TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky a mezioborových studií

Studijní program: B2612 / Elektrotechnika a informatika
Studijní obor: 1802R022 / Informatika a logistika

Návrh interaktivního softwaru pro vzdálené ovládání mobilních robotů

A Design of Interactive Software for Remote Control of Mobile Robots

Bakalářská práce

Autor: **Tomáš Zajíc**
Vedoucí práce: Ing. Miroslav Holada, Ph.D.

V Liberci 17. 5. 2013

TECHNICKÁ UNIVERZITA V LIBERCI

Fakulta mechatroniky, informatiky a mezioborových studií

Ústav informačních technologií a elektroniky Akademický rok: 2012/2013

Zadání bakalářské práce

Jméno a příjmení:	Tomáš Zajíc
Osobní číslo:	M10000052
Studijní program:	B2612 / Elektrotechnika a informatika
Studijní obor:	1802R022 / Informatika a logistika
Název tématu:	Návrh interaktivního softwaru pro vzdálené ovládání mobilních robotů
Vedoucí učitel práce:	Ing. Miroslav Holada, Ph.D.

Zásady pro vypracování:

1. Seznamte se s problematikou projektu průzkumná miniponorka na pracovišti školitele.
2. Seznamte se s možnostmi vzdáleného ovládání mobilních robotů. Zaměřte se na přenos obrazu z kamer robota, informace z dalších senzorů a také zobrazení a nastavení jednotlivých stavů řídícího mikrokontroléru.
3. Naprogramujte kompaktní graficky orientovaný software pro vzdálené ovládání miniponorky.
4. Realizovaný software prakticky otestujte a diskutujte možnosti spuštěni na různých platformách.

V Liberci dne 6. 10. 2012 vedoucí učitel projektu (podpis) .....

Prohlášení

Byl jsem seznámen s tím, že na mou bakalářskou 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 bakalářské práce pro vnitřní potřebu TUL.

Užiji-li bakalářskou 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.

Bakalářskou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím bakalářské práce.

Datum

Podpis

Poděkování

Na tomto místě bych rád poděkoval panu Ing. Miroslavu Holadovi, Ph.D., za odborné vedení bakalářské práce, za poskytnuté informace a cenné rady. Dále bych rád poděkoval spoluautorům miniponorky, pánům Miroslavu Roubíčkovi, Martinu Peklákovi, Jakubu Štěpánkovi a Davidu Svobodovi díky kterým jsem se mohl podílet na rozvíjení projektu miniponorky.

Abstrakt

Návrh interaktivního softwaru pro vzdálené ovládání mobilních robotů

Hlavním cílem této práce bylo vytvoření programu pro komunikaci s řídícím softwarem uvnitř miniponorky. Tento program měl být graficky a uživatelsky přívětivý s co nejjednodušším ovládáním. Avšak nejdříve bylo nutné seznámit se s hlavní koncepcí aktuálních programů pro miniponorku. Vyzkoušet tyto programy na počítači, zjistit jejich nedostatky a vytvořit program pro komunikaci s miniponorkou, který tyto nedostatky odstraní. Program, zabývající se přenosem videa z webkamery, bylo potřeba prozkoumat podrobněji a vylepšit práci s obrazem.

Tato práce pojednává nejdříve o robotech a možnostech jejich vzdáleného ovládání. Zaměřuje se na přenos obrazu z kamer robota, informace ze senzorů, zobrazení a nastavení jednotlivých stavů řídícího mikrokontroléru. V hlavní části se zabývá kompaktně graficky orientovaným softwarem pro vzdálené ovládání miniponorky, který byl hlavním cílem této práce, a na závěr praktickým otestováním samotného softwaru.

Požadovaný software se skládá ze dvou podprogramů. Jeden program byl tvořen v rámci této bakalářské práce - měl komunikovat s miniponorkou a ovládat ji. Druhý byl tvořen v rámci jiné bakalářské práce a zajišťoval přenos obrazu z webkamery v miniponorce. Finálním cílem této práce bylo oba tyto podprogramy spojit do jednoho a vytvořit tak software, který bude schopen ovládat miniponorku a zároveň přijímat data ze senzorů integrovaných v miniponorce, obraz nebo záznamy videa z miniponorky.

Celý software byl sepsán v jazyce C++ za pomoci vývojového prostředí Microsoft Visual Studio 2008.

Klíčová slova: miniponorka, klient, server

Abstract

A Design of Interactive Software for Remote Control of Mobile Robots

The main objective of this work was to create a program for communication with control software inside submersibles. The program should be graphically and user friendly with easy navigation. However, at first it was necessary to acquaint with the main concepts of the current programs for submersibles. Try these programs on computer, find all their weaknesses and create a program for communication with submersibles without all these weaknesses. The program for transmission of video out of the submersibles was necessary to examine and improve the image.

Firstly, this work discusses about robots and possibilities their remote control. It focuses on the transmission of images out of cameras of robot, information from sensors, display and settings each status of the control microcontroller. The main part deals with compact graphically oriented software for remote control of submersibles, which was the main objective of this work and at the end deals with practical test of the software.

The required software is composed of two sub-programs. The first one was created in this work and it should communicate with submersibles and control it. The second one was created in another work and it ensures the transmission of image out of webcam in submersibles. Final objective of this work was to combine these programs and to create software, which will be able to control the submersibles and at the same time accept data from sensors integrated in the submersibles, image or record of video out of submersibles.

All software was written in C++ language by using interactive development environment Microsoft Visual Studio 2008.

Keywords: Submersibles, client, server

Obsah

Seznam obrázků

Seznam ukázek

Seznam použitých zkratek

API	Application Programming Interface – rozhraní pro programování aplikací
DSN	Deep Space Network – síť hlubokého vesmíru
HDD	Hard Disc Drive – pevný disk
IP	Internet Protocol – internetový protokol
LAN	Local Area Network – lokální síť
LED	Light-Emitting Diode – dioda emitující světlo
MSL	Mars Science Laboratory – Marsovská vědecká laboratoř
NASA	National Aeronautics and Space Administration - národní úřad pro letectví a kosmonautiku
PC	Personal Computer – osobní počítač
PLC	Programmable Logic Controller – programovatelný logický automat
USB	Universal Serial Bus – univerzální sériová sběrnice

Úvod

Celá tato bakalářská práce se týká průzkumné miniponorky. Průzkumná miniponorka je rozsáhlý projekt, jehož cílem je vytvoření mobilního robota, ovládaného ze souše jakýmkoli PC s ovládacím softwarem, schopného pohybovat se na vodní hladině nebo pod ní v libovolném směru. Miniponorka by měla být schopná reagovat na podněty uživatele, přijímat tak příkazy, které následně vykoná a odesílat informace o tom, v jakém stavu se nachází, fotografie a videozáznam z webkamery spolu s dalšími daty z integrovaných senzorů jako jsou stav baterií, teplota nebo úhel náklonu miniponorky. Tento mobilní robot by mohl být jednou využíván při průzkumu dna rybníků a nádrží a jejich mapování.

V současné době se miniponorky využívají v mnoho odvětvích a pro různé činnosti. Velice užitečné jsou u policejních útvarů pro vyšetřování. V mořích pro průzkum dna, potopených lodí nebo nalezených starých korábů. Miniponorek je využíváno tam, kde je to pro člověka nebezpečné nebo tam, kam je velice obtížné se dostat z důvodu malých nebo neznámých prostor, například při zkoumání zatopených jeskyní.

Projekt je tvořen již několik let bakalářskými a diplomovými pracemi a tak vzniká stále schopnější robot a vše se blíží k finálnímu stavu. Tato práce se zabývá softwarem na ovládání robota přes síťové rozhraní. Software by měl mít všechny náležitosti vhodné pro ovládání miniponorky a získávání dat a informací nazpět z miniponorky. Vše by mělo mít jasné grafické rozhraní.

Při tvorbě softwaru bylo využito vývojové prostředí Microsoft Visual Studio 2008 a testování programů probíhalo pod systémem Windows 7 Professional 64bit CZ od firmy Microsoft.

1 Základní informace o projektu průzkumná miniponorka

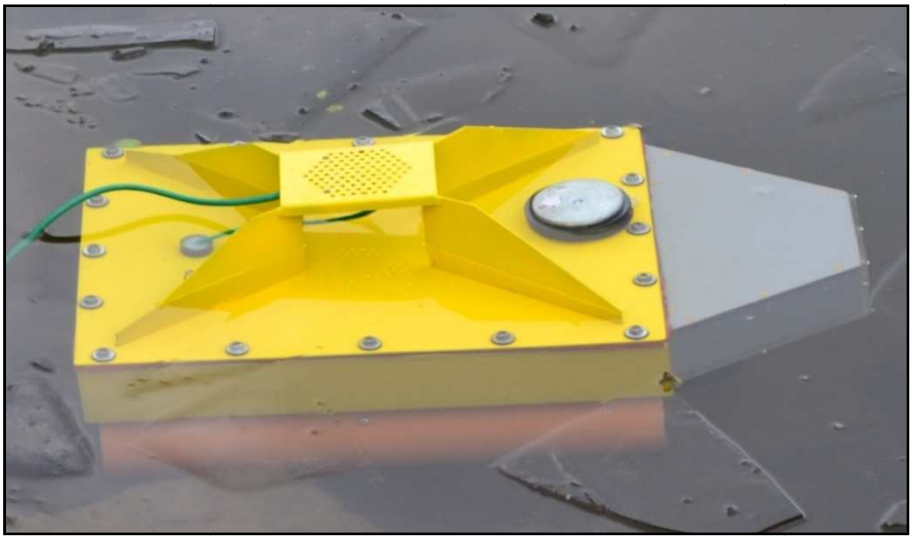
Průzkumná miniponorka je plavidlo, které je schopné se pohybovat jak na hladině vody, tak i pod vodou. Miniponorka má rozměry asi 70x50x40 cm. Jedná se o robotické zařízení, které je používáno pro zkoumání čistších vod a je ovládáno přes počítač umístěný na břehu vody.

Miniponorku je možné rozdělit na tři základní části:

Hlavní část tvoří žlutý dutý kvádr, v němž se nachází veškerá elektronika důležitá pro ovládání miniponorky. Uvnitř je umístěna základní deska z notebooku, mikrokontrolér Picaxe, balastní komory, čerpadla a hlavně také 2 baterie, které slouží k napájení celé miniponorky v provozu.

Druhou část tvoří komora připevněná k hlavní části vpředu. Tato komora má tvar prostorového rovnoběžníku, který je vpředu vybaven plexisklem. Za plexisklem je umístěná klasická notebooková webkamera, která je používána ke snímání prostředí před ponorkou v průběhu ponoru.

Jako třetí část je možné označit dvě trubky, které jsou umístěny vespod miniponorky. V těchto trubkách jsou připevněny motory s vrtulemi, které svým roztočením uvádí celou miniponorku do pohybu. Vrtule použité jako pohon jsou slabší a užší a tak nemají příliš vysoký hnací výkon.



Obr. 1: Průzkumná miniponorka

Na vrchu hlavní části miniponorky se nachází úchyt, díky kterému se s ní velice dobře manipuluje v případě přenosu nebo přemístění. Dále je zde možné nalézt také ventilek, který umožňuje před ponorem zvýšit tlak uvnitř plavidla a tak se ujistit, že celý model je dobře utěsněn a neobsahuje žádné trhliny, kterými by mohla do ponorky natéct voda a tím dojít k nevratnému poškození zařízení miniponorky. Ventilek je galuskový a pomocí redukce lze použít jakýkoliv kompresor, který vytvoří potřebný tlak uvnitř miniponorky.

Jak už je uvedeno výše, pod hlavní částí plavidla se nachází dvě trubky, ve kterých jsou umístěny motory a jejich vrtule, které uvádí ponorku do pohybu. Trubky, jež jsou kolem vrtulí, zvyšují jejich výkon a chrání je před vnějším nebezpečím v podobě kamenů či větších nečistot pod vodou. Mezi drobné problémy takovéhoto sestavení trubek, motorů a vrtulí je možnost vtékaní nečistot do trubky a její ucpání. Na koncích trubek byly mřížky, které právě vtékaní nečistot bránily, ale vznikl tím jiný problém. Mřížky se rychle zanesly a efektivita pohonu se snížila na minimum.

