VŠB – Technická univerzita Ostrava

Fakulta strojní

Katedra automatizační techniky a řízení

Analýza nákladu dřeva z 3D modelu

Analysis of Cargo of Wood Logs from 3D Model

Student: Bc. Jan Sikora

Vedoucí diplomové práce: Ing. David Fojtík, Ph.D.

Ostrava 2018

VŠB - Technická univerzita Ostrava
Fakulta strojní
Katedra automatizační techniky a řízení

Zadání diplomové práce

Student: **Bc. Jan Sikora**

Studijní program: N2301 Strojní inženýrství

Studijní obor: 3902T004 Automatické řízení a inženýrská informatika

Téma: Analýza nákladu dřeva z 3D modelu
Analysis of Cargo of Wood Logs from 3D Model

Jazyk vypracování: čeština

Zásady pro vypracování:

1. Seznamte se a popište aktuální řešení skenování nákladu dřeva do základního 3D počítačového modelu realizovaném ve společnosti MONDI SCP, a. s. v Ružomberku (Slovensko).
2. Seznamte se a popište knihovnu Emgu CV (OpenCV) se zaměřením na analýzu 3D počítačových modelů. Zároveň zvolte a popište vhodnou knihovnu pro vizualizaci 3D počítačového modelu.
3. Popište metodu určení funkce průběhu rychlosti průjezdu jízdní soupravy nákladního automobilu bránou a navrhněte a realizujte algoritmus, který z naskenovaných dat vytvoří realistický 3D model soupravy.
4. Vytvořte a popište Windows aplikaci, která ze záznamů průběhu skenování jízdní soupravy nákladního automobilu s nákladem dřeva vytvoří a zobrazí realistický 3D model soupravy a nákladu.
5. Navrhněte metodu analýzy 3D počítačového modelu jízdní soupravy nákladního automobilu s nákladem dřeva, která 3D model rozdělí na samostatnou část nákladu dřeva a část jízdní soupravy nákladního automobilu a navrhněte směr dalšího řešení.

Seznam doporučené odborné literatury:

DEVDEPT. *Eyeshot Documentation.* Bologna, Italy, ©2006-2017. Dostupné také z: http://documentation.devdept.com/100/Common/webframe.html#topic2.html

FOJTÍK, David, Petr PODEŠVA, Miroslav MAHDAL a Milan MIHOLA. Scanning of trucks to produce 3D models for analysis of timber loads. In: *Proceedings of the 2016 17th International Carpathian Control Conference, ICCC 2016.* Danvers: IEEE, 2016. s. 194-199. ISBN 978-1-4673-8606-7.

KAEHLER, Adrian a Gary R. BRADSKI. *Learning OpenCV 3: computer vision in C++ with the OpenCV library.* Sebastopol, CA: O'Reilly Media, 2017. ISBN 1491937998.

LAGANIÈRE, Robert. *OpenCV 2 computer vision application programming cookbook: over 50 recipes to master this library of programming functions for real-time computer vision.* Birmingham, U.K.: Packt Open Source Pub., 2011. ISBN 1849513252.

SHI, Shin. *Emgu CV essentials: develop your own computer vision application using the power of Emgu CV.* Birmingham, UK: Packt Publishing, 2013. ISBN 1783559527.

SICK AG. *LMS400 Laser Measurement Sensors: Operating Instructions.* Waldkirch, Germany, ©2013-2017. Dostupné také z: https://www.sick.com/media/docs/8/98/698/Operating_instructions_LMS400_Laser_measurement_sensors_en_IM0010698.PDF

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. David Fojtík, Ph.D.**

Datum zadání: 08.12.2017

Datum odevzdání: 21.05.2018

doc. Ing. Renata Wagnerová, Ph.D.
vedoucí katedry

doc. Ing. Ivo Hlavatý, Ph.D.
děkan fakulty

Místopřísežné prohlášení studenta

Prohlašuji, že jsem celou diplomovou práci včetně příloh vypracoval samostatně pod vedením vedoucího diplomové práce a uvedl jsem všechny použité podklady a literaturu.

V Ostravě: 18. května 2018 .....

podpis studenta

PODĚKOVÁNÍ

Rád bych touto cestou vyjádřil poděkování svému vedoucímu Ing. Davidu Fojtíkovi, Ph.D. za odborné vedení, cenné rady, připomínky a ochotu při řešení problémů, které se během této práce vyskytly.

Prohlašuji, že:

- jsem si vědom, že na tuto moji závěrečnou diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. Zákon o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (dále jen Autorský zákon), zejména § 35 (Užití díla v rámci občanských či náboženských obřadů nebo v rámci úředních akcí pořádaných orgány veřejné správy, v rámci školních představení a užití díla školního) a § 60 (Školní dílo),
- beru na vědomí, že Vysoká škola báňská – Technická univerzita Ostrava (dále jen „VŠB-TUO“) má právo užít tuto závěrečnou diplomovou práci nekomerčně ke své vnitřní potřebě (§ 35 odst. 3 Autorského zákona),
- bule-li požadováno, jeden výtisk této diplomové práce bude uložen u vedoucího práce,
- s VŠB-TUO, v případě zájmu z její strany, uzavřu licenční smlouvu s oprávněním užít dílo v rozsahu § 12 odst. 4 Autorského zákona,
- užít toto své dílo, nebo poskytnout licenci k jejímu využití, mohu jen se souhlasem VŠB-TUO, která je oprávněná v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly VŠB-TUO na vytvoření díla vynaloženy (až do jejich skutečné výše),
- beru na vědomí, že - podle zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů - že tato diplomová práce bude před obhajobou zveřejněna na pracovišti vedoucího práce, a v elektronické podobě uložena a po obhajobě zveřejněna v Ústřední knihovně VŠB-TUO, a to bez ohledu na výsledek její obhajoby.

V Ostravě: 18. 5. 2018

podpis

Jméno a příjmení autora práce: Jan Sikora

Adresa trvalého pobytu autora práce: Milíkov 257, 739 81

ANOTACE DIPLOMOVÉ PRÁCE

Sikora, J. Analýza nákladu dřeva z 3D modelu. Ostrava: VŠB-Technická univerzita Ostrava, Fakulta strojní, Katedra automatizační techniky a řízení, 2018, 60s. Diplomová práce, vedoucí práce: Fojtík, D.

Úvodem diplomové práce proběhlo seznámení se s aktuálním řešením skenování nákladu dřeva na nákladních soupravách při přejímce dřeva ve společnosti MONDI SCP a.s. Z dat získaných naskenováním povrchu kamionu bylo potřeba určit průběh rychlosti během jeho průjezdu pod skenovací bránou. Se znalostí průběhu rychlosti kamionu je možno vytvořit jeho realistický 3D model. Poslední část práce se věnuje segregací nákladu dřeva převáženého na kamionu od kamionu samotného.

ANNOTATION OF THESIS

Sikora, J. Analysis of Timber Loads from 3D model. Ostrava: VŠB-Technical university of Ostrava, Faculty of Mechanical Engineering, Department of Control Sytems and Instrumentation, 2018, 60s. Thesis, thesis head: Fojtík, D.

Introduction of the thesis is dedicated to get know actual solution of scanning of timber loads on trucks during taking wood in company MONDI SCP a.s. Process of speed of truck during its pass along a scanning gate was calculated from data acquired by scanning of the truck's surface. A realistic 3D model of the truck was created by using information of its process of speed. Last part of the thesis is dedicated to segregation of timber load transported on the truck from the truck itself.

OBSAH

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

°C	jednotka teploty Celsia
Ø	značka geometrického průměru
2D	dvourozměrný (Two Dimensional)
3D	třírozměrný (Three Dimensional)
dB	logaritmická jednotka decibel udávající podíl dvou hodnot
Ethernet	souhrn technologie počítačový sítí LAN, MAN
GPU	grafický procesor (Graphic Processing Unit)
Hz	jednotka frekvence hertz
IP adresa	identifikuje síťové rozhraní v počítačové síti (Internet Protocol address)
IR	infračervené záření (InfraRed)
kg	jednotka hmotnosti kilogram
km/h	jednotka rychlosti kilometr za hodinu
LED	dioda emitující světlo (Light Emitting Diode)
lux	jednotka intenzity osvětlení lux
mA	jednotka elektrického proudu miliampér
Mb/s	jednotka rychlosti toku informací v elektronice megabit za sekundu
mm	jednotka délky milimetr
P	pixel
PC	osobní počítač (Personal Computer)
RS-232	sériová linka
RS-422	sériová linka využívající rozdíl potenciálu mezi vodiči
V	jednotka elektrického napětí volt

1 ÚVOD

Mezi nejdůležitější fáze průmyslové výroby bezpochybně patří výstupní nebo průběžná kontrola. Automatizace průběžné a výstupní kontroly patří mezi nejžádanější a zároveň nejnáročnější systémy průmyslové automatizace. Nejdůležitější částí těchto systému je senzorický podsystém, jenž musí zajistit podrobné a komplexní informace o kontrolovaném produktu tak aby bylo možné provést kontrolu současnými technickými prostředky.

Náročnost automatizace kontroly tvarů a velikostí objektů, závisí na tvarové komplikovanosti těchto objektů. Pro kontrolu jednoduchých tvarů jsou často používány přípravky s kontaktními měřidly, do kterých jsou výrobky automaticky přemisťovány a poté proměřeny. Kdežto tvarově složité objekty je výhodné převézt do počítačových 3D modelů, na kterých je vykonána kontrola pomocí speciálního softwaru. Přesnost a spolehlivost řešení závisí na přesnosti a spolehlivosti počítačového 3D modelu.

Skenovaní objektů do 3D počítačového modelu může být provedeno rozmanitými metodami za pomocí řady prostředku a zařízení. Ke skenování objektů mohou být použity kontaktní metody, využívané u přesných souřadnicových měřicích strojů. Rovněž mohou být užity bezkontaktní metody používající různé optické metody. K takovým metodám se řadí například metody stereometrického zpracováni obrazu, metody laserových skenovacích systémů, a metody holografického snímaní objektů.

Výběr skenovací metody závisí na mnoha aspektech, jako například cena měřicího zařízení, velikosti snímaných objektů, požadované přesnosti skenování a rychlosti snímaní.

V současnosti jsou s oblibou využívány 2D laserové skenery, které se ve třetí ose pohybují podél skenovaného objektu, nebo se pohybuje skenovaný objekt. Předností tohoto řešení je relativně snadná implementace, nízká cena, široká nabídka všelijakých typů skenerů s různými přednostmi a rozsahy.

2 AKTUÁLNÍ ŘEŠENÍ SKENOVÁNÍ NÁKLADU DŘEVA

Společnost MONDI SCP, a.s. v Ružomberoku se zaměřuje na výrobu papíru a celulózy. Jedná se o největší papírny na Slovensku.

2.1 Historie výroby papíru v Ružomberoku

První zmínka o ruční výrobě papíru v okolí Ružomberka pochází již z roku 1697. Průmyslovou výrobu papíru v Ružomberoku odstartovala společnost Jakub Klein & Co. v roce 1880 v lokalitě Solo. Roku 1906 byl založen Maďarský papírenský podnik v lokalitě Supra na území města Ružomberok. Sloučení těchto dvou fabrik do jedné velké společnosti s názvem Severoslovenské Celulózky a Papierne proběhlo v roce 1958. Současné jméno společnosti, MONDI SCP, a.s. bylo stanoveno v roce 2008. (1)

2.2 Přejímka dřeva do podniku

Základní surovina pro výrobu papíru je dřevo. Surové dřevo je do podniku přiváženo v kamionech. Z důvodu bezpečnosti je každý náklad dřeva před vstupem do firmy skenován.

Kamion s nákladem dřeva přijíždějící do společnosti MONDI SCP, a.s. je zaznamenán na příjezdové silnici pomocí tří kamer detekujících pohyb. Pokud kamery zjistí přítomnost kamionu, je vyslán signál ke skenerům umístěných na vjezdové bráně a začne samotné skenování kamionu. Doba skenování je pevně určená a probíhá 18 vteřin. Řidič kamionu musí dodržovat stanovenou rychlost, která činí 5 – 10 km/h. Jestliže by kamion jel příliš pomalu, nestihla by se naskenovat celá jeho plocha. V opačném případě by došlo k tomu, že získáme malou hustotu vzorků a nešlo by s přesností určit povrch kamionu a nákladu dřeva.

