybu v prostoru. Nejlépe tuto deformaci vystihují tzv. drátová schémata vytvořeného modelu panoramatu na obrázku 7.8 , jednotlivé

strany krychle jsou zvýrazněny tučně.



Obrázek 7.8: Drátová schémata modelu panoramatu

Ovládání

V případě spuštění klientské aplikace se automaticky spustí automatické posouvání po panoramatu, kde rozhled kolem dokola trvá přibližně tři minuty. Aplikaci však lze ovládat dvěma způsoby, a to pomocí myši, anebo pomocí dotykové obrazovky. Pomocí myši při stisknutém levém tlačítku a následném pohybu se posouváme v rámci obou os. Přibližování a oddálování pohledu se provádí kolečkem myši. Při dosažení maximálního oddálení a přiblížení se pohled převrátí. Dotykové ovládání funguje na přirozeném pohybu prstu po zobrazeném pohledu. V podstatě kamera sleduje a kopíruje pohyb prstu po obrazovce. Přiblížení a oddálení v tomto případě ovládání není implementováno.

Výsledné zobrazení

Obrázek 7.9 znázorňuje část výsledného zobrazení v místnosti. Obrázek 7.10 představuje také část výsledného zobrazení, ale ve venkovním prostoru. Videa pohybu po těchto zobrazeních jsou umístěná na přiloženém CD ve složce ukázky. Kvalita návaznosti výsledného zobrazení je vždy přímo závislá na přesném rozmístění kamer.

Obrázek 7.9: Ukázka výsledného zobrazení v místnosti

Obrázek 7.10: Ukázka výsledného zobrazení venku

7.2.3 Návrh řešení využívající stitching

Nad rámec řešení této práce jsem testoval řešení, které by využívalo stitching snímků v reálném čase. Výhoda tohoto řešení spočívá ve volném rozmístění kamer bez nutnosti kalibrace. V tomto řešení je při inicializaci kamer na straně serveru přijat jeden snímek z každé kamery. Tyto snímky se pomocí funkce stitchingu obsažených v knihovně JavaCV (OpenCV) spojí a získají se jejich koordinační parametry, natočení a oříznutí. Tyto kordinační parametry se pak budou aplikovat na všechny následující snímky. Po určitém časovém intervalu se opět vybere jeden snímek z každé kamery a znovu se přepočítají tyto parametry. Přepočet nastává z důvodů změn scény a případného pohybu kamer. Nevýhodou ovšem je, že výsledné snímky mají vždy proměnlivé rozlišení. Toto proměnlivé rozlišení pak je problematické pro stream výsledných snímků na stranu klienta. Testovací aplikace dokázala odeslat jen velmi malé množství snímků, a tedy z důvodů nestability není ani obsahem v této práci.

8 Závěr

Při vytváření aplikace pro zobrazování 360° panoramat se autor snažil vytvořit řešení, které bude jednoduché pro uživatele. Jak z hlediska umístění kamer, tak i z hlediska jednoduchého nastavení aplikace. Hlavním cílem bylo vytvoření prohlížeče těchto panoramat dostupném přes webové rozhraní.

8.1 Složitost řešení

Výsledná aplikace se skládá ze dvou částí serveru a klienta. Obě tyto části poskytují stabilní základ pro možnost dalšího rozšíření. Při vytváření této práce musel autor řešit několik problémů, které se objevovaly postupně s vývojem. Mezi nejpodstatnější problémy patřilo zprovoznění více webových kamer na jednom počítači, odesílání velkého množství dat na stranu klienta a nastavení rozmístění a kalibrace kamer.