Na komoře umístěné v přední části hlavního modulu je možné vidět několik LED diod. Na spodním okraji je instalováno šest LED diod pro osvětlení. Více LED diod najdeme níže. Ty jsou určeny jako hlavní světlo. Bohužel i při použití velkého množství světelných zdrojů, není intenzita světla dostačující a nepřináší tedy příliš velké zlepšení obrazu, který snímá kamera. V současné době je většina LED diod mimo provoz, protože s největší pravděpodobností došlo k prosáknutí vody do prostoru, kde jsou LED diody umístěny a tak došlo ke zkratu.

V tomto roce byly do ponorky přidány nové komponenty a senzory, které by měly vylepšit její ovládání, pohyb a pomoci tak vytvořit komplexnějšího robota, který se bude schopen vypořádat se vším, co ho při zkoumání dna rybníků čeká. Hlavním přínosem je integrace dvou balastních komor místo jedné. Díky tomuto vylepšení bude ponorka schopna se pod vodou naklánět a tak bude snazší pohyb v horizontální a zároveň vertikální poloze. Obě komory se budou moci napouštět a vypouštět nezávisle na sobě a tím bude jen na uživateli, jak bude tato novinka využita. Dále byl do ponorky integrován akcelerometr, díky němuž bude možné sledovat, jak je miniponorka nakloněna, přesně o kolik stupňů a podle toho bude možné reagovat a napouštět nebo vypouštět jednu či druhou balastní komoru.

1.1 Historie průzkumné miniponorky

Projekt průzkumná miniponorka vznikl díky Ing. Miroslavu Holadovi, Ph.D. Cílem tohoto dlouhodobého projektu je prostřednictvím bakalářských nebo diplomových prací seznámit studenty s vývojem mobilních robotů a jejich ovládáním. Jelikož robotů, které se pohybují po zemi, hledají cestu z bludiště nebo manipulují svým ramenem, je na všech školách mnoho, tento projekt se zabývá robotem, který se bude pohybovat pod vodou. Projekt průzkumná miniponorka byl vytvořen v akademickém roce 2009/2010 a v následující části bude popsáno, jak je postupně vyvíjen.

Práce a projekty v akademickém roce 2009/2010:

Jiroušek Petr:	Průzkumná miniponorka – návrh a realizace ovládacího softwaru
Oliva Jan:	Průzkumná miniponorka – návrh a realizace kamerového systému
Peklák Martin:	Průzkumná miniponorka – návrh a realizace systému napájení
Roubíček Miroslav:	Průzkumná miniponorka – návrh a realizace elektropohonu

Práce a projekty v akademickém roce 2010/2011:

Peklák Martin:	Návrh elektronické výbavy experimentální dálkově řízené miniponorky
Roubíček Miroslav:	Návrh konstrukce experimentální dálkově řízené průzkumné miniponorky
Vričan Ondřej:	3D Vizualizace průzkumné miniponorky

Práce a projekty v akademickém roce 2011/2012:

Peklák Martin:	Projekt miniponorka – možnosti statického a dynamického
Svoboda David:	Projekt miniponorka – nízkoúrovňový řídící software
Štěpánek Jakub:	Elektronická výbava mobilního robota
Zajíc Tomáš:	Projekt miniponorka – grafický ovládací terminál

Práce a projekty v akademickém roce 2012/2013:

Svoboda David:	Návrh řídícího systému pro ovládání mobilních robotů
Peklák Martin:	Systém vyvažování průzkumné miniponorky
Zajíc Tomáš:	Návrh interaktivního softwaru pro vzdálené ovládání mobilních robotů

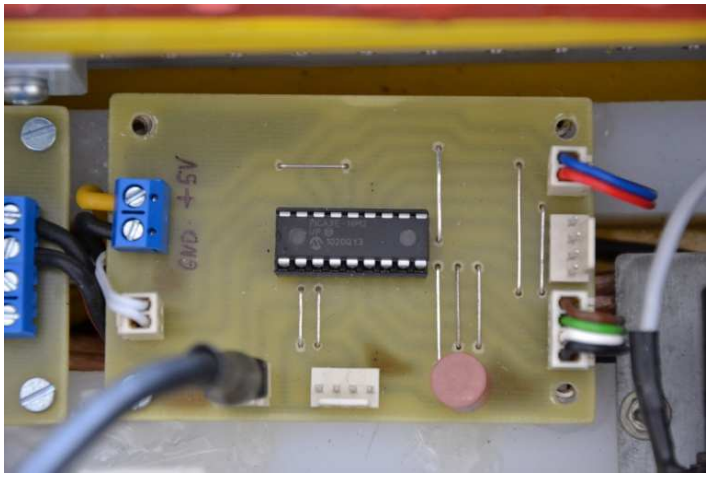
1.2 Řízení a ovládání miniponorky

Základní prvek, který ovládá každého robota, je počítač. Může se jednat o klasické PC, notebook, ale také o PLC nebo starší mikrokontroléry, jejichž programování je velice jednoduché a logické, jak už vyplývá z názvu PLC. Počítač a notebook jsou velice rychlé a není obtížné využití v praxi. Problém je jejich velikost, PLC a mikrokontroléry jsou menší. Miniponorka je však dostatečně prostorově řešena a tak nebyl problém integrovat základní desku z notebooku.

V některých případech je velice výhodné zkombinovat několik zařízení. V projektu miniponorka bylo využito právě této možnosti, protože velikost průzkumné miniponorky tuto variantu umožňovala. Do miniponorky byla posazena základní deska z notebooku od firmy HP. Ta měla pouze nejzákladnější vybavení, jako je HDD, USB porty, integrovanou síťovou kartu a podobně. Na HDD byl nainstalován systém Windows XP od firmy Microsoft a do systému byly nahrány ovladače pro zařízení uvnitř miniponorky. Dále zde byly nainstalovány programy nutné pro spuštění komunikace, jako je server, webkamera, a další podpůrný software. S miniponorkou se komunikovalo právě díky integrované síťové kartě a rozhraní LAN (RJ-45).

Uvnitř miniponorky se dále nacházel mikrokontrolér PICAXE 18M2. Tento mikrokontrolér je často používán v projektech pro svou velikost a vhodné funkce. Podporuje až 16 vstupů/výstupů, rozšířené funkce, jako jsou konfigurovatelné input/output piny, a paralelní úkoly. [3]

Kombinací notebooku a mikrokontroléru byl vytvořen komplexnější robot, než kdyby byl použit pouze mikrokontrolér nebo pouze notebook.



Obr. 2: PICAXE 18M2

2 Seznámení se s možnostmi ovládání robotů

V této části je popsáno, jakým způsobem jsou ovládáni podobní roboti a jakým způsobem fungují. V případě programování robotů je nutné vědět, jak se bude pohybovat, a jak přesně bude fungovat, aby bylo možné program co nejlépe napsat a vyladit.

2.1 Základní dělení robotů

První roboti byli ovládáni pomocí pevného neboli statického programu, který nebyl schopen nijak reagovat na aktuální podmínky a neobsahoval žádné senzory nebo čidla. Naopak dnešní roboti se vyrábějí s různými čidly a senzory a tak jsou schopni reagovat na okolní podmínky a tím jsou mnohem schopnější než jejich předchůdci. Dále je možné roboty rozdělit podle jejich pohybu. Zaprvé mohou být roboti stacionární. Stacionární roboti nejsou schopni pohybu. Do této skupiny robotů patří průmyslové manipulátory, které plní svou úlohu a nepotřebují se přemisťovat z místa na místo. Příklad takového robota vidíme na obrázku níže. Úkolem tohoto robota je pouze pohybovat ramenem a tak není třeba pohybu z místa na místo. [4]

Obr. 3: Ilustrační obrázek robotického ramene [5]

Z toho vyplývá, že druhým typem robotů jsou mobilní neboli pohybliví roboti. Tito roboti se během své práce mohou pohybovat. Takovým zajímavým příkladem jsou určitě roboti na Marsu. Cílem těchto robotů je zkoumání povrchu Marsu a proto se bez pohybu neobejdou. Dalším příkladem mobilního robota je určitě tento projekt.

Průzkumná miniponorka, jak už z názvu vyplývá, by měla být používána pro zkoumání dna vodních ploch a nádrží a tudíž se bez pohybu také neobejde. Rozdíl mezi těmito dvěma roboty je takový, že robot na Marsu se pohybuje po souši za úplně jiných gravitačních podmínek, než jsou na Zemi a miniponorka se pohybuje na hladině vody nebo pod vodou, kde je nutné vyvažování pomocí balastních komor, aby nedošlo k samovolnému vynoření a tím přerušení průzkumu dna. [4]

Další možností, jak rozlišovat roboty, je podle způsobu jejich pohybu. Tento pohyb může být kloubový úhlový a kloubový otočný nebo teleskopický a posuvný. Roboty je také možné rozdělit do skupin podle toho, zda jsou řízeni, ovládáni, regulováni, autonomní nebo inteligentní. [4]

2.2 Senzory a jejich rozdělení

Senzory jsou obecně brány jako zdroje informací pro řídící systém. Senzory fungují na jednoduchém principu. Měří nějakou fyzikální veličinu, kterou následně převádí na signál, který odešlou a tento signál je zpracován v měřících a řídících systémech. Nejčastěji se jedná o elektrický signál. Typy senzorů můžeme měřit podle měřené veličiny, media sloužícího k přenosu signálu (elektrické, pneumatické,…), fyzikálního principu, druhu styku s prostředím (dotykové, bezdotykové) nebo podle stupně integrace. Jako příklady senzorů je možno uvést snímač teploty, snímač naklonění, snímač výšky hladiny a další. Tyto senzory se nachází i v průzkumné miniponorce. [6]

2.3 Základní principy ovládání robotů

V předchozích kapitolách byly popsány základní principy robotů, jakým způsobem se liší a jak fungují. Všechny tyto informace jsou velice důležité, protože dříve než je možné robota ovládat, je nutné znát, jak funguje a jaké má vlastnosti. Ovládání robotů může probíhat nespočtem možných způsobů. Základní myšlenka ovládání robotů není jen o tom dát mu příkaz, který má vykonat, ale také je dobré získat pomocí robota nějaké informace z okolí, ve kterém se právě nachází. K tomu slouží právě senzory, kamery a další zařízení, které je díky programům možné ovládat a také posílat data z míst, kam by bylo pro člověka obtížné se dostat.

2.4 Přenos obrazu z kamer robota

Nejdříve je nutné se podívat na to, jak můžeme přenášet obraz z robota až k uživateli. Je mnoho různých způsobů, jak se vzdáleně k obrazu z robota dostat, a jak ho zpracovávat. Základní rozdíly jsou v zařízení, které je pro přenos využíváno nebo v samotném přenosu, který je pro získání dat využíván. Pří komunikaci s robotem lze využít přenosu dat přes kabel USB, pomocí rozhraní LAN, vzduchem přes wifi nebo využitím rádiových vln. V tomto projektu je prozatím využívána komunikace přes rozhraní LAN, ale do budoucna se očekává využití rádiových vln, které lze posílat i pod vodou a tak by mohla být realizována bezdrátová komunikace s miniponorkou.

2.4.1 Přenos obrazu přes telefon s Wi-Fi

Jednou z možností jak pracovat s obrazem robota je za použití mobilního telefonu, který podporuje wifi. Základem je mít nainstalovaný program, který bude schopen práce s obrazem a jeho odesláním. Jako příklad je možné uvést program WebCamera Plus. Obraz lze díky tomuto programu posílat po všech dostupných kanálech včetně wifi. Při použití těchto zařízení za nastavení nižšího rozlišení (400x240 nebo 320x240) dochází k velmi malému zpoždění (100-150ms). [7]

Obr. 4: Robot připojený k telefonu, který je ovládán pomocí Wi-Fi [7]

V projektu miniponorka se také plánuje ovládání a přenos informací vzduchem, ale bohužel tento způsob není možný. Důvodem je, že signál wifi se špatně šíří pod vodou a tak by nebylo možné se s miniponorkou pod hladinou nijak spojit.