Naměřená data získaná skenováním povrchu kamionu jsou následně upravená a analyzovaná, zdali dřevo uložené na přívěsu kamionu splňuje požadované bezpečnostní předpisy, či nikoliv. Kontroluje se, zdali výška klanic převyšuje výšku uloženého dřeva dle normy, jestli dřevo nevyčnívá z prostoru vyznačeného klanicemi, nebo jestli je dřevo správně upevněno na návěsu pomocí popruhů. V případě pozitivní kontroly nákladu je kamion vpuštěn do areálu, kde je následně vyložen. V opačném případě je kamion odeslán zpět k přeložení nákladu.

Jestliže kamion projde úspěšně vstupní kontrolou, pokračuje svou cestu do areálu podniku. Za vjezdem do areálu podniku se nachází váha, na kterou kamion najede a tím

se zjistí hmotnost naloženého kamionu. Po vyložení nákladu je kamion opět zvážen, a z rozdílů hmotností se určí hmotnost dopraveného dřeva.

2.3 Detektor pohybu

Kamery umístěné na kontrolní bráně jsou natočeny ve směru příjezdové cesty tak, aby detekovaly přítomnost kamionu několik desítek metrů před tím, než kamion dorazí k samotné bráně.

Program detekující přítomnost kamionu pracuje na principu toho, že obraz prázdné příjezdové cesty nasnímaný kamerami je uložen na pozadí. Každých pár sekund jsou porovnávány současné obrazy z kamer s uloženým pozadím. Porovnávání snímků je prováděno pixel po pixelu, a vyhodnocuje se, jestli před kamerami projel objekt velikosti kamionu. Kamery vyšlou signál pro zahájení skenování pouze v případě detekování velkého objektu. Tímto je zabráněno spuštění skenovacího cyklu v případě pohybu chodce nebo jiných menších objektů před kamerami. Program k detekci pohybu před vjezdovou bránou je rovněž naprogramován tak aby kamery nevyslaly signál ke zpuštění skenovací sekvence při změně osvětlení v průběhu dne nebo při vzniknutí velkého stínu před kamerami, který může vzniknout například v důsledku zamračení oblohy.

Obr. 2.1 Detail umístění jedné z kamer

2.4 Skenery umístěné na vjezdové bráně

Po detekování přítomnosti kamionu je zpuštěna skenovací sekvence, prostřednictvím čtyř skenerů umístěných na speciální konstrukci, jež se nachází při vjezdu do podniku. Tato konstrukce je zobrazena na obrázku 2.2 a představuje jakousi vstupní bránu do podniku.

Tři skenery snímající příčný průřez naloženého kamionu jsou umístěné po obvodu konstrukce. Po stranách brány jsou umístěné dva skenery, přičemž každý z nich snímá kamion z jiné strany. Třetí skener je umístěn na horní části brány a snímá kamion seshora. Společná skenovací rovina kolmá na směr pohybu kamionu je tvořena z výsečí těchto tří skenerů, tak aby snímala náklad kamionu současně ze tří stran.

Skenovací rovina čtvrtého skeneru je rovnoběžná s vozovkou a snímá podélný profil kamionu projíždějícího vstupní bránou. Díky změnám tohoto profilu se vyhodnocuje jak ujetá vzdálenost kamionu, tak jeho rychlost.

Obr. 2.2 Skenovací brána

3 HARDWARE

K detekci přítomností kamionu na příjezdové silnici je využívaná kamera CANTONK KIP300CK60A. Samotné skenování kamionů s nákladem dřeva realizují čtyři laserové skenery LMS 400 umístěné na vstupní bráně.

3.1 Laserový skener LMS 400

Optoelektrický senzor LMS 400 snímá dvourozměrně měřicí prostor pomocí laseru. Toto měřící zařízení nevyžaduje reflexní prvky ani značky na měřeném objektu, protože se jedná o aktivní systém s červeným laserem. Rovněž není nezbytné, aby byl měřený objekt osvětlen během samotného měření. (2)

Obr. 3.1 Laserový skener LMS 400 (2)

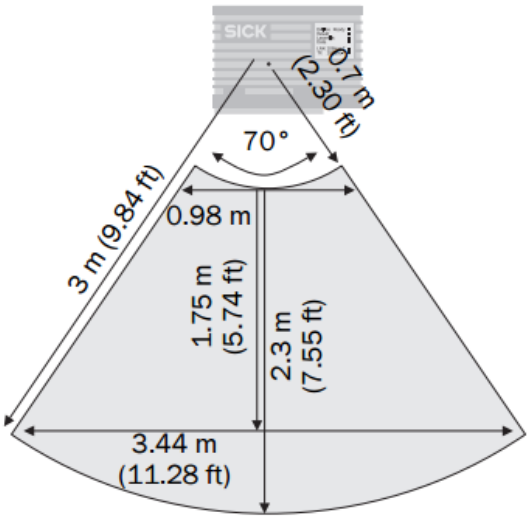
Měření pomocí senzoru LMS 400 je postaveno na principu fázového posunu. Fázový posun mezi vyslaným a přijímaným paprskem je převeden na frekvenci, ze které následně LMS 400 určuje vzdálenost měřeného objektu od nulového bodu. (2)

Pomocí rozhraní Ethernet jsou naměřené hodnoty odesílány v reálném čase do PC, kde probíhá další zpracování naměřených dat.

Tab. 3.1 Základní vlastnosti LMS 400

Délka	**179 *mm***
Šířka	**129,9 *mm***
Výška	**106,7 *mm***
Hmotnost	**2,3 *kg***
Napájecí napětí	**Stejnosměrné napětí 24 *V* ±15%**
Teplota okolí během provozu	**Od 0 °*C* do +40 °*C***
Vlhkost vzduchu	**max. 90%, nekondenzující**
Maximální okolní osvětlení	**2000 *lux***
Systematická chyba měření	**±4 *mm***
Statická chyba měření	**±3 *mm***
Skenovací frekvence	**180 – 500 *Hz***
Spínací vstupy	**4x digitální vstupy (U = 24 *V*, f_{max} = 10 *kHz*)**
Spínací výstupy	**4x digitální (I_{max} = 400 *mA*)** **1x analogový (I = 4 *mA* – 20 *mA*)**
Rozhraní	**1x RS-232, 1x RS-422, 1x Ethernet**

Vzdálenost mezi měřicím objektem a nulovým bodem senzoru LMS 400, musí být minimálně 0,7 m. Přičemž maximální měřicí rozsah činí 3 m. Během měření mimo stanovený rozsah, již není zajištěna statická chyba měření, která nabývá hodnoty ±3 mm. Pro zpřesnění měření se používají různé filtry. Pracovní prostor tohoto laserového skeneru je vyhrazen úhlem 70°. Měřicí plocha je přehledně znázorněna na obr. 3.2. (2)



Obr. 3.2 Pracovní plocha laseru LMS 400 (2)

3.2 Kamera CANTONK KIP300CK60A

Tato kamera byla zřejmě vybrána z důvodu její vysoké odolnosti vůči přírodním vlivům. Další výhodou této kamery je noční vidění a vysoké rozlišení snímaného obrazu.

Obr. 3.3 Venkovní IP kamera KIP-300CK60A (3)

Tato venkovní kamera je vybavena sedmdesáti dvěma infračervenými LED diodami s dosahem až 60 m. Tyto IR LED diody jsou automaticky sepnuty při intenzitě světla pod 10 Luxů. (3)

Tab. 3.2 Základní vlastnosti KIP-300CK60A

Délka	**295 *mm***
Šířka	**120 *mm***
Výška	**103 *mm***
Hmotnost	**1,7 *kg***
Napájení	**Stejnosměrné napětí 12 *V* ±10% nebo Ethernet**
Teplota okolí během provozu	**Od -10 °*C* do +50 °*C***
Vlhkost vzduchu	**max. 95%**
Rozlišení	**1080 *P***
Rozložení pixelů	**2048 x 1536**
Citlivost	**0,01 *Lux***
Odstup signálu a šumu	**> 50 *dB***

4 OVLADAČE PRO SKENERY SICK LMS 400

Konfigurace laserových skenerů LMS 400 pomocí komunikačního protokolu je dosti složitá. Proto byly naprogramovány ovladače vedoucím diplomové práce Ing. Davidem Fojtíkem, Ph.D v programovacím jazyce C#. Tyto ovladače umožňují jednoduchou konfiguraci skenerů v .NET prostředí. Hlavní výhodou vytvořených ovladačů je možnost synchronizovaného skenování objektů pomocí více skenerů LMS 400, které jsou umístěné v různých polohách od měřeného objektu a zároveň mohou být skenery natočené v různých úhlech vzhledem k měřenému objektu. Tímto je zvýšeno procento povrchu objektu, které je možno nasnímat.

Stručný souhrn tříd, viz tabulka 4.1, obsažených v knihovně ovladačů vytvořených pro synchronizované měření pomocí sady laserových skenerů SICK LMS 400. Všechny třídy jsou podrobně popsány v jednotlivých podkapitolách.

Třídy

Tab. 4.1 Třídy využívané ovladačem

Název třídy	Popis
CLMS400	Třída ovladače skeneru SICK LMS400
CScan	Třída popisující jeden sken
CSynchControl	Třída popisující ovládací jednotku
CLMS400List	Třída popisující skupinu použitých skenerů

Práce s ovladači se neobejde bez znalosti výčtových typů. Většina z nich je potřeba k nastavení samotných skenerů. Další výčtové typy jsou zapotřebí ke zpracování naměřených dat. Kompletní seznam výčtových typů včetně jejích položek se nachází v tabulce 4.2.

Výčty

Tab. 4.2 Výčtové typy využívané ovladačem

Název výčtového typu		Popis výčtového typu
Název položky	Hodnota	Popis položky
ImsEventType		Druh předávání naskenovaných dat
EachScan	0x00	Zasílá se každý sken samostatně
GroupScans	0x01	Zasílá se množina n-skenů v jednom balíčku
SynchGroupScans	0x02	Zasílá se množina synchronizovaných skenů v jednom balíčku
ImsCoordinateSystem		Souřadnicový systém
PolarCoordinates	0x0	Polární souřadnicový systém
CartesianCoordinates	0x1	Kartézský souřadnicový systém
ImsDataFormat		Druh skenovaných dat
DistanceRemission	0x20	Vzdálenost a remise
DistanceOnly	0x21	Pouze vzdálenost
RemissionOnly	0x22	Pouze remise
ImsUserLevel		Úroveň oprávnění
MaintenancePersonnel	0x02	Personál obsluhy
AuthorisedClient	0x03	Oprávněný klient
Service	0x04	Servis
ImsTriggerSource		Zdroj spouště skeneru
input_1	0x00	Vstup 1
input_3	0x02	Vstup 3
software_trigger	0x04	Použití softwarového trigeru
CAN_BUS	0x05	Sběrnice
input_1_AND_3	0x06	Vstup 1 a vstup 3
input_1_OR_3	0x07	Vstup 1 nebo vstup 3
test_trigger	0x08	Testovací triger
Master	0x09	Master
None	0x0F	Bez použití trigeru
ImsIOBase		Způsob definování parametrů opoždění signálů
timeBased	0x00	V závislosti na čase
distanceBased	0x01	V závislosti na vzdálenosti
ImsLogic		Způsob aktivace logického vstupu
activeLow	0x00	Aktivní 0
activeHigh	0x01	Aktivní 1
ImsDigitalOutputNumber		Digitální vstup skeneru
digitalOut1	0x01	Digitální výstup 1
digitalOut2	0x02	Digitální výstup 2
digitalOut3	0x03	Digitální výstup 3
digitalOut4	0x04	Digitální výstup 4

Název výčtového typu		Popis výčtového typu
Název položky	Hodnota	Popis položky
lmsEncoderType		Metoda vyhodnocování enkodéru
NoEncoder	0x00	Bez enkodéru
DIn2	0x01	Din2
PhaseDIn2divDIn4	0x02	Fáze
LevelDIn2divDIn4	0x03	Úroveň
ConstantVelocity	0x04	Konstantní rychlost
lmsLaserControl		Způsob řízení laseru
deactivated	0x00	Neaktivní
ownSource	0x01	Vlastní zdroj
gateControlled	0x02	Řízeno bránou
lmsFilterType		Typ filtru
medianFilter	0x01	Vytváří medián z naměřených hodnot
edgeFilter	0x02	Filtruje hodnoty výrazně odlišné od sousedících
rangeFilter	0x04	Filtruje všechny hodnoty menší než určitá vzdálenost
meanFilter	0x08	Průměrování hodnot naměřených vzdáleností

4.1 Třída CLMS400

Jádro ovladačů tvoří objekty třídy *CLMS400* reprezentující jednotlivé skenery SICK LMS 400. Třída obsahuje metody sloužící k nastavení skeneru, komunikaci se skenerem, příjmu naskenovaných dat, případně k filtraci naměřených hodnot a jejích transformací do vybrané souřadnicové soustavy.