8.2 Vliv kodeků a hardwarová náročnost

Použité kodeky byly vybrány, tak aby měly co nejlepší vlastnosti pro použití streamování videa na internetu. Vybrané kodeky mají mírné odlišnosti v hardwarové náročnosti. Odlišnosti jsou způsobené také i rozdílnými použitými nástroji pro dané kodeky. Pro srovnání výkonu jednotlivých kodeků na zvoleném testovacím hardwaru serveru uvedeném v kapitole 7.1.4 bylo provedeno měření spotřebované operační paměti a procentuální konzumace výkonu procesoru. Jako testovací nastavení bylo zvoleno ro-

zlišení 320x240 při snímkovací rychlosti 10 snímků za sekundu a velikost vyrovnávací paměti 1 000KB. Všechna výsledná data jsou průměrné hodnoty za časový interval jedné minuty. K serveru byl v době měření připojen jeden klient. Tyto výsledky jsou uvedeny v tabulce 8.1

Kodek	WEBM	MP4	OGV
Zatížení CPU	56%	56%	15%
Spotřebovaná RAM	236,84 MB	111,6 MB	135,18 MB

Tabulka 8.1: Zatížení serveru

Pro měření na straně klienta bylo použito stejné nastavení. V tomto případě měření probíhalo na dvou testovacích sestavách. Pro klientskou část byla provedena dvě měření. První slouží pro otestovaní implementace v různých prohlížečích na platformě Windows (verze 10) a Linux (Fedora 22). Bylo tedy provedeno měření procentuální konzumace výkonu a velikosti spotřebované paměti na prohlížečích Microsoft Internet Explorer 11, Mozzila Firefox a Google Chrome verze 44. Prohlížeč Internet Explorer pro chod kodeku WEBM musí mít integrované rošíření pro podporu tohoto formátu. Výsledky tohoto měření jsou uvedeny v tabulce 8.2 pro systém Windows a v tabulce 8.3 pro systém Linux. Testovací sestava pro operační systém Linux je stejné konfigurace jako uvedená testovací sestava serveru. Sestava pro systém Windows má následující parametry:

- Procesor: Intel Core 2 Duo T5870 (2.00 GHz)
- Operační paměť: 2GB DDR2
- Pevný disk: 500GB (5400rpm)

Prohlížeč	Internet Explorer			Mozzilla Firefox			Google Chrome		
Kodek	WEBM	MP4	OGV	WEBM	MP4	OGV	WEBM	MP4	OGV
Zatížení CPU	29%	26%	34%	31%	34%	28%	28%	36%	20%
Spotřebovaná RAM	56 MB	54 MB	78 MB	153 MB	157 MB	61 MB	61 MB	82 MB	86 MB

Tabulka 8.2: Výkon prohlížeče Windows

Prohlížeč	Mozzilla Firefox			Google Chrome		
Kodek	WEBM	MP4	OGV	WEBM	MP4	OGV
Zatížení CPU	11%	17%	10%	33%	43%	34%
Spotřebovaná RAM	126 MB	138 MB	129 MB	195 MB	215 MB	221 MB

Tabulka 8.3: Výkon prohlížeče Linux

Druhé provedené měření poukazuje na zatížení přenosového pásma. Bylo zvoleno měření množství přenesených dat za daný časový interval. Časový interval byl určen na jednu minutu. Výsledky měření jsou znázorněny v tabulce 8.4.

Kodek	WEBM	MP4	OGV
Množství dat/min	6 048 kB	8 520 kB	14 400 kB

Tabulka 8.4: Množství přenesených dat

V rámci provedených měření se jako nejvhodnější použitelný kodek jeví WEBM. Oproti naměřeným hodnotám má vůči kodeku MP4 podstatnou výhodu právě v rámci streamování dat, a to konkrétně v rychlém načtení příchozího streamu. Tato vlastnost je způsobena velmi nízkou velikostí startovací vyrovnávací paměti tzv. precache. Nevýhodou tohoto formátu je nutnost rozšíření pro prohlížeč Internet Explorer. Oproti kodekům WEBM a MP4 má kodek OGV nízké zatížení serveru. Rozdíl v kvalitě výstupního videa nebyl v porovnání všech těchto tří možností kodeků znatelný. Pro použití na zařízeních majících omezení v rychlosti a množství dat se opět nejlepší výsledky ukázal kodek WEBM.