2.4.2 Komunikace s roboty na vysokou vzdálenost

Roboti, které je třeba ovládat nebo z nich získávat důležité informace, nemusí být tak blízko, jak by se dalo čekat. Americká společnost NASA poslala již několik robotů na 156 milionů kilometrů vzdálenou planetu Mars. Roboti jsou na takovou vzdálenost ovládáni pomocí sítě hlubokého vesmíru DSN. Obsahují dvě dvoumegapixelové kamery, které získávají obrázky z Marsu a dále je posílají přes DSN do Marsovské vědecké laboratoře MSL, která je umístěna na Zemi. Robot na Marsu dále obsahuje chemickou laboratoř, která zpracovává prvky a chemické sloučeniny na Marsu a vše opět posílá zpět na Zemi přes DSN a tak je možné díky tomuto robotovi získávat informace o prostředí z míst, kam by bylo obtížné se dostat a to díky možnosti komunikovat přes vesmírnou síť na vzdálenost 156 milionů kilometrů. [8], [9]

Obr. 5: Ilustrační obrázek robota na Marsu [10]

Na obrázku 5 vidíme robota na Marsu. Program, který ho ovládá je napsán tak, aby robot popojížděl po povrchu Marsu a sám vyhledával objekty ke zkoumání. Jak je na obrázku vidět, tak je vybaven robotickou rukou s již zmiňovanou chemickou laboratoří a nad robotem je umístěna kamera, která vše zachycuje a poté je vše odesláno. Program, který robota ovládá, musí být co nejpřesnější, protože dodatečně už je obtížné ho upravit na tak vysokou vzdálenost. Jediná chyba v programu by mohla znamenat obrovské škody a proto je nutné vývoj ovládacího programu nezanedbat a udělat co nejpřesněji vzhledem k podmínkám ve kterých se bude nacházet.

Největším problémem komunikace s tímto robotem je zpoždění, které vzdálenost, na kterou komunikuje, přináší. Toto zpoždění je minimálně 8 minut a může být i vyšší v závislosti na poloze Marsu vůči zemi. Komunikace je pomocí rádiových vln, které jsou neustále vysílány na Zemi. Toto zpoždění má obrovský vliv na ovládání, protože zde na Zemi vidíme, kde se robot nacházel minimálně před osmi minutami. Z tohoto problému vychází rizika taková, že robot se může přiblížit k útesu a už není možné ho včas zastavit. Proto program, který je v robotovi nainstalován, musí být schopný reagovat na povrchové podmínky a včas se všem rizikům vyhnout, anebo musí být dostatečně pomalý, aby bylo možné ho včas zastavit a poslat jiným směrem. [8], [9]

Bakalářská práce je v tomto směru mnohem jednodušší, protože je uživatel s miniponorkou neustále v kontaktu a ví co se právě děje uvnitř i vně s minimálním zpožděním a tak může jako uživatel okamžitě reagovat na problémy a chyby, které by mohly nastat. Další výhodou této práce je, že je přesně známo prostředí, kde se bude miniponorka nacházet a nemusí být řešena jiná gravitace, jako je to u robotů na Marsu. Nevýhodou miniponorky je špatná viditelnost pod vodou, kde se těžko docílí perfektní viditelnosti, jako je tomu na Marsu.

2.4.3 Komunikace s roboty přes rozhraní LAN

LAN neboli lokální síť je vlastně počítačová síť na malém území. Výhodou je přenosová rychlost, která dosahuje řádově až GB/s. Nejrozšířenější technologie jsou Ethernet a Wi-Fi. [11]

Projekt miniponorka využívá právě síťové komunikace LAN a konkrétně technologii Ethernet, protože Wi-Fi nemá pod vodou dostatečný dosah signálu. Toto rozhraní je velice rychlé a díky malému zpoždění při posílání dat a obrazu je vhodné právě pro komunikaci s miniponorkou. Lze využít Windows API funkcí, které tuto komunikaci mohou zprostředkovávat. Komunikace LAN je jedna z nejpoužívanějších komunikací s roboty.

Nevýhodou u této komunikace je kabel, který musí mít robot připojen a tak je například u miniponorky problém s odporem kabelu ve vodě a dochází k brzdění celého pohybu. Možným řešením tohoto problému je nadlehčení kabelu nad hladinu, například pomocí polystyrenu. Tímto zlepšením kabel není pod vodou a není tak velký odpor při pohybu.

3 Kompaktní graficky orientovaný software pro vzdálené ovládání miniponorky

Hlavním cílem této bakalářské práce bylo vytvoření kompaktního graficky orientovaného softwaru pro vzdálené ovládání miniponorky. To znamená napsat program, díky kterému bude moci uživatel ovládat miniponorku a bude přijímat video z webkamery, která se v miniponorce nachází. Vše mělo být vytvořeno tak, aby byl program co nejvíce uživatelsky příjemný, to znamená, aby byl přehledný a bylo jasné vše, co se v miniponorce děje, jak se pohybuje, a byly zobrazeny důležité parametry ze senzorů, které jsou v ponorce integrovány. Program by měl být jednoduchý a graficky přehledný, takže by nemělo být složité se v něm okamžitě orientovat. Vše by mělo být ovládáno tlačítky v programu nebo na klávesnici. Jelikož je v názvu bakalářské práce napsáno, že má být vytvořen interaktivní software, vyplývá z toho, že bude software mezi sebou nějakým způsobem komunikovat. Celý tento software komunikuje přes síťové rozhraní LAN a jedná se o komunikaci klient – server.

3.1 Komunikace klient-server

Jak už je psáno výše, v této práci se uplatňuje komunikace klient-server. Je to síťová architektura, která odděluje klienta, který je obvykle graficky uživatelský, a server. Klient a server spolu komunikují přes počítačovou síť. Z toho vyplývá, že tato komunikace musí obsahovat jak server, tak i klienta.

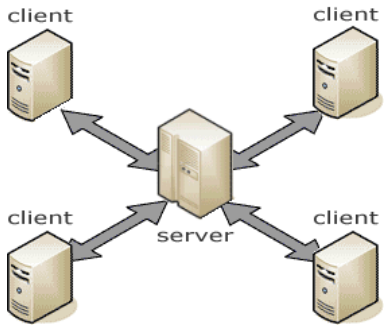
Klient-server realizuje komunikaci mezi dvěma počítačovými programy, kdy první program (klient) žádá o služby druhý program (server). Každý klient může posílat data jednomu nebo i několika připojeným serverům. Servery naopak mohou tyto žádosti zpracovávat a vracet požadovanou informaci zpět klientům. Klient-server tedy obsahuje pouze dvě části a to serverovou a klientskou. [12]

Základní charakteristika klienta:

- Klient je aktivní
- Posílá žádosti na server
- Je připojen k malému počtu serverů
- Čeká a dostává odpovědi
- Komunikuje přímo s koncovými uživateli díky grafickému rozhraní [12]

Základní charakteristika serveru:

- Server je pasivní
- Naslouchá na síti a reaguje na žádosti připojených klientů
- Po přijetí požadavku jej obslouží
- Může vzdáleně instalovat/odinstalovat aplikace a přenášet data ke klientům [12]



Obr. 6: Schéma fungování komunikace klient-server [13]

Hlavní výhodou komunikace klient-server je, že jsou úkoly rozděleny mezi různé počítače a tak je zde mnohem snadnější údržba. Díky tomuto je možné server nahradit, opravit, modernizovat nebo přemístit, aniž by to ovlivnilo funkci klientů. Další výhodou je, že jsou všechny údaje uloženy na serverech, které jsou mnohem bezpečnější než většina klientů. Servery mohou kontrolovat přístup a zdroje a z toho vyplývá, že přistupovat a měnit data mohou jen někteří oprávnění klienti. Mezi hlavní nevýhody patří problém s přetěžováním sítě. [12]

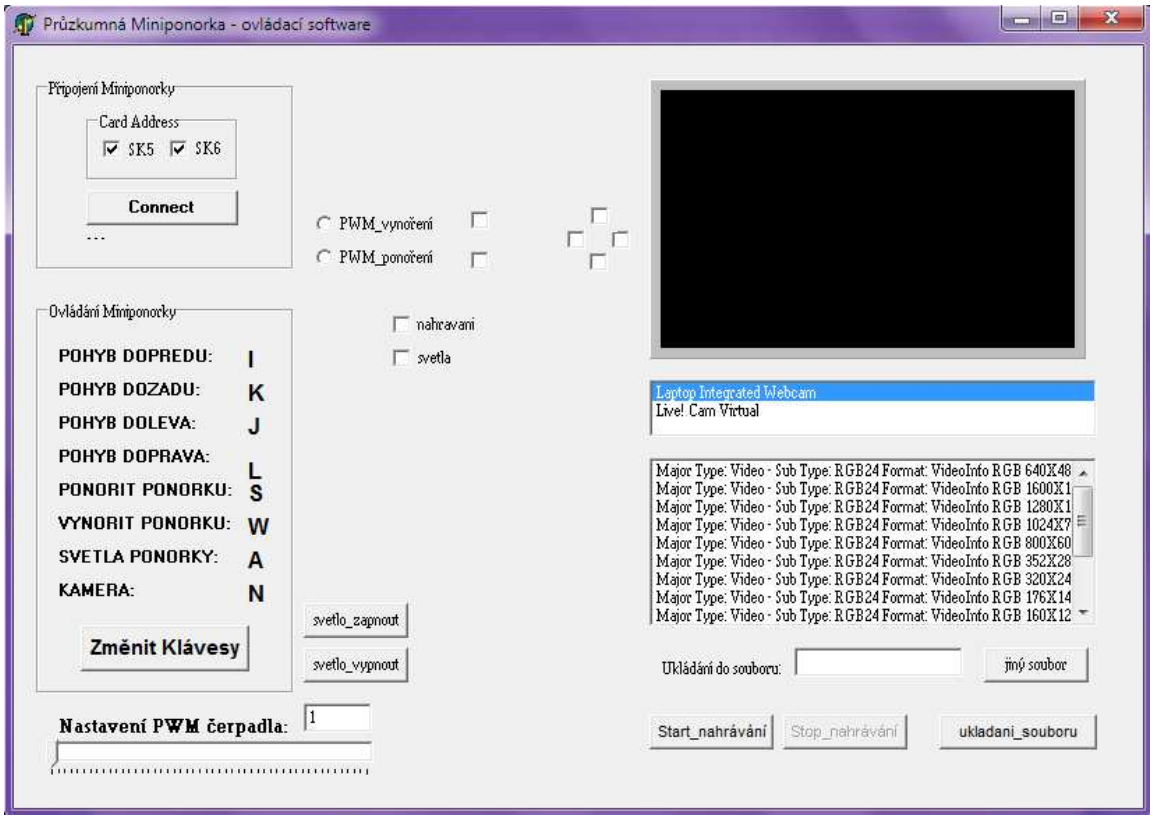
Výhody však převažují nad nevýhodami a tak v rámci projektu průzkumné miniponorky byla právě komunikace klient-server použita. Je využita zvláště z důvodu, že v miniponorce je integrována základní deska z notebooku se síťovým připojením a tak je možné komunikovat pomocí rozhraní LAN a vše nastavit přes vzdálenou plochu. V miniponorce probíhá komunikace přesně ve smyslu rozhraní klient-server. Uvnitř miniponorky jsou na pevném disku nainstalovány programy (servery) a k těmto serverům se připojují programy, které mají uživatelé na břehu (klienti). V ponorce se nachází servery dva. První server, který je v miniponorce se stará o přenos videa, to znamená, zprostředkování odesílání dat, která obsahují videozáznam dění před miniponorkou, což zaznamená webkamera, která je umístěna v přední části plavidla. Druhý server v miniponorce zprostředkovává zasílání dat ze senzorů v miniponorce a dále ovládání samotné miniponorky.

3.2 Historie programů pro průzkumnou miniponorku a jejich nedostatky

V této části bakalářské práce budou zmíněny programy, které byly již napsány dříve, a bude zde popsáno, jak fungovaly, a jaké byly důvody jejich dalšího vylepšování. Každá část bude doplněna obrázkem, který bude reprezentovat daný program. První software byl napsán v jazyce pascal, zbylé programy byly psány už jen v jazyce C++.

3.2.1 Původní program psaný v jazyce pascal

Jak už bylo zmíněno, první software, který byl napsán v akademickém roce 2009/2010, byl napsán v jazyce pascal ve vývojovém prostředí delphi. Tento program byl velice jednoduchý a přehledný. Uživatel se připojil k serveru v miniponorce a přijímal od něho video z webkamery, která nahrávala vše co se děje před plavidlem. Ovládání probíhalo pomocí kláves nebo pomocí tlačítek, které byly v programu připraveny.