Konstruktory

Tab. 4.3 Konstruktory třídy CLMS400

Název	**Popis**
CLMS400(int, string, int)	Vytvoří skener obsahující pouze informace potřebné ke komunikaci (IP a TCP).
CLMS400(int, string, int, lmsCoordinateSystem, double, double, double)	Vytvoří skener s informacemi o jeho reálné poloze, úhlu natočení, souřadnicovém systému a informacemi potřebnými ke komunikaci.

Destruktory

Tab. 4.4 Destruktory třídy CLMS400

Název	Popis
~CLMS400()	Skener přestane měřit data.

Vlastnosti

Tab. 4.5 Vlastnosti třídy CLMS400

Název	Typ	Popis
AngularStepWidth	UInt16	Získá úhel kroku měření.
BulkScanCount	UInt16	Získá nebo nastaví počet skenů v každé zprávě.
DataLenght	uint	Získá délku paketu v bajtech.
DigitalInputsSynchPulseFrom	UInt16	Získá nebo nastaví počáteční fázi digitálních vstupů pro synchronizací pulsem.
DigitalInputsSynchPulseTo	UInt16	Získá nebo nastaví koncovou fázi digitálních vstupů pro synchronizací pulsem.
DistanceScaling	Uint16	Získá stupňování hodnot vzdáleností. Hodnoty vzdáleností budou násobené tímto faktorem.
EventType	lmsEventType	Získá nebo nastaví typ události zasílající skeny.
FormatMeasuredValues	lmsDataFormat	Získá nebo nastaví definici obsahu a velikostí zprávy s naměřenými daty.
IPEPoint	IPEndPoint	Získá IP adresu a port skeneru LMS400.
IsConnected	bool	Zjistí, zdali je skener připojen či nikoliv.
IsReadingProcessActivated	bool	Zjistí, zdali probíhá proces čtení dat či nikoliv.
LastDigitalInputs	UInt16	Získá poslední přijatý status o digitálních vstupech.
LastErrorCode	UInt32	Získá poslední přijatý chybový kód.
NumberMeasuredValues	UInt16	Získá počet naměřených hodnot ve zprávě.
RemissionEndValue	UInt16	Získá dolní nebo horní limit remise v %.
RemissionScaling	UInt16	Získá stupňování hodnot remise. Hodnoty remise budou násobeny tímto faktorem.
RemissionStartValue	UInt16	Získá dolní nebo horní limit remise v %.
ScannerID	int	Získá identifikátor skeneru.
ScannerPositionAngle	double	Získá úhel natočení skeneru.
ScannerPositionX	double	Získá souřadnici X skeneru.
ScannerPositionY	double	Získá souřadnici Y skeneru.
ScanningFrequency	UInt16	Získá skenovací frekvenci v *Hz*.
StartingAngle	int	Získá počáteční úhel měřícího paprsku.
StepScanCounter	int	Získá krok skenu.

Metody

Tab. 4.6 Metody třídy CLMS400

Název	Návratový typ	Popis
BeginMeasuringByTriger()	bool	Připraví proces měření spouštěného prostřednictvím trigru. Vlastní měření se pak spustí trigrem, který je nutné nakonfigurovat předem prostřednictvím metody *SetGate.*
Connect()	bool	Naváže spojení se skenerem.
CreateMessagePacket(string)	Byte[]	Vytvoří paket ve formě pole bajtů z řetězce.
Disconnect()	bool	Rozváže spojení se skenerem.
DoReadGroupScans()	void	Hlavní vlákno procesu nepřetržitého čtení skenovaných dat a zasílání dat ve skupinách.
DoReadScan()	void	Hlavní vlákno procesu nepřetržitého čtení skenovaných dat.
DoReadSynchGroupScans()	void	Hlavní vlákno procesu nepřetržitého čtení skenovaných dat včetně synchronizace a zasílání dat ve skupinách.
GetOneScan()	EventArgsMeasuredData	Provede jeden scan za použití softwarového trigrování.
GetOneScanBySwTriger()	EventArgsMeasuredData	Provede jeden scan za použití softwarového trigrování.
LoadConfiguration()	bool	Načte aktuální konfiguraci ze skeneru.
LogIn(lmsUserLevel, uint)	bool	Provede přihlášení na požadovanou úroveň přístupu.
LogOut()	bool	Provede odhlášení skeneru z vyšší úrovně.
ReceiveAnswer(string, string, out uint)	bool	Načte odpověď a zkontroluje jej se vzorem a vrátí hodnotu odpovědi.
ReceiveConfirmation(string)	bool	Načte potvrzující zprávu ze skeneru a porovná ji se vzorem.
SaveParameters()	bool	[1)] Trvale uloží aktuální konfiguraci skeneru.
ScannDataToCartesianCoordinates(BinaryReader, out float[,], out float[])	bool	Převede vyhrazenou část paketu naměřených dat ve formě *BinaryReade*r na dvojrozměrné pole *scandata* v kartézských souřadnicích a pole *scanremissions*. Souřadnice jsou přepočítány do specifického souřadnicového systému podle úhlu natočení a souřadnic skeneru.
ScannDataToPolarCoordinates(BinaryReader, out float[,], out float[])	bool	Převede vyhrazenou část paketu naměřených dat ve formě *BinaryReader* na dvojrozměrné pole *scandata* v polárních souřadnicích a pole *scanremissions*.
SelectFilter(lmsFilterType)	bool	Aktivuje vybrané filtry.
SendRequest(string)	bool	Odešle požadavek skeneru.
SetAnalogOutput(UInt16)	bool	[1)] Nastaví analogový výstup. V případě chyby nastaví *ErrorCode*.

Název	Návratový typ	Popis
SetDigitalInputs(Int16 , UInt16, lmsLogic, Int16 , UInt16, lmsLogic, Int16 , UInt16, lmsLogic, Int16 , UInt16, lmsLogic)	bool	[1)] Nastaví podmínky aktivace digitálních vstupů.
SetDigitalInputs(Int16, UInt16, lmsLogic)	bool	[1)] Nastaví podmínky aktivace všech digitálních vstupů stejně.
SetDigitalOutput(lmsDigitalOutputNumber, bool)	bool	[1)] Nastaví digitální výstup. V případě chyby nastaví *ErrorCode*.
SetEncoder(lmsEncoderType)	bool	[1)] Nastaví typ a připojení enkodéru.
SetGate(lmsTriggerSource, UInt16, UInt16, UInt16, UInt16)	bool	[1)] Provede konfiguraci brány skeneru s určením zdroje trigru a jeho parametrů pro případ stejné konfigurace jak pro spuštění, tak pro ukončení měření pomocí trigru.
SetGate(lmsTriggerSource, UInt16, UInt16, UInt16, UInt16, lmsTriggerSource, UInt16, UInt16, UInt16, UInt16)	bool	[1)] Provede konfiguraci brány skeneru s určením zdroje trigru a jeho parametrů pro případ jiné konfigurace trigru pro start měření a konfigurace trigru pro ukončení měření.
SetIOBase(lmsIOBase)	bool	[1)] Zvolí typ jednotek/parametrů, které se použijí při konfiguraci binárních vstupů de-bounce, zpoždění (delay) a prosloužení (expansion) při konfiguraci brány.
SetLaserControl(lmsLaserControl, lmsTriggerSource, UInt16, UInt16)	bool	[1)] Nastavení řízení laseru. Používá se k zapnutí laseru prostřednictvím trigru a vypnutí po stanoveném čase nebo urazení předepsané vzdálenosti.
SetMeanFilter(UInt16)	bool	Nastaví filtr průměrování tj. počet hodnot skenů pro výpočet průměru. Pozor, snižuje se počet zasílanch skenů
SetMedianFilter()	bool	Nastaví vlastnosti pro medián filtr.
SetRangeFilter(float, float)	bool	Nastaví limity filtru rozsahu. Při měření budou akceptovány pouze hodnoty v daném rozsahu. Jinak bude vracena hodota 0.
StarMeasuring()	bool	Metoda zahájí funkci nepřetržitého skenování podle aktuální konfigurace a spustí proces čtení naskenovaných dat. Naměřené skeny objekt předává událostí *NewScanData*.
StartBySWTriger(byte)	bool	Softwarový triger. Metoda zahájí proces naměření požadovaného počtu skenů. Naměřené skeny objekt předává událostí *NewScanData*.
StartScanReader()	bool	Metoda zahájí proces kontinuálního čtení dat zasílaných skenerem.
StopMeasuring()	bool	Metoda ukončí funkci nepřetržitého skenování a proces čtení dat.
StopScanReader()	bool	Metoda ukončí proces kontinuálního čtení dat zasílaných skenerem.

Poznámky: 1) Pro úspěšné provedení příkazu je nezbytné provést přihlášení na úrovni “Authorised client”

Delegáti

Tab. 4.7 Delegáti třídy CLMS400

Název	Popis
BulkScanDataHandler(object, EventArgsBulkData)	Představuje metodu, která bude zpracovávat události NewBulkScanData.
ScanDataHandler(object, EventArgsMeasuredData)	Představuje metodu, která bude zpracovávat události NewScanData.

Události

Součástí třídy *CLMS400* jsou dvě události. Obě události odesílají data naměřená skenerem, ale každá z události ukládá naměřená data v jiném formátu.

4.1.1 Událost NewBulkScanData

Tato událost odesílá data naměřená skenerem. Argumentem událostí je objekt třídy *EventArgsBulkData*.

EventArgsBulkData

V průběhu měření je každý úspěšně naměřený sken uložen jako objekt třídy *CScan*. Hlavní vlastností třídy *EventArgsBulkData* je kolekce reprezentující skupinu skenů. Jednotlivé skeny jsou do této kolekce postupně ukládány v průběhu měření. Po dosažení zadaného počtu skenů ve skupině, jsou naměřená data odeslána. Třída *EventArgsBulkData* dědí své vlastnosti z třídy *EventArgs*.

Konstruktory

Tab. 4.8 Konstruktory třídy EventArgsBulkData

Název	Popis
EventArgsBulkData(UInt32)	Inicializuje novou instanci *EventArgsBulkData* třídy s prázdnými daty. Deklarovaná je pouze vlastnost *SynchScanCounter*.
EventArgsBulkData(List<CScan>, lmsDataFormat, double, double, double, int, UInt16, UInt32)	Inicializuje novou instanci *EventArgsBulkData* třídy. Parametry jsou všechny vlastností třídy v pořadí dle tab. 4.9.