8.3 Celková funkčnost a možná vylepšení

V rámci této práce bylo vytvořeno řešení server-klient, kde server zprostředkovával snímky z kamer a zajišťoval jejich stream ke klientovi. Klient byl vytvořen tak, aby byl přístupný z webového rozhraní a zajišťoval jednoduchost ovládání. Celkové

řešení bylo odzkoušeno a v rámci provedených měření zaznamenaných výše byly zpracovány poznatky z těchto měření. Aplikace je vhodná pro použití streamování velkých prostorů a krajin. Autorovým impulsem k vytvoření této práce bylo streamování 360° živého pohledu na Liberec.

Autor chce na této problematice dále pracovat a bude se snažit vylepšit některé části, které uváděl v textu této práce. Jednou z možností zkvalitnění je možnost volného rozmístění kamer a živého stitchingu obrazu. Další navrhované vylepšení se týká snížení množství přenášených dat, kde příjem streamu videa by byl odebírán jen pouze z kamer v aktuálním náhledu. Výhodným vylepšením může v budoucnosti také být vytvoření serverové aplikace na platformu Windows.

8.4 Zhodnocení

V průběhu této práce musel autor řešit stále se objevující problémy jak hardwarové, tak i softwarové, a tudíž výsledná aplikace při svém vývoje procházela řadou razantních změn. Stanovené cíle práce byly splněny a v rámci získaných poznatků byly navrženy možnosti vylepšení. Všechny body zadání této práce byly splněny a je vytvořeno funkční řešení s jednoduchou implementací. Aplikace prošla testováním na uvedené hardwarové konfiguraci s očekávaným funkčním výsledkem.

Literatura

[1] Makzan: Programujeme hry v HTML5, Computer Press, 2012, EAN:9788025137314

[2] Zakas, Z. N.: JavaScript pro webové vývojáře, Computer Press, 2009, EAN:9788025125090

[3] COMPUPHASE,. Panoramic images. Panoramic images [online]. 2000, (7) [cit. 2015-09-09]. Dostupné z: `http://www.compuphase.com/panorama.pdf`

[4] DIRKSEN, Jos. Learning Three The JavaScript 3D Library for WebGL. New Edition. Birmingham: Packt Publishing, 2013. ISBN 17-821-6628-9.

[5] MATSUDA, Kouchi a Rodger LEA. WebGL programming guide: interactive 3D graphics programming with WebGL. xxv, 516 pages. Always learning. ISBN 03-219-0292-0.

[6] PARISI, Tony. Programming 3D applications with HTML5 and WebGL. First edition. xv, 384 pages. ISBN 978-144-9362-966.

[7] AUSTERBERRY, David. The technology of video and audio streaming. 2nd ed. Burlington, MA: Focal Press, 2004. ISBN 02-408-0580-1.

[8] SIMPSON, Wes. Video over IP: a practical guide to technology and applications. 2nd ed. Boston: Focal Press, c2006, xix, 493 p. ISBN 978-024-0805-573.

[9] ThreeJS Docs. MRDOOB. ThreeJS Docs [online]. [cit. 2015-09-09]. Dostupné z: `http://threejs.org/docs/`

[10] OpenCV [online]. 2015 [cit. 2015-09-09]. Dostupné z: `http://opencv.org/`

[11] WEITZ, Allan. The Tools and Techniques of Panoramic Photography [online]. (1) [cit. 2015-09-09]. Dostupné z: `http://www.bhphotovideo.com/explora/photography/tips-and-solutions/tools-and-techniques-panoramic-photography`

[12] CLARK, Richard. Beginning HTML5 and CSS3: the Web evolved : next generation Web standards. New York: Distributed to the book trade worldwide by Springer Science+Business Media, c2012, xx, 600 p. Expert's voice in Web development. ISBN 1430228741.

[13] HAGGAR, Peter. Practical Java: programming language guide. Reading, Mass.: Addison-Wesley, c2000, xxx, 279 p. ISBN 0201616467-.

[14] SOUKUP, Roman. Škola digitální fotografie. 1. vyd. Praha: Grada, 2006, 146 s., 28 s. barev. obr. příl. Průvodce (Grada). ISBN 80-247-1077-3.