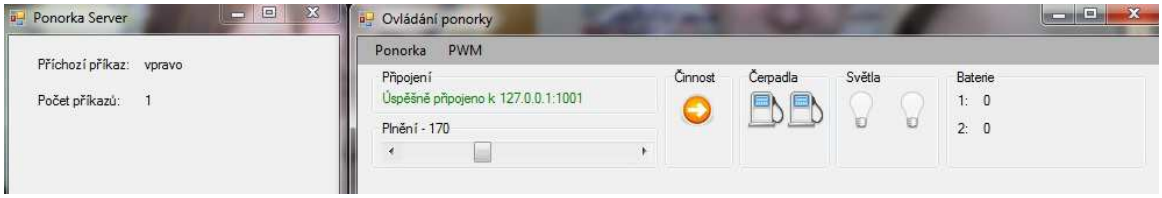
Obr. 7: Náhled programu napsaného v jazyce pascal a vývojovém prostředí delphi

Nevýhodou tohoto programu je právě jazyk, ve kterém je psaný. Tento jazyk je vytvořen speciálně pro výukové potřeby a tak není nejvhodnější pro psaní programů pro roboty. Dalším nedostatkem jsou tlačítka, která nejsou nejlépe řešená, protože použité

„CheckBoxy“ jsou pro uživatele nepřehledné a nejasné a bylo by lepší přidat nějaký obrázek nebo text který by oznamoval aktuální pohyb nebo stav. Velkým problémem tohoto programu byla nedostatečná stabilita, protože nefungovalo vše jak má a tak by v praxi mohlo docházet k výpadkům, které by mohly znamenat ztrátu kontroly nad miniponorkou. Ale i přes tyto chyby a nedostatky byl tento program pěkně sestaven a cílem tedy bylo vytvořit podobný program v jazyce C++.

3.2.2 První komunikační program do miniponorky psaný v jazyce C++

V loňském roce byl v rámci bakalářského projektu napsán software v jazyce C++, který byl schopný komunikovat na bázi klient-server. Tento program však byl pouze jednoduchý a nebyl připraven pro další vylepšování. Funkčnost softwaru byla jen ovládací, to znamená, že klient pouze odesílal data serveru v ponorce a ten vykonal žádosti. Komunikace probíhala pomocí Windows socketu (winsock), který zajišťoval síťovou komunikaci. Přenos videa byl přes tuto komponentu nejistý a tak nebylo možné v tomto programu dále pokračovat.



Obr. 8: Názorný obrázek předchozí verze software, který ovládal miniponorku

Grafickou úpravu tohoto programu jsem dělal osobně jako svůj bakalářský projekt. Na levé straně obrázku je vidět server, který přijímá žádosti od klienta na pravé straně obrázku. Na serveru vidíme text, který značí, jakým směrem se ponorka pohybuje nebo jaký úkon právě vykonává. V klientovi také vidíme vše co se v ponorce děje a vše je zde za pomocí obrázku odlišené. Pokud se ponorka pohybovala, naznačovala to šipka směřující stejným směrem. Pokud čerpadla napouštěla nebo vypouštěla, znázornilo se to na obrazcích v klientovi. Stejným způsobem fungovala i světla.

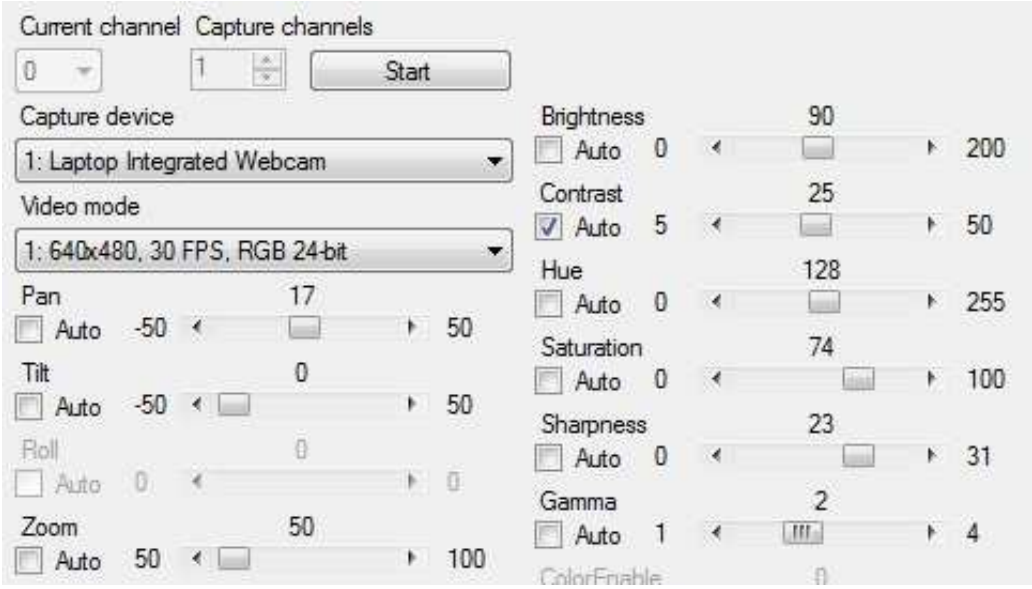
3.3 Program pro přenos obrazu z webkamery

V následující části této bakalářské práce je popsán program pro přenos obrazu z webkamery v miniponorce. Cílem bylo prozkoumat program pro posílání videa

z webkamery v reálném čase a pokusit se v něm něco upravit přidat nebo vylepšit. Nejdříve bylo nutné věnovat se úpravě samotného videa a poté bylo nutno přidat možnost ovládání a otestovat možné zobrazení informací o pohybu v obraze z webkamery.

3.3.1 Testování programu pro přenos obrazu z webkamery

V první části této bakalářské práce bylo nutné prostudovat aktuální program na přenos obrazu z webkamery, která se nachází v miniponorce. Tento program byl psán také v jazyce C++ a umožňoval uživateli používat mnoho funkcí práce s videem a obrazem. Software na přenos obrazu byl pouze serverový program, který by mohl být ovládán přes vzdálenou plochu. Byl schopný získat obraz z webkamery a různě ho transformovat. K obrazu z webkamery byla přidána i funkce „Zoom“, která umožňovala přiblížení na určitou část videa.



Obr. 9: Názorná ukázka možnoností nastavení obrazu z webkamery

V nastavení po spuštění programu se automaticky nalezne dostupné video zařízení a uživatel jej poté může vybrat i s volbou rozlišení obrazu, které toto video zařízení nabízí. Po výběru zařízení a spuštění videa v programu naskočí několik dalších nastavení. Mezi hlavními možnostmi nastavení obrazu je již zmiňovaný zoom neboli přiblížení obrazu, dále možnost nastavení kontrastu, světlosti a vlastně všech základních úprav obrazu, které by mohly zlepšit data získaná při zkoumání dna vodních ploch. Samozřejmě, čím nižší kvalita a rozlišení videa, tím je přenos videa z webkamery

až k uživateli plynulejší a rychlejší a naopak, čím bude uživatel náročnější a bude chtít co nejlepší obraz a záznamy dna, bude tento přenos obrazu pomalejší a méně plynulý.

3.3.2 Vylepšení programu pro přenos obrazu z webkamery

Po pečlivém prostudování programu pro přenos obrazu z webkamery bylo potřeba obraz, který se zobrazuje v programu nastavit tak, aby bylo možné ho zvětšit na celou obrazovku a do režimu celé obrazovky přidat údaje o tom co se v ponorce právě děje, jak se ponorka pohybuje a informace ze senzorů uvnitř miniponorky. Aby bylo možné obraz zobrazit na celou obrazovku, bylo potřeba prozkoumat parametry obrazu. V programu je původní prostor, kde je později video, vyplněn jednoduchým „Labelem“, a po spuštění přijímání videa se na tomto místě vytvoří „PaintBox“, do kterého se vše vykresluje. Pokud obraz zvětšíme na celou obrazovku a poté ho chceme opět vrátit zpět do původní podoby, je potřeba si uložit všechna aktuální data, která značí umístění „formu“ a umístění obrazu ve „formu“, aby je bylo při zpětné akci možno opět využít a tak nastavit „form“ na původní velikost. Pro uložení původních hodnot byly využity základní funkce komponent v jazyce C++ a to funkce „Top“, „Left“, „Width“ a „Height“. Důležité bylo také nastavit dokování „formu“ na „none“ a zrušit okraje „formu“, aby byl dojem celé obrazovky pro uživatele co nejpříjemnější. Aby obraz na celé obrazovce nepřekrýval obsah „formu“, který tam byl předtím, použije se funkce „bringtofront“, která vezme obraz a překryje vše, co bylo původně nad obrazem. Pro zobrazení obrazu na celou obrazovku je nutné použít dvojklik na původní obraz. Dvojklik vytvoří událost a ta provede již uvedené změny programu. Pro vrácení obrazu zpět do původní velikosti opět stačí pouze použít dvojklik nebo stisknout klávesu Esc. Po zmenšení na původní velikost je nutné opět nastavit okraje „formu“ na „sizeable“.

Po nastavení zvětšování a zmenšování obrazu se tato práce posunula k druhému kroku a to vytvoření mini-ovládání pro miniponorku. Základem bylo připravit činnosti, které jsou po ponorce vyžadovány a těm nastavit nějaké události. Vše bylo možné ovládat pomocí tlačítek na klávesnici, které po stisku vykonaly svou činnost, a program byl doplněn o grafické zobrazení dané činnosti. Aby bylo možné miniponorku ovládat, bylo nutné stisknout tlačítko „Start“ v programu, a tím se odeslaly základní data ze senzorů v ponorce, které zobrazovaly teplotu, stav baterií a stav jednotlivých komponent uvnitř miniponorky. Stisknutím tlačítka „Start“ se vytvořila událost, která zobrazila pomocí

funkce „visible->true“ další tlačítko „Plout“. Stisknutím tlačítka „Plout“ bylo již možné ovládat miniponorku tlačítky na klávesnici.

Obr. 10: Náhled na program pro ovládání miniponorky v serverovém softwaru

Na obrázku 10 vidíme náhled programu, který by mohl ovládat miniponorku. Jelikož byl tento program pouze testovací, tak nebyla odesílána ani přijímána žádná data. Cílem bylo naučit se ovládat funkce jazyka C++ a vývojové prostředí Visual Studio 2008. Na obrázku vidíme nastavené ovládání, které je stejné jako ve finální verzi programu této bakalářské práce. Ovládání tedy probíhá klasicky tlačítky „W“, „A“, „S“ a „D“, kdy po stisku jedné klávesy se ponorka vydá právě tím směrem. Jsou zde připravena i světla. Jelikož se v miniponorce nachází 2 světlomety, je vhodné nechat na uživateli, který ze světlometů využije. V tomto programu je osvětlení ovládáno tlačítky „G“ a „H“. Po stisku se světlo buď rozsvítí, nebo zhasne v závislosti na tom, zda právě svítí nebo ne. Na obrázku je vidět i signalizace nabití baterie, která by zobrazovala aktuální nabití baterií a případně by signalizovala nutnost jejich nabití. Dále jsou na obrázku vidět čerpadla, která znázorňují jejich činnost. Program je napsán tak, aby nebylo možné čerpadlo napouštět a zároveň vypouštět. Pokud je v činnosti napouštění, podbarví se znázorněné čerpadlo pro napouštění do oranžova. Čerpadla jsou ovládána tlačítky „V“ a „N“. Na pravé straně programu jsou zobrazeny jednoduché „Labely“, které uživateli oznamují informace ze senzorů, anebo o tom, co se v ponorce právě děje.

Jak už bylo zmíněno výše, tento program byl pouze testovací a vše probíhalo jen jako příprava na finální verzi programu, která je popsána v následujících částech této bakalářské práce.