Vlastnosti

Tab. 4.9 Vlastnosti třídy EventArgsBulkData

Název	Typ	Popis
Scans	List<CScan>	Získá nebo nastaví kolekci skenů.
ScanCounter1	UInt16	Získá číslo prvního skenu.
TelegramCounter1	UInt16	Získá číslo telegramu prvního skenu.
SystemCounter1	UInt16	Získá číslo systému prvního skenu.
ScansCount	int	Získá číslo skenu.
SynchScanCounter	UInt32	Získá číslo skenu pro synchronizaci.
ScannerPositionAngle	double	Získá úhel natočení skeneru.
ScannerPositionX	double	Získá souřadnici X skeneru.
ScannerPositionY	double	Získá souřadnici Y skeneru.
StartingAngle	double	Získá počáteční úhel měřícího paprsku.
AngularStepWidth	double	Získá úhel kroku měření.
DataFormat	lmsDataFormat	Získá formát dat.

Metody

Tab. 4.10 Metody třídy EventArgsBulkData

Název	Návratový typ	Popis
ScanValues(lmsCoordinateSystem, int)	float[,]	Vrátí dvojrozměrné pole dat zvoleného skenu ve vybraných souřadnicích.

4.1.2 Událost NewScanData

Na rozdíl od události *NewBulkScanData*, která odesílá data jednotlivých skenů po skupinách, tato událost odesílá data po každém úspěšně naměřeném skenu. Událost *NewScanData* se používá spíše k testovacím účelům. Argumentem událostí je objekt třídy *EventArgsMeasuredData*.

EventArgsMeasuredData

Třída *EventArgsMeasuredData* obsahuje vlastnosti pouze jednoho skenu. Stejně tak jak třída *EventArgsBulkData*, i tato třída dědí své vlastnosti z třídy *EventArgs*.

Konstruktory

Tab. 4.11 Konstruktory třídy EventArgsMeasuredData

Název	Popis
EventArgsMeasuredData(lmsDataFormat , lmsCoordinateSystem, float[,], float[])	Inicializuje novou instanci třídy *EventArgsMeasuredData.* Parametry jsou postupně: formát dat, souřadnicový systém, data a remise.
EventArgsMeasuredData(lmsDataFormat , lmsCoordinateSystem, float[,], float[], UInt16, UInt16, UInt16, UInt16, UInt16)	Inicializuje novou instanci třídy *EventArgsMeasuredData*.

Vlastnosti

Tab. 4.12 Vlastnosti třídy EventArgsMeasuredData

Název	Typ	Popis
DataFormat	lmsDataFormat	Získá formát dat.
CoordinateSystem	lmsCoordinateSystem	Získá souřadnicový systém.
ScanValues	float[,]	Získá dvourozměrné pole naskenovaných dat.
ScanPoints	string[]	Získá jednotlivé naměřené body ve vybraných souřadnicích.
Remisions	float[]	Získá remise naskenovaných dat.
DigitalInputs	UInt16	Získá digitální vstupy.
ScanCounter	UInt16	Získá číslo skenu.
TelegramCounter	UInt16	Získá číslo telegramu.
SystemCounter	UInt16	Získá číslo systému.

4.2 Třída CScan

Třída *CScan* popisující jeden sken je základním elementem naměřených dat. Objekty této třídy jsou generovány skenerem, reprezentovaným objektem třídy *CLMS400*, pomocí funkcí určených ke čtení dat.

Konstruktory

Tab. 4.13 Konstrukotry třídy CScan

Název	Popis
CScan(UInt16[], UInt16[], UInt16 , UInt16 , UInt16 , UInt16 , UInt16 , UInt32)	Vytvoří nový sken, obsahující naměřené hodnoty vzdálenosti a remise. Sken rovněž obsahuje informace o skeneru a informace potřebné k synchronizovanému skenování.

Vlastnosti

Tab. 4.14 Vlastností třídy CScan

Název	Typ	Popis
Distances	UInt16[]	Získá naměřené vzdálenosti.
Remisions	UInt16[]	Získá naměřené remise.
DigitalInputs	Uint16	Získá digitální vstupy skeneru.
EncoderPosition	Uint16	Získá pozici enkodéru.
ScanCounter	Uint16	Získá číslo skenu.
TelegramCounter	Uint16	Získá číslo telegramu.
SystemCounter	Uint16	Získá číslo systému.
SynchScanCounter	Uint32	Získá číslo skenu pro synchronizaci.

4.3 Třída CLMS400List

Pro úlohy jenž vyžadují skenování objektů pod různými úhly za použitím vícero laserových skenerů *SICK LMS400* je velice výhodné použít třídu *CLMS400List*, která pod sebou zastřešuje více objektů *CLMS400* třídy. Třída *CLMS400List* je jednoduše kolekcí objektů třídy *CLMS400*, které jsou přizpůsobené k synchronizovanému skenování.

Vlastnosti

Tab. 4.15 Vlastnosti třídy CLMS400List

Název	Typ	Popis
BulkScanCount	UInt16	Získá nebo nastaví počet skenů v každé zprávě.
MaxScannerDelay	UInt32	Získá nebo nastaví maximální množství skupinových skenů, které budou čekat na data z nejpomalejšího skeneru.
DigitalInputsSynchPulseFrom	UInt16	Získá nebo nastaví počáteční fázi digitálních vstupů pro synchronizací pulsem.
DigitalInputsSynchPulseTo	UInt16	Získá nebo nastaví koncovou fázi digitálních vstupů pro synchronizací pulsem.

Metody

Tab. 4.16 Metody třídy CLMS400List

Název	Návratový typ	Popis
Add(string, int)	CLMS400	Přidá skener do seznamu skenerů popsaný pouze IP adresou a TCP portem.
Add(string, int, lmsCoordinateSystem, double, double, double)	CLMS400	Přidá skener do seznamu skenerů popsaný IP adresou, TCP portem, a jeho souřadnicemi a úhlem natočení.
Clear()	bool	Smaže seznam skenerů i s jejích naměřenými daty.
Connect()	bool	Naváže spojení se všemi skenery v seznamu.
Disconnect()	bool	Rozváže spojení se všemi skenery v seznamu.
LoadConfiguration()	bool	Načte aktuální konfiguraci všech skenerů v seznamu.
StarMeasuring()	bool	Zahájí funkci nepřetržitého skenování podle aktuální konfigurace skenerů a spustí proces čtení naskenovaných dat.
StopMeasuring()	bool	Ukončí funkci nepřetržitého skenování a proces čtení dat.
BeginMeasuringByTriger()	bool	Připraví proces měření spouštěného prostřednictvím trigru.
StartScanReader()	bool	Zahájí proces kontinuálního čtení dat zasílaných skenery.
StopScanReader()	bool	Ukončí proces kontinuálního čtení dat zasílaných skenery.
StartBySWTriger(byte ScanCount = 1)	bool	Zahájí proces naměření požadovaného počtu skenů.
LogIn(lmsUserLevel, uint)	bool	Provede přihlášení u všech skenerů na požadovanou úroveň přístupu.
LogOut()	bool	Provede odhlášení skenerů z vyšší úrovně.
SetGate(lmsTriggerSource, UInt16, UInt16, UInt16, UInt16)	bool	[1)] Provede konfiguraci brány skeneru s určením zdroje trigru a jeho parametrů pro případ stejné konfigurace jak pro spuštění, tak pro ukončení měření pomocí trigru.

Konstruktory

Tab. 4.17 Konstruktory třídy CLMS400List

Název	Popis
CLMS400List()	Inicializuje prázdnou instanci třídy *CLMS400List*.
CLMS400List(Uint16, UInt32, UInt16, UInt32)	Inicializuje novou instanci třídy *CLMS400List*.

Delegáti

Tab. 4.18 Delegáti třídy CLMSList

Název	Popis
GroupBulkScanDataHandler(object , EventArgsGroupBulkData)	Představuje metodu, která bude zpracovávat události NewGroupBulkScanData.

Události

Třída *CLMS400List* obsahuje pouze jednu událost, kterou je *NewGroupBulkScanData*, jež odesílá skupiny naměřených skenů ze všech skenerů současně. Argumentem událostí je objekt třídy *EventArgsGroupBulkData*.

EventArgsGroupBulkData

Tato třída je v podstatě kolekcí objektů *EventArgsBulkData*. Třída dědí své vlastnosti ze třídy *EvantArgs*.

Konstruktory

Tab. 4.19 Konstruktory třídy EventArgsGroupBulkData

Název	Popis
EventArgsGroupBulkData(List<EventArgsBulkData>)	Inicializuje novou instanci třídy *EventArgsGroupBulkData.* Jediným parametrem je kolekce *EventArgsBulkData*.

4.4 Třída CSynchControl

Tato třída reprezentuje ovládací jednotku neboli hardwarový triger, který se stará o synchronní skenování všech laserových skenerů využitých v dané aplikaci. Triger generuje pulzy, díky kterým jsou všechny zúčastněné skeny synchronizovány.

Konstruktory

Tab. 4.20 Konstruktory třídy CSynchControl

Název	Popis
CSynchControl(string, int, byte, byte, byte, UInt16, byte, byte)	Inicializuje novou instanci třídy *CSynchControl* reprezentující ovládací jednotku. Parametry jsou informace potřebné ke komunikaci, k nastavení pulzů a k nastavení informačních bitů pro zasílání paketů.

Metody

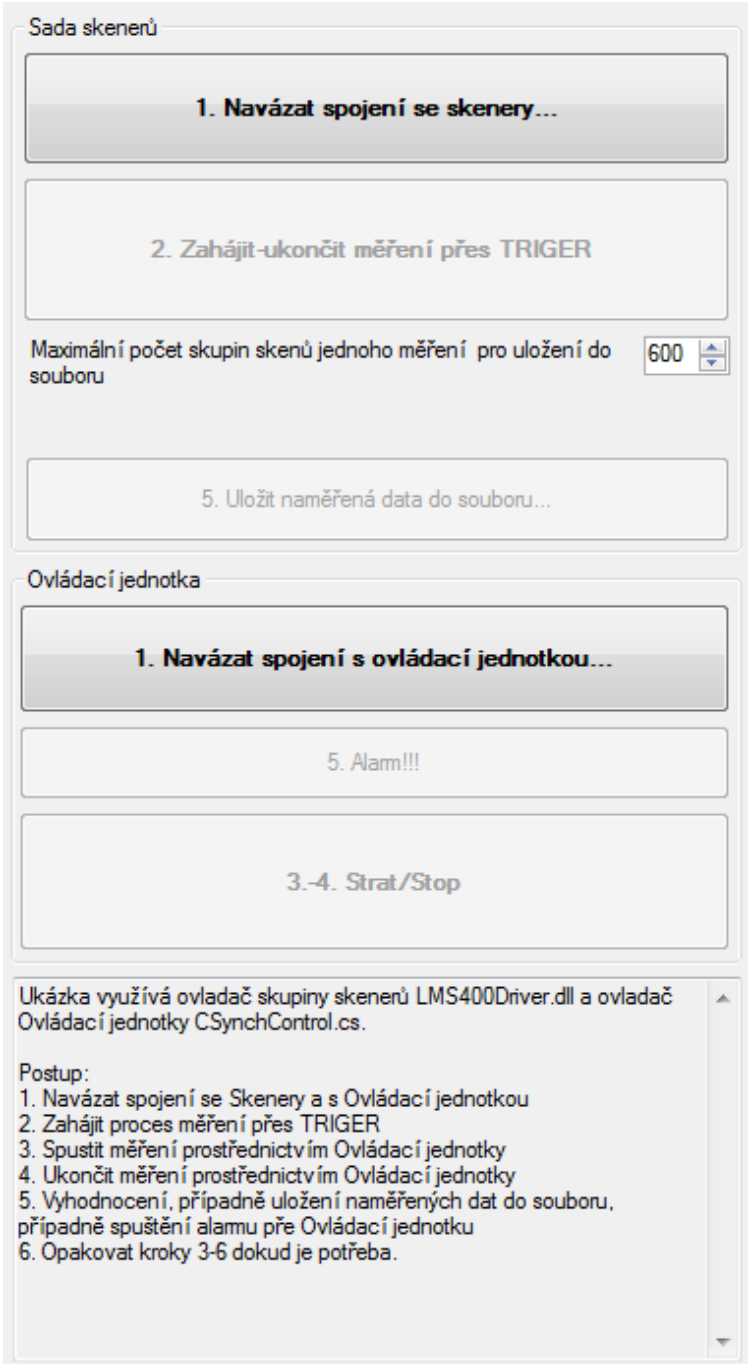
Tab. 4.21 Metody třídy CSynchControlc

Název	Návratový typ	Popis
Connect()	bool	Naváže spojení s řídicí jednotkou.
Disconnect()	bool	Rozvázaní spojení řídicího modulu.
ScannerSynchStart()	bool	Spuštění měření skenerů přes triger a jejich synchronizace pomocí pulzů.
ScannerStop()	bool	Ukončení synchronizovaného měření.
SetAlarm(bool)	bool	Nastaví alarm.
CreateMessagePacket(string)	Byte[]	Vytvoří paket s naskenovanými daty.

5 APLIKACE PRO SKENOVÁNÍ KAMIONŮ

Pro jednoduchou práci se skenery umístěnými na kontrolní bráně před vjezdem do společnosti MONDI SCP a.s. byla vytvořena uživatelská aplikace, díky které je kontrola nákladu kamionů, směřujících do podniku, mnohem jednodušší na obsluhu.

Design hlavního panelu aplikace je zobrazen na obrázku 5.1. Postup při skenování je stručně vysvětlen přímo na hlavním panelu, což snižuje nároky na obsluhu.



Obr. 5.1 Hlavní panel aplikace

Zprvu je zapotřebí deklarovat objekty a proměnné, jenž aplikace využívá ke skenování, následnému ukládaní a zpracování dat.

```
//Deklarace proměnných jednotlivých skenerů
CLMS400 _SC1 = null, _SC2 = null, _SC3 = null, _SC4 = null;
//Deklarace kolekce skenerů
CLMS400List _ScannersList;
//Deklarace ovládacího modulu
CSynchControl _ControlBox;
//pole naměřených dat vybraných skenů každého skeneru (pro vykreslování dat)
volatile float[,] _Data_SC1, _Data_SC2, _Data_SC3, _Data_SC4;
//Fronta všech skenů jednoho měření
public List<EventArgsGroupBulkData> AllMeasuredData;
```

V prvním kroku je zapotřebí navázat spojení se všemi čtyřmi skenery. Nejprve dojde k vytvoření kolekce skenerů, s definovanými parametry popsanými ve zdrojovém kódu níže, která má za úkol vytvářet balíčky po padesáti skenech z jednotlivých skenerů. Následně je zapotřebí vytvořit událost, která zasílá synchronizované balíčky ze všech čtyř skenerů. Na závěr jsou definovány jednotlivé skenery. A to jak jejích IP adresy, které jím byly předem navoleny, tak i jejích reálná poloha na měřicí bráně.

```
//Založení kolekce skenerů. Kolekce vrací skeny v balíčcích s definovanou
velikostí, volí se HW synchronizace přes digitální vstupy (realizuje ovládací
modul)
_ScannersList = new CLMS400List((UInt16)50  //Počet skenů v jednom balíku
naskenovaných dat
                                ,(UInt16)5   //Maximální počet balíčků, o který
se může skener opozdit oproti nejrychlejšímu, neodpovídá-li skener po danou
dobu pak se nahrazuje prázdnými skeny
                                ,(UInt16)5   //Hodnota DigitalInputs paketu
skenu počátečního stavu synchronizačního pulsu - data s touto hodnotou se
nezasílají
                                ,(UInt16)1); //Hodnota DigitalInputs paketů
skenu konečného stavu synchronizačního pulsu - první paket s touto hodnotou
iniciaizuje počítadlo SynchScanCounter

//Provázání skupiny s událostní rutinou, která zasílá synchronizované balíčky
ze všech skenerů současně
_ScannersList.NewGroupBulkScansData += new
CLMS400List.GroupBulkScansDataHandler(_ScannersList_NewGroupBulkScansData);

//Založení ovladačů jednotlivých skenerů a zařazení do kolekce
//1. Horní skener
_SC1 = _ScannersList.Add("192.168.0.1" //IP adresa skeneru
                        , 2111         //TCP port
                        , lmsCoordinateSystem.CartesianCoordinates //Volba
souřadnicového systému - doporučuje se Kartézská soustava souřadnic
                        , 0.0          //Pouze pro Kartézskou soustavu: Posunutí
v ose X (v mm)- Poloha osy skeneru vůči společnému souřadnicovému systému
                        , 3000.0       //Pouze pro Kartézskou soustavu:
Posunutí v ose Y (v mm) - Poloha osy skeneru vůči společnému souřadnicovému
systému
                        , 0.0);        //Pouze pro Kartézskou soustavu: natočení
(ve stupních) osy skeneru vůči společnému souřadnicovému systému
```

```
//2. Pravý skener
_SC2 = _ScannersList.Add("192.168.0.2", 2111,
lmsCoordinateSystem.CartesianCoordinates, 1500.0, 1500.0, -90.0);
//3. Levý skener
_SC3 = _ScannersList.Add("192.168.0.3", 2111,
lmsCoordinateSystem.CartesianCoordinates, -1500.0, 1500.0, +90.0);
//4. Podélný skener, je pro vizualizaci umístěn do počátku společného
souřadnicového systému který je v pomyslně uprostřed vozovky
_SC4 = _ScannersList.Add("192.168.0.4", 2111,
lmsCoordinateSystem.CartesianCoordinates, 0.0, 0.0, 180);
```

Následně je rovněž zapotřebí navázat spojení s ovládacím modulem, jež se stará o synchronizaci všech zúčastněných skenerů.

```
//Ovládací modul má 4 výstupy, z toho se používá výstup 2 jako spouštěč alarmu,
3 jako TRIGR spouštění skeneru a 4 jako synchronizační pulz (jednička se
nepoužívá)
//Modul HW trigerm spouští měření skenerů a zároveň provádí pulz, kterým se
skenery synchronizují.
_ControlBox = new CSynchControl("192.168.0.101" //IP adresa ovládacího boxu
                                ,2112           //TCP Port ovládacího boxu
                                ,0x0C           //Výběr výstupů, které budou
ovlivněny pulzem C = 1100, výstup 3 a 4
                                ,0x0C           //Počátek pulzu a strat měření C
= 1100, výstup 3 a 4 se nastaví na jedničku
                                ,0x04           //Konec pulzu C = 0100, výstup 4
zůstává a 3 se vypne
                                ,100            //Doba pulzu v ms. Po této době
se nastaví konec pulzu
                                ,0x00           //Výstup, který ukončí měření
(vyplne triger)
                                ,0x02);          //Určení výstupu, který ovládá
alarm výstup 3 (výstup 2 )
```

Po úspěšném navázání spojení se skenery a synchronizačním modulem je možno pokračovat v dalších krocích. V opačném případě je zobrazeno chybové hlášení.

Zahájení samotného procesu synchronizovaného měření a příjem dat je vykonáno pomocí ovládacího modulu.

Následně je deklarována událostní rutina, která přijímá synchronizované skupiny skenů ze všech zúčastněných skenerů.

```
//Událostní rutina příjmu synchronizovaných skupin skenů ze všech skenerů
delegate void ReceiverGroupBulkScansData(object sender, EventArgsGroupBulkData
e);    //nezbytná součást události - ovladač používá pro každý skener
samostatné vlákno
void _ScannersList_NewGroupBulkScansData(object sender, EventArgsGroupBulkData
e)
{
      //+ Meziprocesní výměna dat (nezbytné součást události na formuláři)
      zajistí zpracování vláknem formuláře
      if (this.InvokeRequired)
      {
            ReceiverGroupBulkScansData d = new
            ReceiverGroupBulkScansData(_ScannersList_NewGroupBulkScansData);
            this.BeginInvoke(d, new Object[] { sender, e });
            return;
}
```

Aktuální naměřené body v kartézských souřadnicích ze všech čtyř skenerů jsou rovněž vykreslovány na obrazovce ve 2D souřadnicích, s tím že je vykreslován vždy první sken skupiny každého skereu.

6 KNIHOVNA EmguCV

OpenCV (Open Source Computer Vision Library) je otevřená knihovna, která obsahuje několik stovek algoritmů orientovaných na strojní vidění. OpenCV má podporu pro jazyky C, C++, Python, Java, atd. OpenCV byl navržen pro výpočetní účinnost a se silným zaměřením na aplikace reálného času. Pro jazyky jako C#, VB, VC++, IronPython a různé další pracující na platformě .NET, byl navržen ekvivalent OpenCV v podobě knihovny EmguCV. (4)

Jelikož celý projekt diplomové práce, počínaje skenováním kamionů, přes úpravu dat, a konče analýzou 3D obrazu, je naprogramován v programovacím jazyce C#, byla pro práci s daty využívána knihovna EmguCV.

Knihovna EmguCV obsahuje několik desítek jmenných prostorů, přičemž každý z nich se specializuje na určité odvětví strojního vidění. Níže v této kapitole jsou popsány pouze částí EmguCV knihovny, které byly využity v projektu.

6.1 Jmenný prostor Emgu.CV

V tomto jmenném prostoru se nacházejí funkce ke zpracovávání obrazu.

Tab. 6.1 Třídy a jejich metody ze jmenného prostoru Emgu.CV

Název třídy	Popis třídy
Název funkce	Popis funkce
CvInvoke	Pomocí této třídy jsou vyvolávány OpenCV funkce
ApproxPolyDP	Aproximuje body polygonální křivkou se zadanou přesností

Pro jednoduché pozorování 3D objektů pod různými úhlý, bylo využíváno speciální okno určené pro vizualizaci objektů ve 3D prostoru, které je součástí knihovny EmguCV. Jelikož samotná EmguCV knihovna obsahuje perfektní nástroj pro vizualizaci objektů ve 3D prostoru, není zapotřebí využívat nástroje jako například devDept Eyeshot, jenž jsou zdarma k dispozici pouze na zkušební období třiceti dnů. Nástroje jako devDept Eyeshot jsou určené nejen k vizualizaci 3D objektů, ale slouží také například k analýze 3D objektů, generování povrchu objektů z mračna bodů, ukládání obrázků ze 3D objektů, a k vykonávání celé řady dalších úkonů. Pro naši aplikaci, kde si vystačíme s jednoduchým zobrazením objektů ve 3D prostoru a jejích vizuální kontrolou, jsou tyto dodatečné funkce zbytečné.

Tab. 6.2 Třídy a jejich metody určené k vykreslovaní 3D objektů ve vizualizačním oknu

Název třídy	Popis třídy
Název funkce	Popis funkce
Viz3D	Reprezentuje 3D vizualizační okno
SetBackgroundMeshLab	Nastaví barvu pozadí
ShowWidget	Přidá widget do vizualizačního okna
Spin	Vykreslí všechny widgety ve vizualizačním okně
WCloud	Tento widget definuje mračno bodů
WCoordinateSystem	Tento widget reprezentuje osy souřadnicového systému

Třída *Viz3d* dědí své vlastnosti ze třídy *UnmanagedObject*.

6.2 Jmenný prostor Emgu.CV.Structure

Tento jmenný prostor zahrnuje všechny struktury užívané při práci s EmguCV knihovnami.

Tab. 6.3 Struktury ze jmenného prostoru Emgu.CV.Structure

MCvPoint3D32f	Reprezentuje bod ve 3D prostoru, jehož souřadnice X, Y a Z jsou ve formátu float.
MCvScalar	Reprezentuje skalár se čtyřmi vlastnostmi typu double. Používá se převážně jako argument pro širokou škálu funkcí z knihovny EmguCV.

6.3 Jmenný prostor Emgu.CV.Util

Tento jmenný prostor je kolekcí užitečných nástrojů.

Tab. 6.4 Užitečné nástroje ze jmenného prostoru Emgu.CV.Util

VectorOfFloat	Třída reprezentující vektor obsahující proměnné ve formátu float.
VectorOf3DPoint3D32f	Třída reprezentující vektor obsahující 3D body, jejichž souřadnice jsou ve formátu float.
VectorOfPointF	Třída reprezentující vektor obsahující 2D body, jejichž souřadnice jsou ve formátu float.