3.4 Program pro komunikaci s miniponorkou - klient

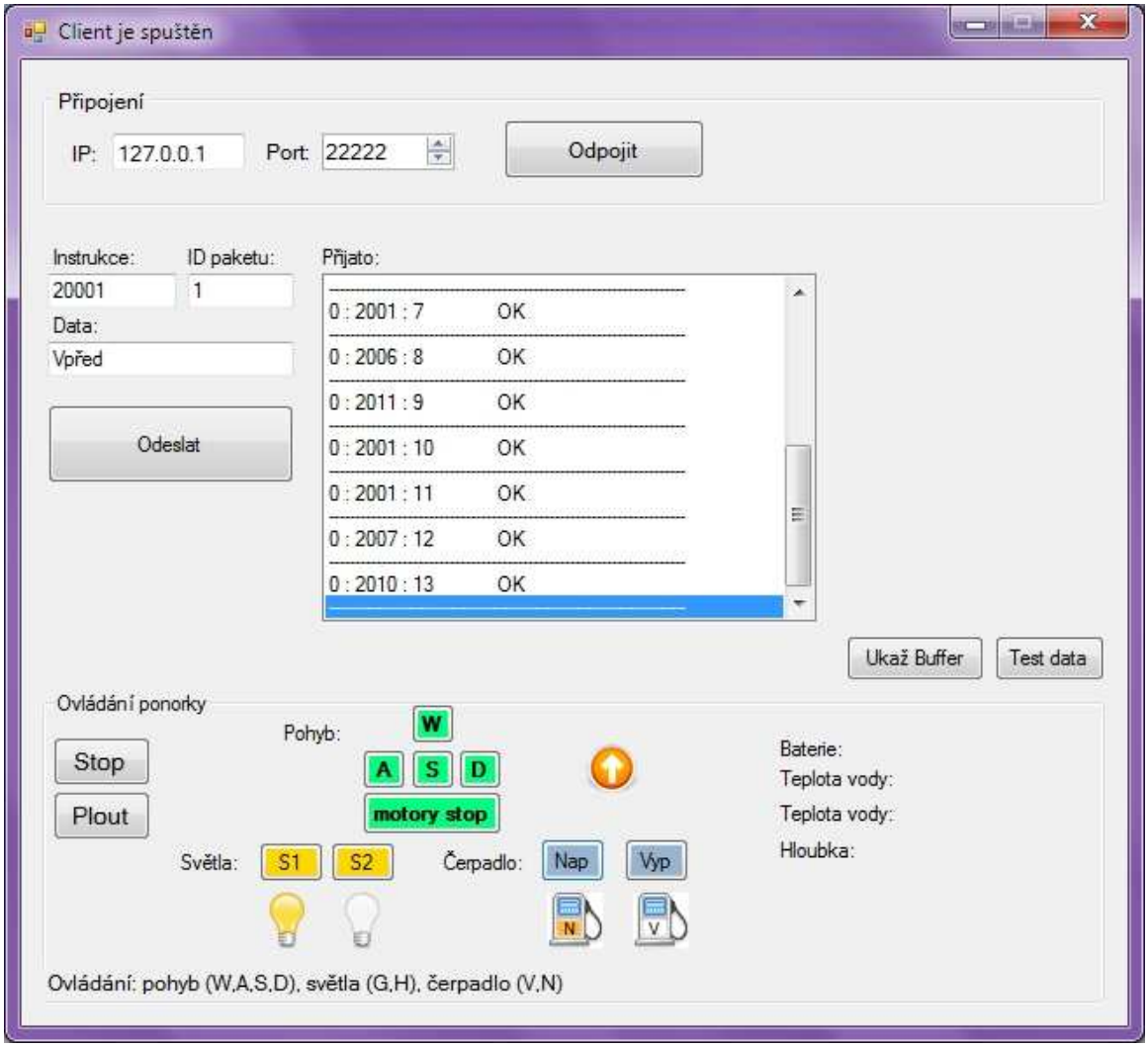
Nejdůležitější částí této práce bylo napsat a vytvořit program, který bude schopný odesílat data do miniponorky a přijímat informace nazpět. Tento program byl nazván PonorkaComm. Název je logický, protože se jedná o komunikaci s ponorkou. Program obsahuje klienta a virtuální ponorku (server) o které se píše v kapitole 3.5. Klient je program, který slouží k odesílání dat na server a přijímá zpět data ze serveru. Komunikace probíhá přes síťové rozhraní LAN a je k tomu využito Windows API funkcí, které tuto komunikaci zprostředkovávají. Základním pilířem této komunikace byl komunikační program Katana, který ovládal robota, jehož základní funkcí byl pohyb ramene. Navržení komunikace s ponorkou funguje na stejném principu jako již zmiňovaná komunikace Katana.

3.4.1 Navržení programu klienta pro ovládání miniponorky

Nejdůležitější částí navržení programu klienta bylo rozvržení jeho funkcí, to znamená, co by měl umět, přes jaký port a IP adresu se bude připojovat, jaké instrukce bude posílat a jak bude graficky navržen. Základní komunikace, která byla navržena, je přes IP adresu „127.0.0.1“ a přes port „22222“. Pokud je aktivní server (server je spuštěn a dostupný), je možné se k němu přes IP adresu a port připojit a následně komunikovat a odesílat příkazy miniponorce nebo data pro získání informací ze senzorů. Program klienta by měl určitě obsahovat i dialogové okno, které bude uživatele informovat o tom, jaká data již do klienta dorazila, a která byla vyřízena. Každá operace nebo odeslaná zpráva by měla mít své identifikační číslo (ID), podle kterého se bude uživatel moci orientovat. Toto identifikační číslo slouží v programu k tomu, aby byl schopen rozeznat příkazy mezi sebou a vyřídil je tak, jak mu byly uživatelem zadány. Spolu s identifikačním číslem by měly být odeslány i data o příkazu, který má být v ponorce vykonán. Tato data by měla být konstantou, kterou bude mít každý příkaz přiřazen. Jeden příkaz bude mít tedy i jednu konstantu, aby bylo možné je na serveru rozeznat a podle přijaté konstanty daný příkaz vykonat. V ovládací části by měl mít klient tlačítka a obrázky, které budou jednotlivé příkazy a činnost miniponorky reprezentovat tak, aby uživatel jednoznačně věděl, zda vše funguje správně a zda se miniponorka pohybuje nebo koná činnost tak, jak bylo zamýšleno.

3.4.2 Grafická realizace programu klienta pro ovládání miniponorky

Poté co byl program klienta pro ovládání miniponorky navržen, bylo možné tento program realizovat. V následující části bude nejdříve popsáno, jak byl program navržen graficky a poté budou popsány funkce jednotlivých částí programu a samotných tlačítek, které umožňují ovládání miniponorky.



Obr. 11: První grafický a funkční návrh pro program klienta

Na obrázku 11 můžeme vidět design, který byl navržen pro klienta podle návrhových kriterií. Nahoře v liště je zobrazena informace o tom, zda je klientská část vypnuta nebo spuštěna. Pod lištou se nachází připojovací část. IP adresa byla nastavena na již zmiňovanou hodnotu „127.0.0.1“ a port na „22222“. Tyto hodnoty jsou uživateli umožněny měnit, aby bylo případně možné se připojit i k jinému serveru s jiným portem

a jinou IP adresou. Vedle se nachází tlačítko o připojení, které po stisku připojí klienta na server a změní svůj text na „odpojit“ a klient čeká na další instrukce.

Pod připojovací částí je informační část. V této části se uživateli zobrazuje vše co se děje na serveru v závislosti na tom co uživatel odeslal jako příkaz. Vidíme zde „ListBox“, který vypisuje vše, co bylo odesláno ze serveru na klienta, to znamená, všechna zpracovaná i nezpracovaná data, a oznámí uživateli, zda byla nebo nebyla vyřízena. Tato zpráva je vypsána v pořadí velikost dat, konstanta oznamující činnost miniponorky, identifikační číslo příkazu a informaci o tom, zda vše proběhlo v pořádku (OK), vše bylo provedeno (DONE) nebo nastala nějaká neočekávaná chyba při zpracování příkazu (ERR). Všechny informace se zde vypisují tak, jak byly ze serveru přijaty. V této části je také možnost odeslat vlastní data do miniponorky a nechat miniponorku tento příkaz zpracovat. Je možné zvolit vlastní konstantu, ID příkazu a data, která mají být na server odeslány a v miniponorce vykonány. Odeslání dat umožňuje tlačítko „Odeslat“.

Ve spodní části tohoto programu je ovládací část. Tato část obsahuje mnoho tlačítek, „PictureBoxů“ a „Labelů“. Nejdůležitějším tlačítkem je tlačítko „Start“, které není na obrázku 11 vidět, protože se původně nachází na místě tlačítka „Stop“. Po stisku tlačítka „Start“ se aktivuje další tlačítko „Plout“, které následně umožňuje odeslání dat do miniponorky, jak tlačítky v programu, tak tlačítky na klávesnici. Nejdříve jsou v programu tlačítka pro ovládání, které nesou stejný název, jako tlačítka na klávesnici. To znamená pro pohyb vpřed „W“, vzad „S“, doleva „A“ a doprava „D“. Jediným rozdílem je tlačítko „motory stop“, které zastaví motory v miniponorce, ale pokud je ovládána klávesnicí, je tato činnost vykonána automaticky po události „OnKeyUp“. Vedle ovládacích tlačítek je „PictureBox“, který okamžitě oznamuje uživateli, jaké tlačítko stiskl, to znamená, jakým směrem se miniponorka pohybuje. Pod ovládacími tlačítky jsou další, která umožňují uživateli spustit osvětlení v miniponorce nebo zapnout a vypnout čerpadla balastních komor. Vše je doplněno o čtyři „PictureBoxy“, které opět uživatele informují o činnostech, které byly spuštěny nebo vypnuty. Na pravé straně ovládací části jsou „Labely“, které vypisují informace ze senzorů, a jsou po určitém čase aktualizovány. Nakonec je ve spodní části napsáno, jakým způsobem lze miniponorku ovládat na klávesnici a která tlačítka použít. Ovládání probíhá obvyklým způsobem. Jak už bylo zmiňováno, pro pohyb jsou to tlačítka, „W“. „A“, „S“ a „D“, pro rozsvícení nebo zhasnutí světel „G“ a „H“ a pro spuštění nebo vypnutí čerpadel do balastních komor

jsou to tlačítka „V“ a „N“. Tlačítka pro napouštění a vypouštění jsou logicky podle počátečních písmen obou českých slov.

3.4.3 Programovací část programu pro ovládání miniponorky – připojení

V této části bakalářské práce je popsána programovací část programu a tlačítek. Je zde popsáno, jak probíhá připojení k serveru, odesílání dat na server a další důležité informace o ovládání miniponorky.

Po spuštění klienta je nejdříve nutné připojit se k serveru, aby bylo možné s ponorkou komunikovat. V následující upravené části kódu je vidět, jak je provedeno připojení k serveru.

```
if (clientRunning)
{
	client->ShutDownClient();
	client->StopCoreThread();
} else {
	unsigned short port = (numPort->Value);
	ipAddress = (txtIPAddress->Text);
	TCPCSETUPOPT options = {ipAddress, port};
	client->SetupClient(&options);
```

Ukázka 1: Připojení k serveru

Tato část programu není autentická s tou, která se nachází v reálném programu. Důležité je vidět jak dochází ke spojení. Nejdříve je zkontrolováno, zda už je program připojen. Pokud je již připojen, uživatel se snaží od serveru odpojit, a tak je vyslán signál pro odpojení. Pokud je však klient nepřipojen k serveru, jsou do proměnných port a ipAddress uloženy hodnoty, které uživatel může nastavit v programu. Poté jsou obě hodnoty uloženy do „options“ a připojení vykonáno procedurou „SetupClient“. Následně je již program úspěšně připojen. Pokud není server dostupný nebo došlo k nějaké chybě, klient připojit nelze.

3.4.4 Programovací část programu pro ovládání miniponorky – výpis dat

V této části programu je zobrazen „ListBox“, který vypisuje vše co je odesláno ze serveru, ať už se jedná o vyřízené příkazy nebo přijatá data ze senzorů. Vedle „ListBoxu“ je zde také možnost odeslání vlastních dat a příkazů do miniponorky. Uživatel může do každého ze tří „TextBoxů“ zapsat data, která vyžaduje, aby byla zaslána na server.

```
char* stringPointer = (txtData->Text);

int dataLength = txtData->TextLength + 1;

TCPPACKETHEADER head={dataLength,txtInstrukce->Text),(txtIdPaketu-
>Text)};

int result = this->client->SendMsg(&head, stringPointer);
```

Ukázka 2: Odeslání dat na server

Zde si postupně ukládáme data do proměnných a poté jsou odeslána proceduře „SendMsg“. Tato zmíněná část programu opět není autentická s tou, která se nachází v reálném programu, je pouze informativní. Odeslaná data jsou poté zpracována a odeslána na server, odkud je zpět zaslána informace o jejich zpracování a vypsána právě v „ListBoxu“. Dále se v této části nachází dvě tlačítka a to „Ukaž buffer“ a „Test data“. Tlačítko „Ukaž buffer“ slouží k tomu, aby si uživatel mohl v novém okně zobrazit data, která byla přijata ze serveru. Naopak tlačítko „Test data“ slouží k testovacímu odeslání dat na server a následnému zjištění zda komunikace funguje správně.