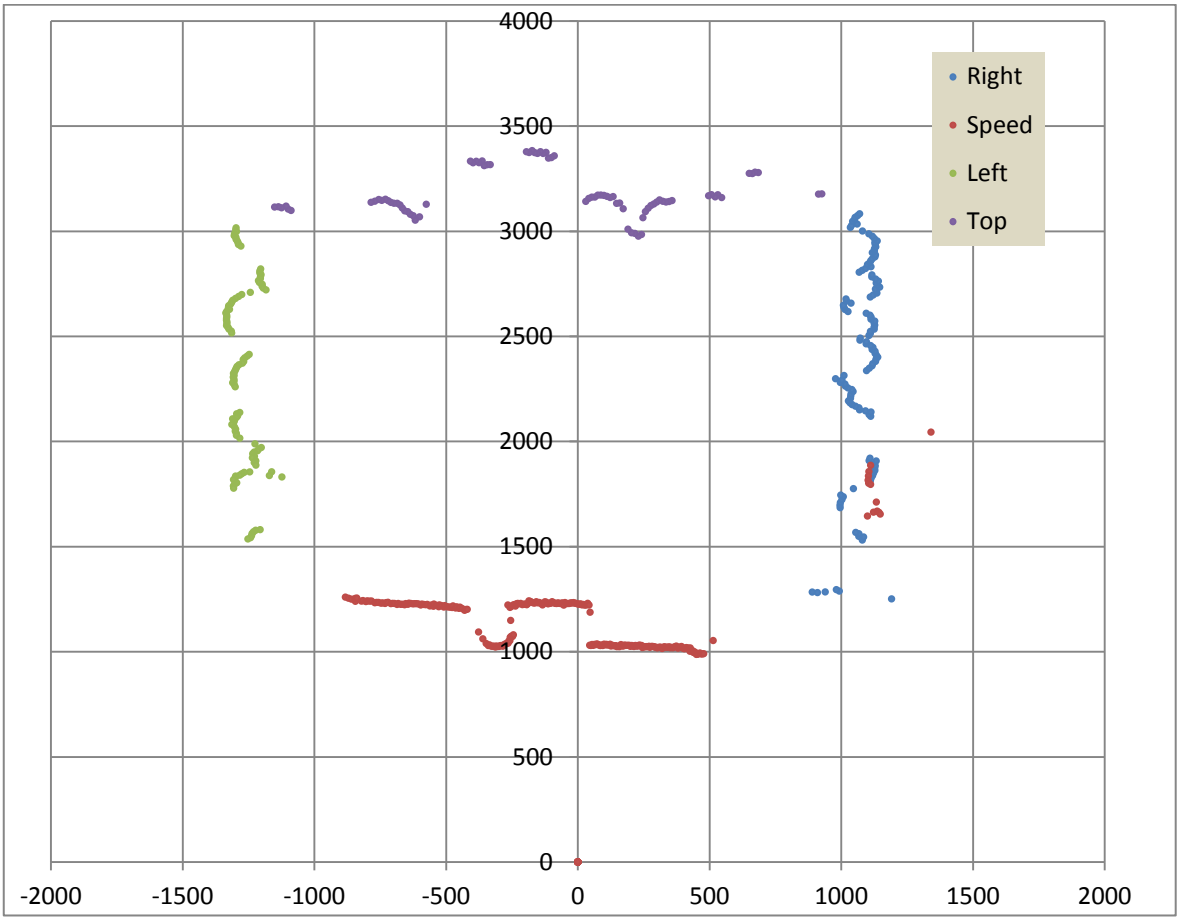
7 DATA ZE SKENERŮ

Na obrázku 7.1 se nachází kamion zachycený pomocí jedné z kamer umístěných na vstupní bráně ve společnosti MONDI SPC a.s. v Ružomberoku. Algoritmy určené k práci s 3D modelem kamionu, jež jsou popsány v následujících kapitolách, byly testovány na tomto naloženém kamionu.

Obr. 7.1 Snímek kamionu nasnímaný kamerou

Pomocí aplikace využívající ovladače určené k synchronnímu skenování za pomocí více skenerů SICK LMS 400, viz. kapitola 4 a 5, byl nasnímán povrch naloženého kamionu. Data získané pomocí tohoto skenování jsou vstupem pro praktickou část diplomové práce.

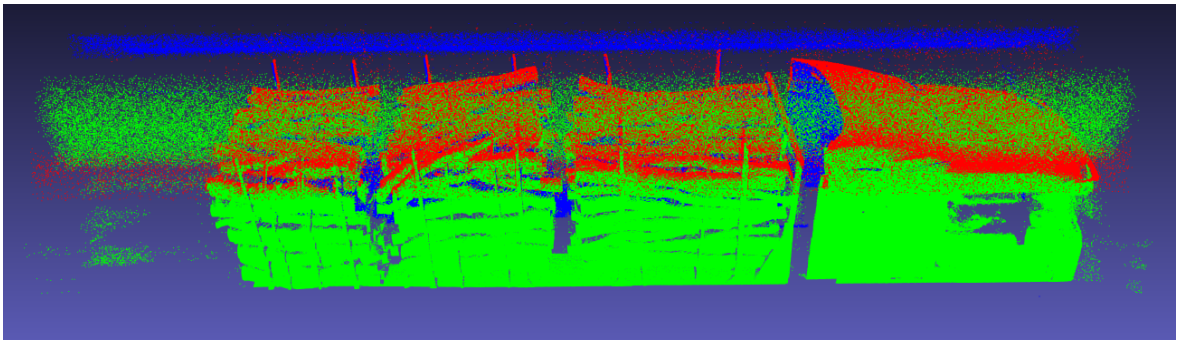
Na obrázku 7.1 lze pozorovat naměřená data náhodně vybraného skenu z levého, horního, pravého a podélného skeneru. Sken, který je vyobrazen na obrázku 7.2, byl pořízen na nákladovém prostoru kamionu s nákladem dřeva.



Obr. 7.2 Náhodně vybraný sken nákladového prostoru

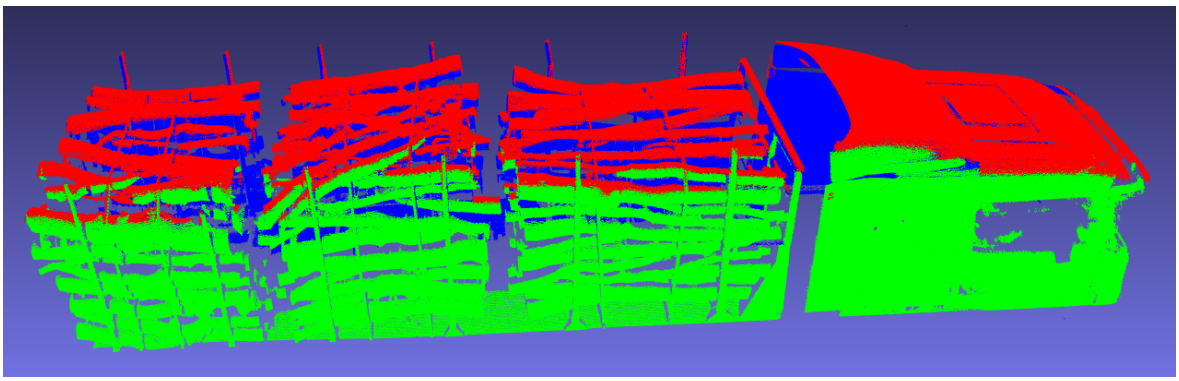
Body nasnímané pomocí levého skeneru (zelené body), horního skeneru (fialové body) a pravého skeneru (modré body) tvoří příčný řez kamionu. Kdežto body z podélného skeneru (červené body) reprezentují podélný profil kamionu. Díky změnám podélného profilu napříč skeny je možno vypočíst průběh rychlosti kamionu.

3D model kamionu byl získán postupným vkládáním všech skenů z levého, pravého a horního skeneru s konstantním odstupem vůči předchozímu skenu. Takto získaný 3D model je zobrazen na obrázku 7.3.



Obr. 7.3 3D model kamionu před filtraci

Z obrázku 7.3 je patrné, že data získané naskenováním kamionu obsahují poměrně značné množství šumu. Tento šum byl eliminován aplikací hranového filtru postupně na všech dílčích skenech pořízených pomocí tří skenerů, které snímají příčný profil kamionu.



Obr. 7.4 3D model kamionu po filtraci

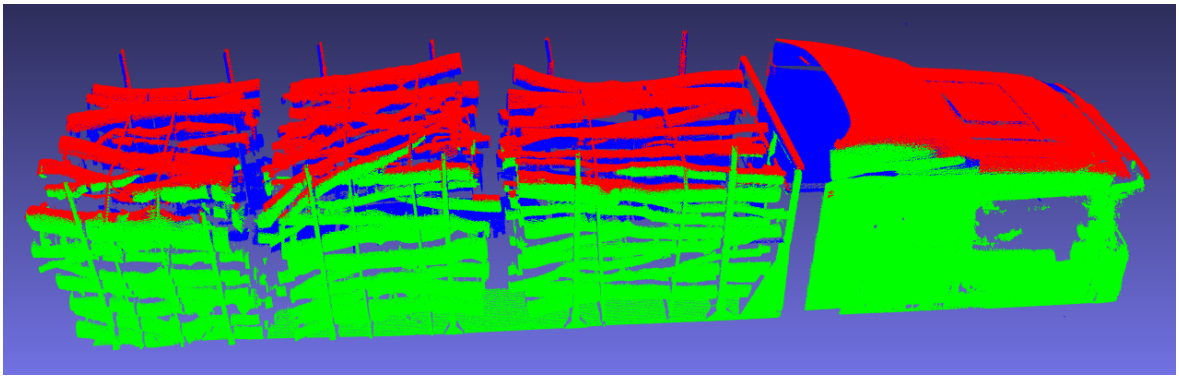
Na obdobném modelu je prováděna kontrola uložení dřeva na návěsu, která se provádí v Ružomberoku před vstupem do podniku papíren. Pro tuto kontrolu je takový 3D model kamionu dostačující. Z modelu lze jednoznačně určit, zdali výška klanic převyšuje výšku uloženého dřeva dle normy, jestli dřevo nevyčnívá z prostoru vyznačeného klanicemi, nebo jestli je dřevo správně upevněno na návěsu pomocí popruhů.

Pro tuto úlohu je ovšem takový 3D model nedostačující protože cílem tohoto projektu je separace nákladu dřeva od kamionu, aby bylo následně možno vypočítat objem převáženého dřeva.

8 REÁLISTICKÝ 3D MODEL KAMIONU

Předepsaná rychlost pro průjezd kamionu skrz skenovací bránu je 5 – 10 km/h. I přes veškerou snahu řidičů, je takřka nemožné při tak nízkých rychlostech udržet konstantní rychlost pohybu kamionu po celou dobu průjezdu bránou.

Na obrázku 8.1 je zobrazen 3D model naskenovaného kamionu s konstantními posuny mezi jednotlivými skeny. To znamená, že uvažovaná rychlost pohybu kamionu je konstantní. Tato úvaha je však chybná, a skutečná rychlost kamionu měnící se v průběhu měření, musí být vypočtena.



Obr. 8.1 3D model kamionu s konstantní vzdálenosti mezi jednotlivými skeny

Na obrázku 8.1 můžeme pozorovat, že kabina kamionu je výrazně protáhla a náklad dřeva se postupně zkracuje ve směru od kabiny. Z toho je evidentní, že řidič kamionu vjel do skenovací brány s nízkou rychlostí a skenery měly čas naskenovat kabinu s větší hustotou skenů. Následně jak kamion zrychloval, hustota skenu se snižovala.