3.4.5 Programovací část programu pro ovládání miniponorky – ovládání

V poslední části programu se nachází ovládání a samotná komunikace s miniponorkou. Nejprve je nutné stisknout tlačítko „Start“, které slouží k aktivaci ovládání. Stisknutím tohoto tlačítka se nejdříve nastaví „visible“ na „false“ a u tlačítek „Stop“ a „Plout“ na „true“. Dále se také aktivují obrázky v „PictureBoxech“ a tak je start pro uživatele mnohem věrohodnější. Aktivované tlačítko „Stop“ je naprogramované tak, aby po stisku odeslalo do ponorky vypnutí všech činností, aby se ponorka nestala neovladatelnou. Toto je vyřešeno zavoláním postupně všech procedur, které ukončí v ponorce činnost. Jako příklad je možné uvést zastavení motorů „client->stop()“.

Další tlačítko, které je aktivováno je „Plout“. Pokliknutím na toto tlačítko se aktivují všechna tlačítka pro ovládání miniponorky. Vše je umožněno proměnnou, která je primárně nastavena na „false“ a zde se aktivuje na „true“ a ve zbytku programu podmiňuje funkcí „if“ ovládání miniponorky a odeslání příkazů. Následně je zde aktivován „timer“, který každých třicet sekund odešle dotaz na stav baterií, teplotu a další informace ze senzorů a následně je vypíše do programu. Jakmile je aktivováno ovládání miniponorky, je možné využívat tlačítka pro pohyb, zapnutí nebo vypnutí obou osvětlení a zapnutí nebo vypnutí balastních komor. Ovládání přes tyto tlačítka funguje u všech na stejném principu. Jako příklad je uveden příkaz pro napouštění čerpadla.

```
if ((vypous == false) && (start == true)) {

      if (napous == false)    {
            pictureBox8->Image = Image::FromFile("img/pump_onn.png");
            pictureBox6->Image = Image::FromFile("img/pump_onn.png");
            napous = true;
            client->cerpadlonz();
                        }
      else                  {
            pictureBox8->Image = Image::FromFile("img/pump_offn.png");
            pictureBox6->Image = Image::FromFile("img/pump_offn.png");

            napous = false;
            client->cerpadlonv();
                        }
```

Ukázka 3: Ovládání pomocí tlačítek v programu

V této části kódu vidíme nejdříve již zmiňovanou kontrolu zda již bylo stisknuto tlačítko „Start“. Zároveň v této podmínce musí platit, že „vypous“ je nastaveno na „false“. Právě proměnná „vypous“ značí aktivní respektive neaktivní vypouštění čerpadla a tak díky této kontrole zamezíme souběžné aktivitě vypouštění a napouštění. Následuje podmínka, která kontroluje, zda už čerpadlo napouští nebo ne. Pokud nenapouští, jsou „PictureBoxům“ přiřazeny obrázky, které uživateli značí, že čerpadla napouští, dále je proměnná „napous“ nastavena na „true“ a tak nebude možné během napouštění vypouštět a hlavně je poslán příkaz na server pro zapnutí napouštění čerpadla. Pokud ovšem čepradlo zapnuto bylo, vykonala se druhá část podmínky, která opět nastavila „PictureBoxům“ obrázky o vypnutí, přepnula proměnnou „napous“ na „false“

a odeslala ke zpracování na server, který po vyřešení zaslal informaci o vypnutí zpět uživateli.

Druhá možnost ovládání miniponorky je tlačítky na klávesnici. Vše funguje jako s tlačítky v programu, jen s tím rozdílem, že u pohybu miniponorky stačí tlačítko v programu stisknout a ponorka se dále tímto směrem pohybuje. Na klávesnici je nutné tlačítko pro pohyb držet, protože vše funguje na základních událostech „KeyDown“ a „KeyUp“. Při události „KeyDown“ je zjištěno, jaké tlačítko je stisknuto a podle toho je odeslán příkaz na server. Naopak událost „KeyUp“ zjistí, které tlačítko bylo uvolněno a stejně jako událost „KeyDown“ odešle příkaz na server. Opět je zde uveden příklad pro napouštění čerpadla:

```
if ((e->KeyCode == Keys::N) && (x == false) && (vypous == false)) {
      x = true;
      if (napous == false)    {
            pictureBox8->Image = Image::FromFile("img/pump_onn.png");
            pictureBox6->Image = Image::FromFile("img/pump_onn.png");
            napous = true;
            client->cerpadlonz();
```

Ukázka 4: Ovládání pomocí tlačítek na klávesnici

Tato část kódu se velice podobá již zmiňované části pro tlačítka na klávesnici. Základní rozdíl je v kontrole stisknutého tlačítka. Druhý rozdíl je v kontrole proměnné „x“. Důvod této kontroly je ten, že chceme odeslat příkaz pro spuštění čerpadla pouze jednou a pokud by tato podmínka nebyla a tlačítko „N“ by bylo stlačeno po delší dobu, došlo by k opakovanému odeslání příkazu. Zbytek programu je stejný a proto není nutné ho zde celý vypisovat.

3.4.6 Programovací část programu pro ovládání miniponorky – konstanty

Každá procedura, která je klientem odeslána ke zpracování a následně odeslána jako příkaz na server má přidělenou svou unikátní konstantu pro daný úkon. Tyto konstanty jsou uloženy v souboru „ponorkaConstants.h“ a jsou vždy volány podle toho, jaký příkaz má být vykonán. Jako příklad je zde opět zapnutí čerpadel:

```
void CponorkaClient::cerpadlonz()   {
      this->sendNoDataCommand(RBTCMD_PONORKA_CERPADLONZ);
```

Ukázka 5: Volání procedury pro poslání dat na server

V této části kódu je vidět samotná procedura pro napouštění čerpadla, která je volána událostí a poté jsou data opět odeslána do další procedury „sendNoDataCommand“, která vyžaduje konstantu, která je uložena v konstantě v závorce. Tyto konstanty jsou očíslovány od dvou tisíc výše. Soubor s konstantami je přístupný jak klientovi, tak serveru, který k němu má přístup a tak je schopen rozeznat příkazy přijaté od klienta a správně je zpracovat. V aktuálním programu se nachází dohromady sedmnáct konstant pro různé úkony, které jsou uloženy tímto způsobem:

```
#define RBTCMD_PONORKA_CERPADLONZ                RBTCMD_PONORKA_BASE + 10
```

Ukázka 6: Příklad uložení konstant v programu

3.4.7 Programovací část programu pro ovládání miniponorky – odeslání

V předchozích částech této práce byly postupně ukázány volané procedury, které postupně směřovaly k odeslání příkazu na server. Nyní bude popsána nejdůležitější část odesílání příkazů.

```
void CRobotClient::sendNoDataCommand(unsigned int command)
{
	char *buffer = NULL;
	TCPPACKETHEADER head = {0, command, this->packetID};
	this->buffer->AddMsg(&head, buffer);

	this->SendMsg(this->buffer->GetLastAddedMsgPointer()->header,
	this->buffer->GetLastAddedMsgPointer()->data);
	this->packetID++;
}
```

Ukázka 7: Odesílání dat na server

V kapitole 3.4.6 byly popsány konstanty. A právě tyto konstanty jsou vstupem do této procedury a jsou poslány jako příkaz na server. Konstanty uložíme do hlavičky „head“ a přídáme i velikost dat (v tomto případě 0) a id příkazu, který je posílán a podle kterého se program i uživatel může orientovat. Následně jsou „buffer“ a hlavička „head“ přídány do zprávy a poté přes proceduru „SendMsg“ odeslány na server, kde dojde k jejímu zpracování a klientovi se vratí data, která budou obsahovat id této zprávy, konstantu, která byla vykonána a informace o úspěšnosti nebo neúspěšnosti vykonání.

3.5 Program pro komunikaci s miniponorkou – server

Aby bylo možné program klienta otestovat, bylo nutné mít k dispozici i serverový progam, který bude schopný přijímat data z programu klienta, zpracovat je a poslat zpět informaci o vyřízení. Virtuální server představuje miniponorku a tak je možné si na něm vyzkoušet základní odesílaní dat, příkazů a příjímat je od různých klientů. Díky tomuto serveru je možné vše důkladně otestovat dříve, než bude uskutečněn ponor a je možné vše správně vyladit a opravit, aby během plánovaného ponoru nedocházelo k výpadkům nebo zbytečným chybám v programech. Komunikace funguje stejně jako u klienta a probíhá přes síťové rozhraní LAN a je k tomu využito Windows API funkcí, které tuto komunikaci zprostředkovávají. Základním pilířem této komunikace byl komunikační program „Katana“, který ovládal robota, jehož základní pohyb byl pouze pohyb ramene. Navržení komunikace s ponorkou funguje na stejném principu jako již zmiňovaná komunikace „Katana“.

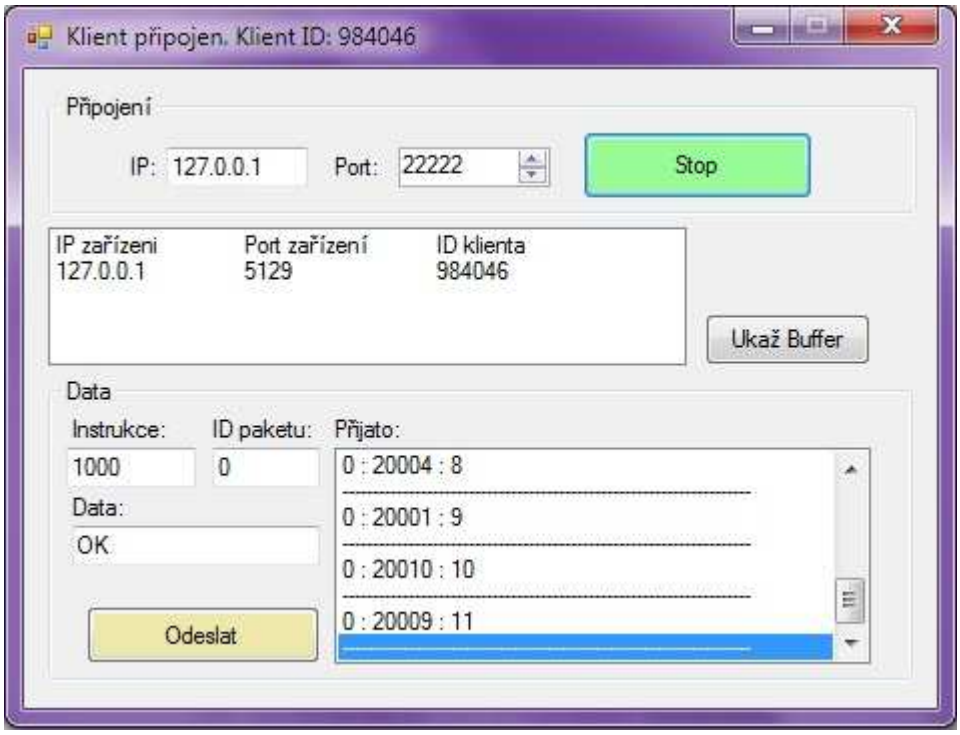
3.5.1 Navržení programu serveru pro ovládání miniponorky

Primárně bylo nutné navrhnout, jak by měl server vypadat a co by měl umět. Aby se k serveru mohl připojit klient, bylo nutné nastavení IP adresy a porty. Hodnoty jsou tedy pro IP adresu „127.0.0.1“ a port „22222“. Server by měl také obsahovat informace o připojených klientech a odkud jsou připojeni. K tomu by mohl sloužit „ListBox“. Dále by tu měl být další „ListBox“, který bude zobrazovat přijaté příkazy. Tato možnost by sloužila převážně pro testovací účely zda vše fungovalo správně.

3.5.2 Grafická realizace programu serveru pro ovládání miniponorky

Program byl sestaven pro co nejvyšší přehlednost, aby při testování byly nejdůležitější informace co nejlépe viditelné. Program obsahuje pouze základní funkční věci, protože se jedná pouze o testový server, který slouží jako virtuální ponorka. Na obrázku 12 je vidět, jak vypadá finální verze programu serveru pro miniponorku. Stejně jako v programu klienta se v horní části nachází stav serveru. V tomto případě je tam napsáno, že je klient připojen, a jaké má ID. V horní části programu je opět možnost nastavení IP adresy a portu, které může uživatel sám měnit. Vedle se nachází tlačítko, které aktivuje server a je možné se k němu následně připojit. V následující části programu se nachází „ListBox“, který obsahuje data o připojeném klientovi. Konkrétně jsou

to informace o IP připojeného zařízení, Portu připojeného zařízení a ID klienta. Vedle „ListBoxu“ je tlačítko „Ukaž buffer“, které stejně jako u klienta zobrazí v novém okně přijaté příkazy a jejich vyřešení.