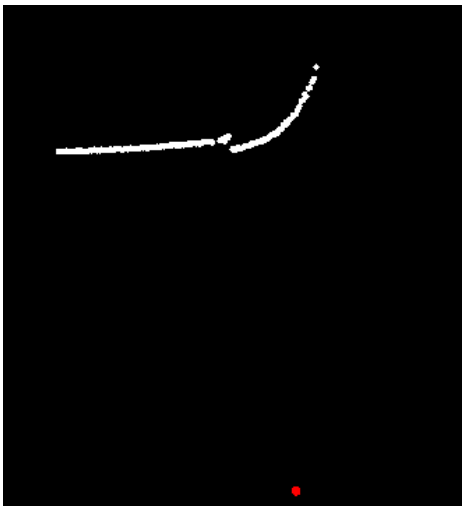
Pro získání realistického 3D modelu kamionu je zapotřebí určit rychlosti jeho pohybu v průběhu skenování.

K přesnému a jednoduchému určení průběhu rychlostí kamionu projíždějícího skrz skenovací bránu může být použit radar. Další variantou je použití laserového dálkoměru, který by byl umístěn za vstupní bránou a snímal by hodnoty vzdáleností přibližující se kabiny kamionu. Průběh rychlostí kamionu by byl vypočítán derivaci naměřených vzdáleností.

Na skenovací bráně je již instalován skener, který snímá podélný profil kamionu a jeho skenovací rovina je rovnoběžná s vozovkou. Pro určení průběhu rychlostí kamionu proto byla využita data získaná pomocí tohoto skeneru. Z rozdílu poloh kamionu mezi jednotlivými skeny podélného skeneru je určen jeho posuv, potažmo rychlost. Jelikož jsou zúčastněné skenery synchronizovány a jejích měřicí frekvence je stejná, může být tento

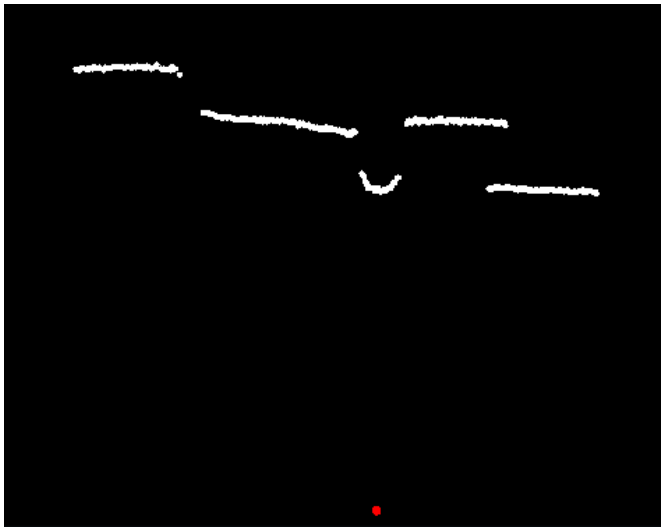
vypočtený posuv aplikován na data ze všech skenerů, a tím lze získat realistický 3D model kamionu.

Pro představu podélného profilu kamionu jsou na obrázcích s pořadovými čísly 8.2 a 8.3 zobrazeny nahodilé skeny pořízené podélným skenerem. Bíle body znázorňují podélný profil kamionu tvořený naskenovanými body. Červený bod vyobrazuje pozici skeneru.



Obr. 8.2 Náhodný sken kabiny pořízený podélným skenerem

Na obrázku 8.2 se nachází podélný profil začátku kabiny s nárazníkem, kdežto na obrázku 8.3 je vyobrazen podélný profil části nákladního prostoru kamionu. V podélném profilu nákladního prostoru jsou zřetelně vidět jednotlivé součástí nápravy kamionu, například ta zaoblená část je povrch klanice.

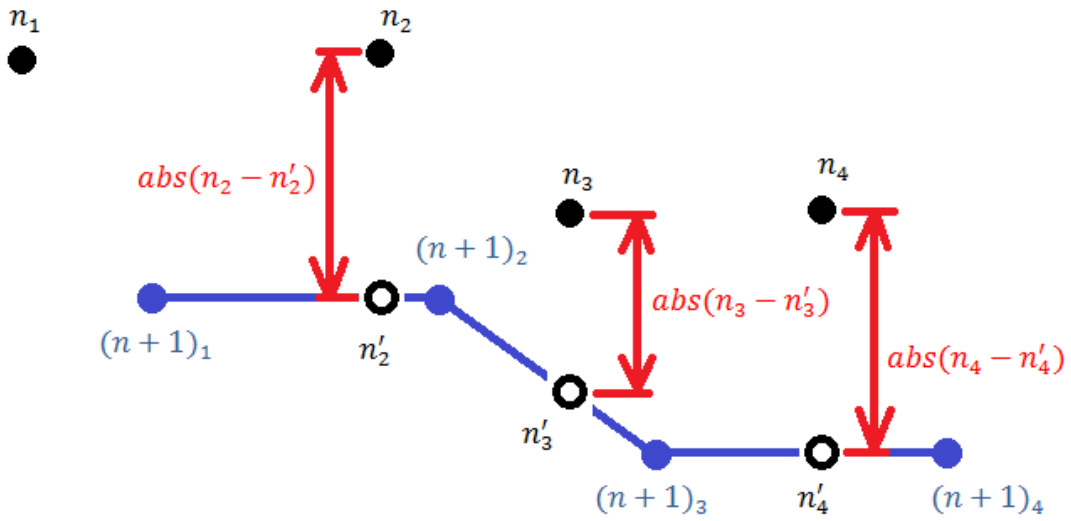


Obr. 8.3 Náhodný sken nákladního prostoru pořízený podélným skenerem

Pro určení posunu kamionu ze dvou po sobě jdoucích skenů z podélného skeneru, byly navrženy a realizovány dvě metody.

8.1 Metoda odhadu posuvu a jeho kontroly

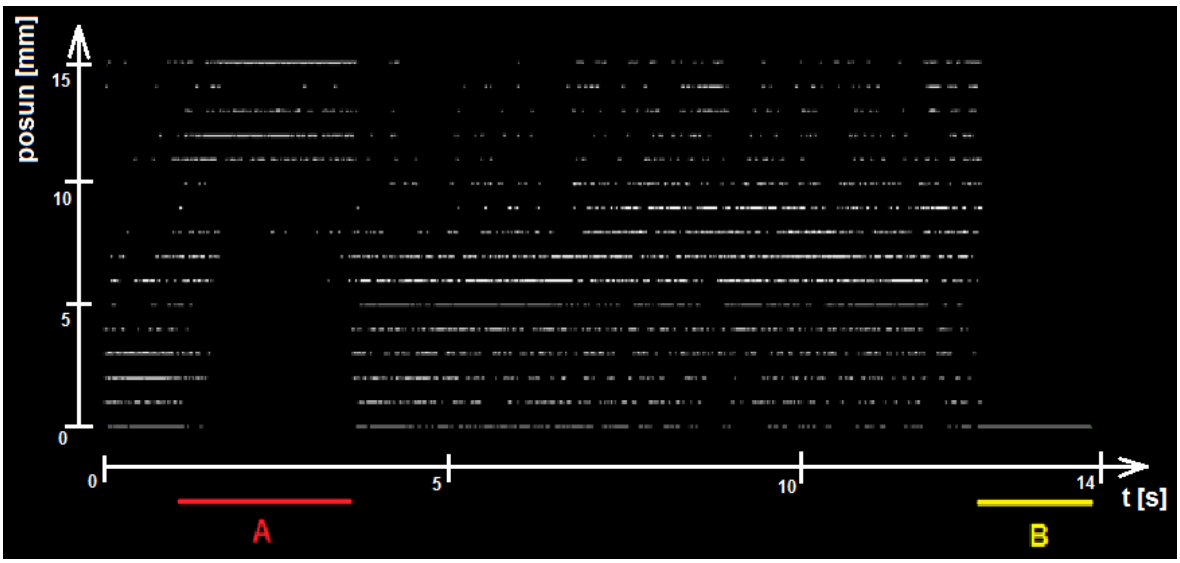
Pomocí této metody je hledán posuv mezi n-tým skenem a následujícím skenem s pořadovým číslem n+1 tak, že n-ty sken je postupné posouvám o 0 – 15 mm s krokem 1 mm. Při dodržení předepsané maximální rychlosti 10 km/h a frekvenci skenování 370 Hz by posuv kamionu mezi jednotlivými skeny neměl přesahovat 8 mm, z důvodu možnosti nedodržení těchto parametrů jsou testovány skeny s posuvem do 15 mm. Pro každý posun jsou vypočítány absolutní hodnoty rozdílů mezi body posunutého skenu n a jejich průmětem na přímce procházející dvěma body skenu n+1,které tento bod nejblíže obklopují v horizontální rovině. Pro lepší představu je tento výpočet graficky znázorněn na obrázku 8.4. Z důvodu přehlednosti jsou na obrázku skeny od sebe vertikálně vzdáleny, kdežto ve skutečnosti na sebe víc přiléhají.



Obr. 8.4 Grafické schéma výpočtu rozdílů mezi posunutým bodem a jeho průmětem

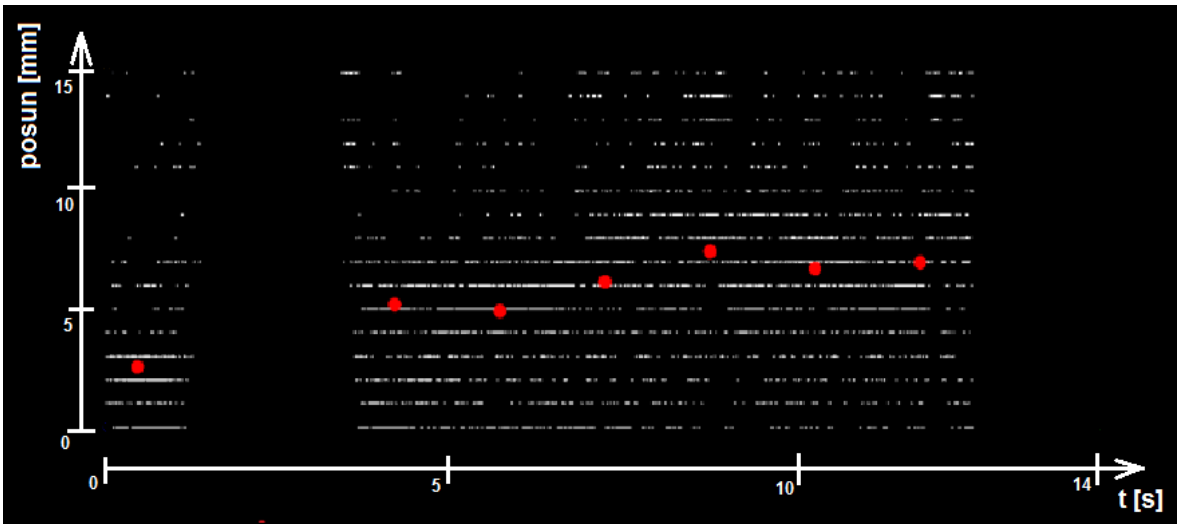
Body posunutého n skenu, které jsou mimo horizontální rozsah skenu n+1, jsou z výpočtu vyřazeny. Pro každý posun je vypočítaná průměrná hodnota ze všech absolutních hodnot, a posun s nejnižší průměrnou hodnotou je stanoven jako skutečný.

Na obrázku 8.5 můžeme pozorovat velikost odhadovaného posuvu mezi každým n-tým skenem a skenem n+1, který dosáhly největší shody. Horizontální osa znázorňuje jednotlivé dvojice sousedních skeny, respektive časový průběh skenování. Počet nasnímaných skenů je roven 5100. Při skenovací frekvenci 370 Hz lze určit čas skenovací sekvence, která je rovná necelých 14 s. Na vertikální ose jsou zaznamenány posuvy všech sousedních dvojic skenů, které dosáhly maximální shody.



Obr. 8.5 Posuvy mezi skeny získané pomocí metody odhadu posuvu a jeho kontroly

Skupina skenu s označením A je velmi problematická, jelikož v této části se nachází kabina kamionu, jejíž podélný průřez připomíná jednoduchou úsečku. Jelikož tato úsečka je mírně nakloněná, větší posuvy se zdají být přesnější, což je chyba. Skupina skenu B je oblast, ve které skenery stále ještě snímaly data, ale kamion již kompletně projel skenovací bránou a tyto skeny jsou prázdné. Obě zmíněné skupiny skenů, jsou vyřazeny z procesu hledání pár důležitých bodů, které vyjadřují průměrný posuv kamionu projíždějícího kontrolní bránou v jednotlivých úsecích. Tyto body jsou znázorněny červenou barvou na obrázku 8.6.