Obr. 12: Program serveru pro ovládání miniponorky

V poslední části programu jsou informace o datech. Na pravé straně je opět „ListBox“, který vypisuje údaje o přijatých příkazech z klienta. Vedle „ListBoxu“ je znovu možnost odeslat data přímo ze serveru klientovi. Tato možnost slouží hlavně k testovacím účelům.

3.5.3 Programovací část programu serveru

Aktivování klienta probíhá na stejném principu, jako když se klient snaží připojit k serveru. Taktéž odesílání dat ze serveru na klienta probíhá stejně jako opačným směrem. Nejdříve je nutné vyplnit údaje o instrukci, ID zprávy a datech jaké chceme na klienta odeslat. Kliknutím na tlačítko „odeslat“ je provedeno odeslání na klienta. Posledním tlačítkem v programu serveru je tlačítko „Ukaž buffer“, které zobrazí v novém okně veškeré údaje o přijatých a odeslaných příkazech z klienta na server. Tyto příkazy by mohly skončit jedním z následujících výsledků a to „ERR“ jako chyba, „WAIT“ (čeká), „OK“ (proběhlo v pořádku), „DONE“ (provedeno) nebo „UNKNOWN“ (neznámý výsledek).

3.6 Finální verze programu pro ovládání průzkumné miniponorky

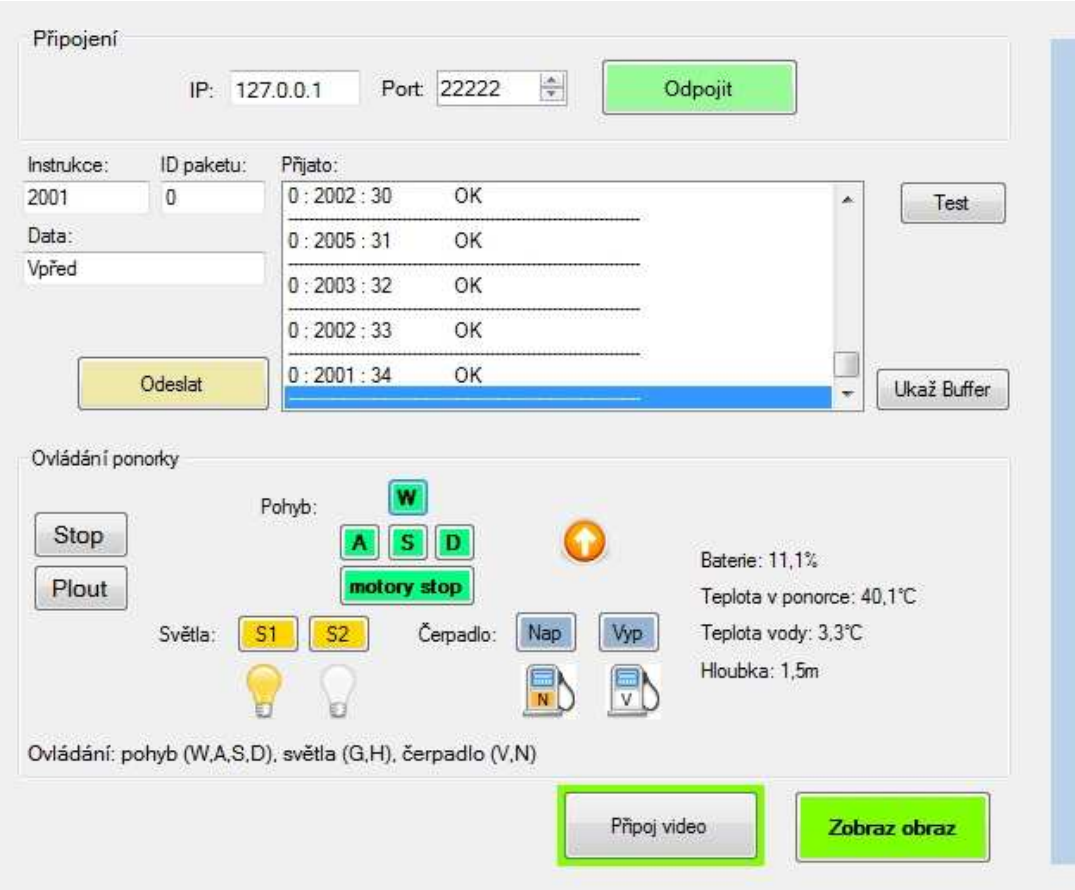
V této části bakalářské práce bude popsána finální verze programu pro ovládání průzkumné miniponorky. Jedná se konkrétně o klienta, kterého bude používat uživatel na břehu a bude pomocí klienta ovládat miniponorku, posílat jí příkazy a získávat co nejvíce dat a informací právě z miniponorky. Hlavní změnou klienta, který byl popsán v kapitole 3.4 je spojení s klientem, který přijímá obraz z webkamery v miniponorce. Oba klienti byli spojeni do jednoho a tak vznikl mnohem komplexnější program, než byl původní. Hlavním důvodem spojení těchto dvou programů byla nespokojenost uživatele. Během ovládání miniponorky se tak nemusí zabývat přepínáním oken mezi klienty, ale zajímá se pouze o stav miniponorky. V následujících částech bude popsán grafický design tohoto programu, a jakým způsobem bylo docíleno spojení dvou klientů.

3.6.1 Cíl finální verze programu

Finální verze programu se tedy skládala ze dvou klientů. První klient byl vytvořen v rámci této práce a byl popsán v kapitole 3.4. Druhý klient, který byl k dispozici, byl od Vladimíra Starce. Tento klient přijímal obraz z webkamery z miniponorky a byl schopen s obrazem pracovat. Mým úkolem bylo přidat program, který přijímá obraz, k mému programu, který ovládá miniponorku a poté vzít pouze obraz a přidat ho do mého programu. Na konci měl tedy vzniknout program, který bude schopen odesílat data do miniponorky a zároveň zobrazovat obraz, který z miniponorky přichází. Vše by mělo být graficky uživatelsky přívětivé, aby se v programu mohl uživatel co nejlépe orientovat. To znamená, že tu nesměly chybět žádné důležité „Labely“, „PictureBoxy“ a informace z miniponorky. Obraz měl být upraven tak, aby ho bylo možné zvětšit na celou obrazovku a v tomto režimu na něm zobrazit také veškeré nutné informace. K této práci s obrazem velmi dobře posloužil program pro práci s videem, který je popsán v kapitole 3.3. Díky tomuto programu jsem se již v začátku bakalářské práce naučil se základní úpravou videa a tak při závěrečném spojení obou programů nebyl problém obraz nadále upravovat. Nejobtížnější na spojení obou klientů bylo samotné načtení videa do mého programu a připojení jiného klienta tak aby s ním v mém programu bylo možné i nadále pracovat a získávat z něj veškeré důležité informace a data, konkrétně o videu. V následující kapitole bude prezentován samotný design finálního programu.

3.6.2 Grafický design finální verze programu

Na obrázku 13 je zobrazena finální verze klienta a dále budou popsány rozdíly od předchozí verze klienta. Jak je již zřejmé, základní rozdíl bude v přidaném programu klienta a obrazu z něj.



Obr. 13: Finální verze programu pro ovládání miniponorky

Na obrázku 12 je dobře vidět, že v ovládací části programu došlo jen k drobným změnám v designu programu. Tyto změny byly hlavně uživatelské a slouží k větší přehlednosti v programu. Mezi dvě hlavní novinky patří tlačítka pod ovládáním. První tlačítko „Připoj video“ slouží k připojení klienta, který přijímá obraz z webkamery. Poté co je připojen druhý klient, musí se v něm nastavit připojení k severu v miniponorce a získat obraz. Pokud vše proběhne v pořádku, je možné stisknout tlačítko „Zobraz obraz“. Toto tlačítko zobrazí obraz v prostoru na pravé straně od ovládání na místě aktuálního modře zbarveného „Labelu“, který pokračuje ještě mimo tento obrázek a je s ním nadále možné pracovat. Celý design je tvořen pro co nejvyšší přehlednost, která je při praktickém využití v terénu velice důležitá, protože se zde vyskytuje příliš mnoho faktorů, které by mohly být rušivé. Přesnější obrázky si je možné prohlédnout v příloze.

3.6.3 Programovací část finální verze programu

V této části práce bude popsán postup pro získání obrazu z jiného programu klienta. K tomuto řešení byl potřeba kompletní program pro přenos obrazu z miniponorky. Nejdříve bylo nutné pracovat právě v tomto programu, kde byl vytvořen nový vnitřní projekt, který byl nazván „remoteCam“. Do tohoto nového projektu byly vloženy všechny důležité hlavičkové soubory z programu klienta. Než byl tento nový projekt přeložen, bylo nastaveno, aby se uložil jako knihovna, aby k němu bylo možné přistupovat z jiného programu. Poté zde byl vytvořen form „remoteCamControl“, který obsahoval tlačítko. Toto tlačítko je zobrazeno v mém programu pod názvem „Připoj video“. Klinutím na toto tlačítko se vyvolá událost a otevře se právě klient tohoto programu pro zobrazení obrazu z miniponorky. V tomto projektu byla vytvořena také procedura „SetObrazLabel“, která má za úkol přenést video z jednoho klienta do druhého a obsah procedury je následující:

```
x->PB = gcnew PaintBox();
x->PB->DoubleBuffer(true);
x->PB->Parent = La->Parent;
x->PB->Left = La->Left;
x->PB->Top = La->Top;
x->PB->Width = La->Width;
x->PB->Height = La->Height;
La->Visible = false;
return;
```

Ukázka 8: Zobrazení obrazu z původního klienta v mém programu

V tomto kódu je vidět, že se nejdříve vytvoří nový „PaintBox“, do kterého se bude později obraz vykreslovat. Poté je mu nastaven „Doublebuffer“, aby se obraz vykresloval plynule a správně. Následně se vytvoří rodič prvku „La“ a převezme jeho vlastnosti jako je „Left“, „Top“, „Width“ a „Height“. „La“ je pouze vnitřní proměná této procedury, která sem vstupuje jako „Label“, který je v mém programu podbarven modrou barvou, neboli ten, na který se bude obraz vykreslovat. Po tom co „PaintBox“ převezme všechny vlastnosti, je možné zrušit viditelnost „Labelu“ a dojde k návratu do mého programu.

Přenesení obrazu do mého programu se provádí pomocí již zmiňovaného tlačítka „Zobraz obraz“. Tato procedura zavolá právě proceduru „SetObrazLabel“, jejíž kód je vidět výše. Kliknutím na tlačítko „Zobraz obraz“ se provede následující kód:

```
if (remoteCamControl1->x)     {
      remoteCamControl1->SetObrazLabel(lblKUK);
      this->remoteCamControl1->x->PB->DoubleClick += gcnew
      System::EventHandler(this, &Form1::PB_DoubleClick);
      label20->Visible = false;
} else      {
      label20->Visible = true;
            }
```

Ukázka 9: Volání procedury „SetObrazLabel“ a vytvoření nové události doubleclick

Jak je v této části kódu vidět nejdříve pomocí funkce „if“ kontrolujeme zda již byl klient s obrazem připojen a pokud ne, tak se aktivuje „Label20“, který oznamí že není připojen klient. Pokud je vše v pořádku, je zavolána procedura „SetObrazLlabel“ a vykonáno tak připojení obrazu. Zároveň se vytvoří i událost „Doubleclick“ právě pro nově přidaný obraz v mém programu a tak je možnost pracovat s touto událostí dále a je možné aktivovat režim celé obrazovky právě touto událostí. Pokud stiskneme tlačítko „Připoj video“ je zobrazen nový klient, a aby se nestalo že po zavření mého klienta zde zůstane druhý klient otevřený, byla vytvořena procedura, která zavře i tohoto klienta.

Když je obraz připojen a tak aktivována událost „Doubleclick“, je možné pracovat i s obrazen na celou obrazovku. I v tomto případě bylo využito zkušeností z testování programu pro video, které je v kapitole 3.3.2. Vše bylo téměř totožné a tak byly použity stejné funkce a obraz se tak mohl zobrazit na celé obrazovce i se všemy „Labely“ a „Pictureboxy“ a bylo vše pro uživatele graficky přehledné. Pozice všech Labelu a pictureboxu jsou nastaveny programově, aby byly na každém monitoru s různým rozlišení na přibližně stejné pozici. V režimu na celou obrazovku je možné vidět údaje ze všech senzorů, jako je stav baterií, teploty, hloubky a přidaní nových dat není díky dobře napsanému programu žádný problém. Dále je tam vidět směr pohybu miniponorky, stav světlometů a také stav čerpadel balastních komor.

Veškeré ovládání miniponorky lze realizovat jak tlačítky v programu i na klávesnici a řešení je úplně stejné jako v kapitole 3.4.5. Tlačítka jsou ošetřena aby fungovala až po připojení klienta. Spojení obou klientů tedy proběhlo v pořádku a program byl připraven pro praktickou realizaci v miniporce. V závěru je potřeba dodat, že je nutné, aby v miniponorce byly spuštěny oba servery, jak pro video tak pro ovládání, aby bylo možné se k miniponorce připojit a odesílat data.

4 Praktické otestování programu

Poslední částí bakalářské práce bylo prakticky otestovat programy, které byly napsány a vytvořeny. Původní plán byl vyzkoušet programy v miniponorce v reálném prostředí u nějaké nádrže, ale bohužel to nebylo před odevzdáním bakalářské práce možné. Při oživování miniponorky, která byla rok nečinná, se nerozeběhl jeden z motorů a tak by na případném ponoru nebyla ponorka schopná pohybu vpřed.

Jedinou možností praktického otestování bylo využití serveru jako virtuální miniponorky a odesílat data na tento server. Následně bylo možné připojení klienta k serveru a odesílaní dat, která server zpracoval a odeslal data zpět klientovi. Během testování byl zkoušen pouze jeden připojený klient a vše probíhalo téměř bez problému. Pokud se vyskytla chyba, bylo okamžitě zjištěno, co bylo za problém a byl softwarově opraven a znovu vyzkoušen. Díky tomuto testování bylo možné otestovat, zda všechny programy správně běží a bude možné je v budoucnu využít pro ovládání miniponorky.

Během testování programů se vyskytlo několik chyb. Prvním problémem bylo ovládání tlačítky na klávesnici. Pokud chtěl uživatel pohybovat miniponorkou a držel tak jedno z pohybových tlačítek, probíhala událost „KeyDown“ stále dokola a tak se stále odesílala informace o pohybu do miniponorky, což u této události není obvyklé. Proto bylo nutné přidat podmínku, která řeší tento problém. Dalším problémem byla čerpadla, která nemohla fungovat obousměrně současně. To znamená, že nebylo možné, aby čerpadla zároveň napouštěla a vypouštěla. Tento problém byl vyřešen přidáním nové proměnné a jednoduchou podmínkou. Po vytvoření finální verze programu a zprovoznění obrazu z webkamery v miniponorce nastal poslední zásadní problém. Po načtení obrazu do „PaintBoxu“ došlo k jeho blikání. Obraz několikrát za sekundu probliknul a kazilo to dojem z celého programu. Bylo nutné tedy opět prostudovat již napsané programy pro přenos obrazu z webkamery z miniponorky. Jako nejlepší řešení tohoto problému se ukázalo využití funkce „doublebuffer“ a tak bylo blikání obrazu odstraněno a program byl úspěšně dokončen.

Reálný ponor tedy bohužel nebyl uskutečněn z již zmíněných důvodů a tak proběhla pouze kontrola programů na počítači. Podle této kontroly jsou programy v pořádku a připraveny k použití.

Závěr

Cílem této práce bylo navržení a vytvoření interaktivního softwaru pro vzdálené ovládání mobilních robotů. Program měl komunikovat s průzkumnou miniponorkou přes síťové rozhrání LAN a mělo být využito koncepce klient – server.

Nejdříve bylo nutné prostudovat již napsané programy a otestovat je, pro získání základních zkušeností pro následné realizování této práce. Původní program pro video bylo vhodné vylepšit a naučit se na něm základní práci s jazykem C++ a vývojovým prostředím Microsoft Visual Studio 2008. Tato práce s videem se později ukázala jako velice produktivní, protože bylo možné v závěru práce na finální verzi programu pro vzdálené ovládání miniponorky využít získané zkušenosti a velice rychle upravit program do co nejvíce uživatelsky přívětivého stavu.

Finální verze programu byla vytvořena spojením mého klienta pro ovládání miniponorky a klienta pro získávání obrazu z webkamery miniponorky. Do mého programu bylo přidáno zobrazení obrazových dat, která jsou nejdříve načtena do původního klienta a poté přesměrována do mého programu. Ovládání miniponorky je možné dvěma způsoby a to tlačítky na klávesnici nebo tlačítky v programu. Během testování bylo nutné vytvořit i server, jako virtuální ponorku, a na něm vše otestovat, aby byl program připraven pro test softwaru miniponorky v reálném prostředí.

Toto závěrečné testování bohužel nebylo možné z důvodu závady na motoru miniponorky. Do budoucna by tedy bylo nejlepší, kdyby byl hardware testován dříve a bylo možné včas vše opravit a vyzkoušet nově naprogramovaný software.

Tato práce by měla být přínosná pro další pokračování v projektu miniponorka. V dalších letech by bylo možné tento program spojit s dalším klientem, který zpracovává naklonění miniponorky a tak by mohl vzniknout ještě komplexnější software. Samozřejmě bude na těchto programech ještě mnoho další práce a má finální verze určitě není tou poslední.

Použitá literatura a zdroje informací

[1] *Robotika.cz* [online]. 2010 [cit. 2013-05-11]. Dostupné z: http://robotika.cz/cs

[2] HLAVÁČ V., SEDLÁČEK M.: Zpracování signálu a obrazu, Skripta FEL ČVUT, Praha 2000, ISBN 80-01-02114-9

[3] Mikrokontrolér PICAXE-18M2. *GM electronic* [online]. 1990 - 2013 [cit. 2013-05-11]. Dostupné z: http://www.gme.cz/mikroprocesory-picaxe/mikrokontroler-picaxe-18m2-p772-031/

[4] Robot. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-2013 [cit. 2013-05-11]. Dostupné z: http://cs.wikipedia.org/wiki/Robot

[5] Merkur Robotická ruka. *Heureka* [online]. 2000-2013 [cit. 2013-05-11]. Dostupné z: http://stavebnice-merkur.heureka.cz/merkur-roboticka-ruka/galerie/

[6] Senzor. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-2013 [cit. 2013-05-11]. Dostupné z: http://cs.wikipedia.org/wiki/Senzor

[7] SimpleJack WiFi. *Serva a jejich ovládání* [online]. 2008 [cit. 2013-05-11]. Dostupné z: http://www.serva.cz/konstrukce/simplejack-wifi/

[8] International Space Station. *NASA* [online]. 2013 [cit. 2013-05-11]. Dostupné z: http://www.nasa.gov/mission_pages/station/research/experiments/730.html

[9] Robot Curiosity už pracuje. Tady je první foto z Marsu. *Aktualne.cz* [online]. 012 [cit. 2013-05-11]. Dostupné z: http://aktualne.centrum.cz/zahranici/amerika/clanek.phtml?id=753497[10] RUDEŽ,

[10] Tanja. INFOGRAFIKA 'Da slijetanje Curiosityja nije uspjelo, NASA to ne bi ponovila 10 godina'**.** Jutarnjilist [online]. 2012 [cit. 2013-05-11]. Dostupné z: http://www.jutarnji.hr/template/article/article-print.jsp?id=1045986

[11] Local Area Network. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-2013 [cit. 2013-05-11]. Dostupné z: http://cs.wikipedia.org/wiki/Local_Area_Network

[12] Klient-server. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-2013 [cit. 2013-05-11]. Dostupné z: http://cs.wikipedia.org/wiki/Klient-server

[13] VNC a Vzdálená plocha - kouzlo vzdáleného přístupu. *Pctuning* [online]. 2009 [cit. 2013-05-11]. Dostupné z: http://pctuning.tyden.cz/software/jak-zkrotit-internet/12639-vnc_a_vzdalena_plocha-kouzlo_vzdaleneho_pristupu

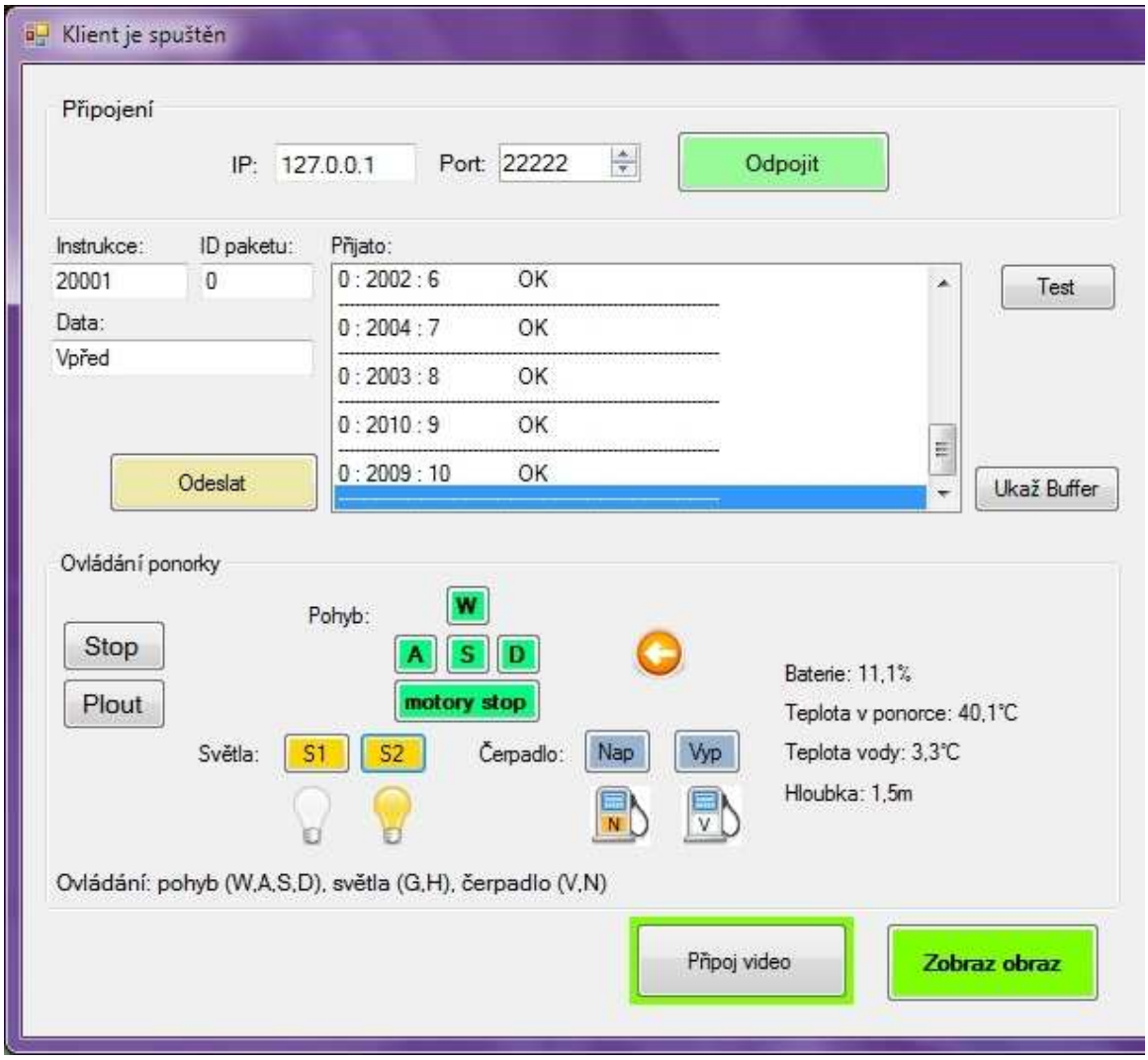
[14] PETZOLD, Charles. *Programming Windows*. Redmond, Washington: Microsoft Press, 1998. 5. ISBN 1-57231-995-X.

Přílohy

Příloha A – Přiložené CD

- Bakalářská práce (ve formátu PDF)
- Napsaný software
- Obrázky designu finální verze software (ve formátu JPG)

Příloha B – Design finální verze programu



Obr. 14: Design ovládací části finálního softwaru

Obr. 15: Fullscreen obrazu s návrhem designu