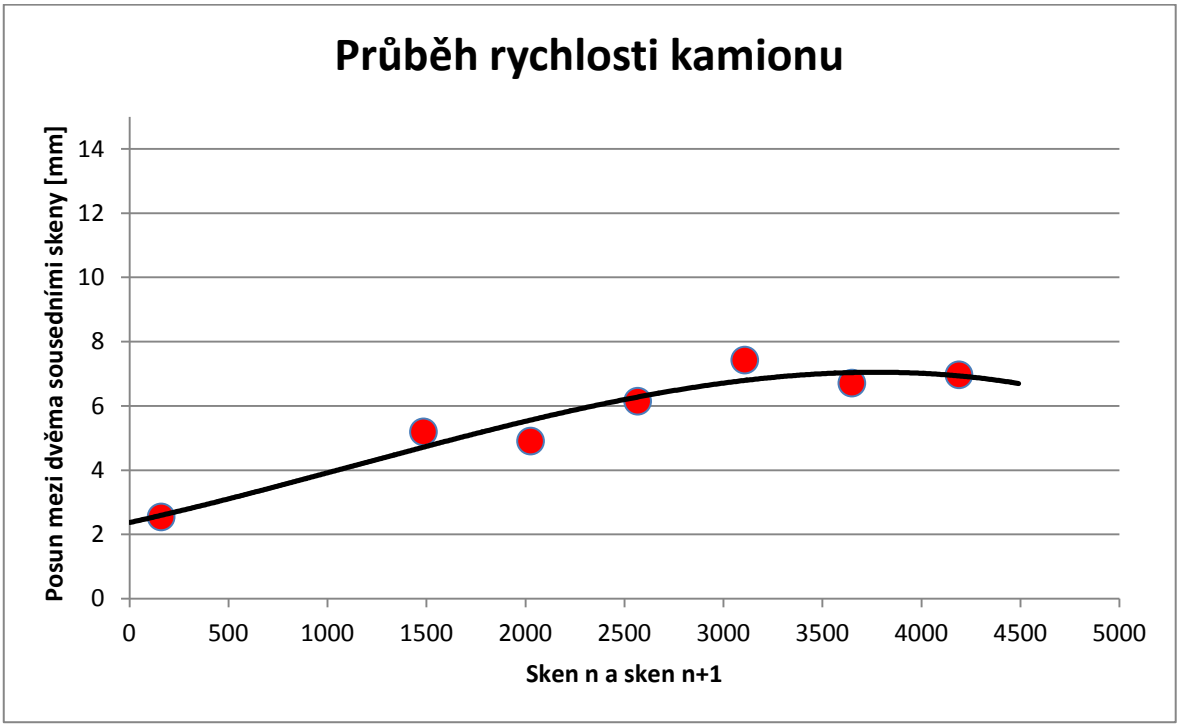


Obr. 8.6 Body pro tvorbu kubické křivky

Nalezenými body je proložená kubická křivka, získaná metodou nejmenších čtverců, a tím je získán spojitý průběh rychlosti kamionu projíždějícího pod skenovací bránou.

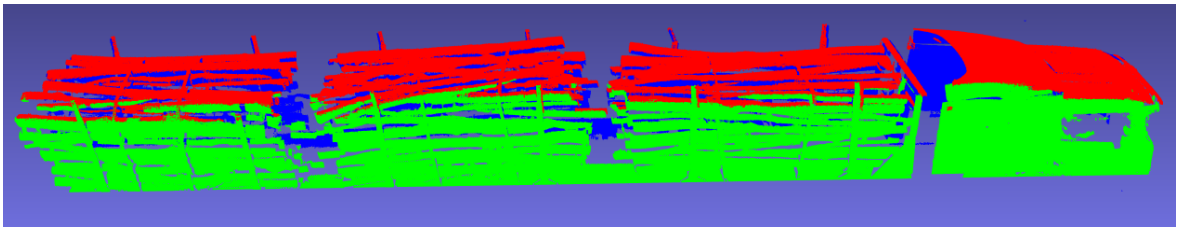
Rovnice kubické křivky vypočítaná pomocí metody odhadu posuvů a jeho kontroly:

$$y = -7{,}839 \cdot 10^{-11} \cdot x^3 + 2{,}632 \cdot 10^{-7} \cdot x^2 + 1{,}362 \cdot 10^{-03} \cdot x + 2.371$$



Obr. 8.7 Průběh rychlosti kamionu získaný pomocí metody posuvu a jeho kontroly

Pomocí této kubické křivky lze zjistit skutečný posun kamionu mezi jednotlivými skeny. Na obrázku 8.8 je zobrazen 3D model kamionu, který byl získán pomocí metody odhadu posuvu a jeho kontroly.



Obr. 8.8 3D model kamionu získaný metodou odhadu posuvu a kontroly

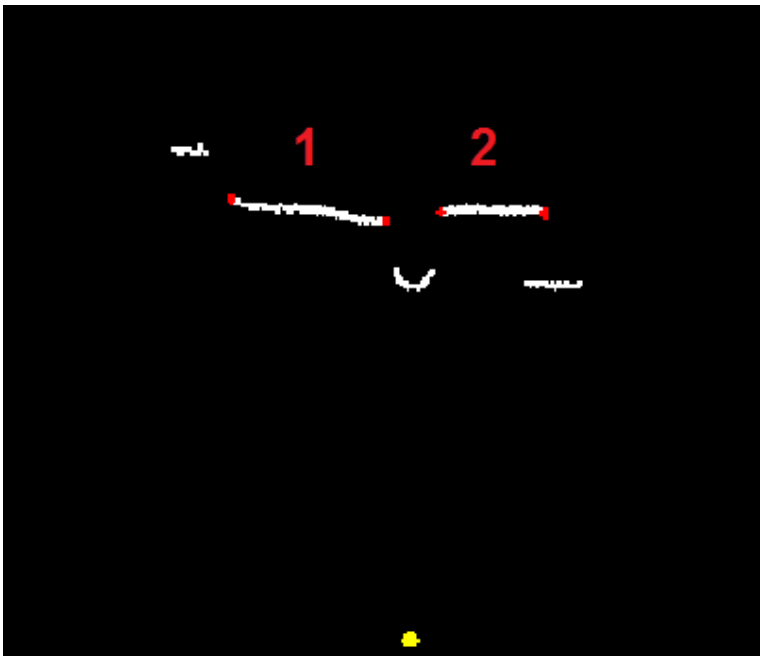
Na 3D modelu lze holým okem pozorovat, že nákladní prostor kamionu má reálné rozměry, ale kabina je stále protáhlá. Protažení kabiny je způsobeno tím, že její vodorovný profil je tvořen z větší částí úsečkou a ta má tendenci odhadovat větší posun než ve skutečnosti je.

Obecně vzato, je tato metoda přesná pro úseky kamionu s rozmanitým podélným profilem, kdežto u jednoduchých profilů vzniká chyba.

8.2 Metoda segmentů

Napříč všemi skeny pořízenými podélným skenerem, jsou vyhledány výrazné úseky kamionu, skládající se z předem určeného minimálního počtu bodů.

Ukázka nalezených segmentů se nachází na obrázku 8.9, kde červenými body jsou zvýrazněné počáteční a koncové body segmentů, a žlutý bod znázorňuje polohu skeneru.



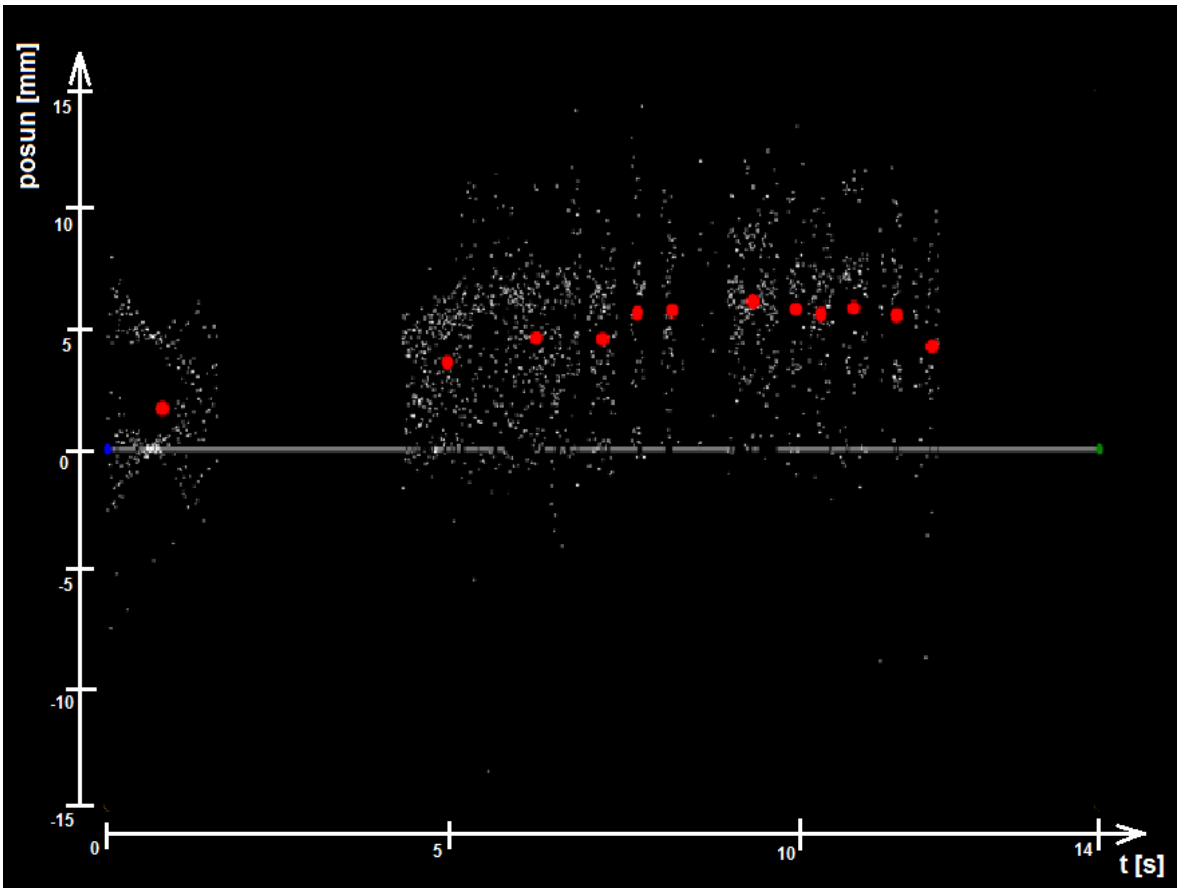
Obr. 8.9 Ukázka vyhledaných segmentů na náhodném skenu

Segmenty jsou tvořeny počátečním a koncovým bodem vyhledaného úseku. Každý segment je rovněž určen jak číslem skenu, ve kterém se nachází, tak specifickým indexem, který je totožný pro všechny stejné úseky napříč skeny.

Posun mezi každou po sobě jdoucí dvojicí skenů je vypočítán zprůměrováním posuvů všech segmentů vyskytujících se v obou těchto skenech. Přičemž posuv jednoho segmentu je roven průměru posunutí prvního a posledního bodu segmentu.

Segmenty, které se vyskytují na méně než osmi skenech, jsou z výpočtů vyřazeny, jelikož představují nestabilní úseky.

Vypočítané posunutí mezi každou sousední dvojicí skenů jsou přehledně zobrazeny na obrázku 8.10. Horizontální osa představuje časový průběh kamionu projíždějícího skrz skenovací bránu. Na vertikální ose jsou vypočítané posuvy pro každou sousední dvojici skenů.

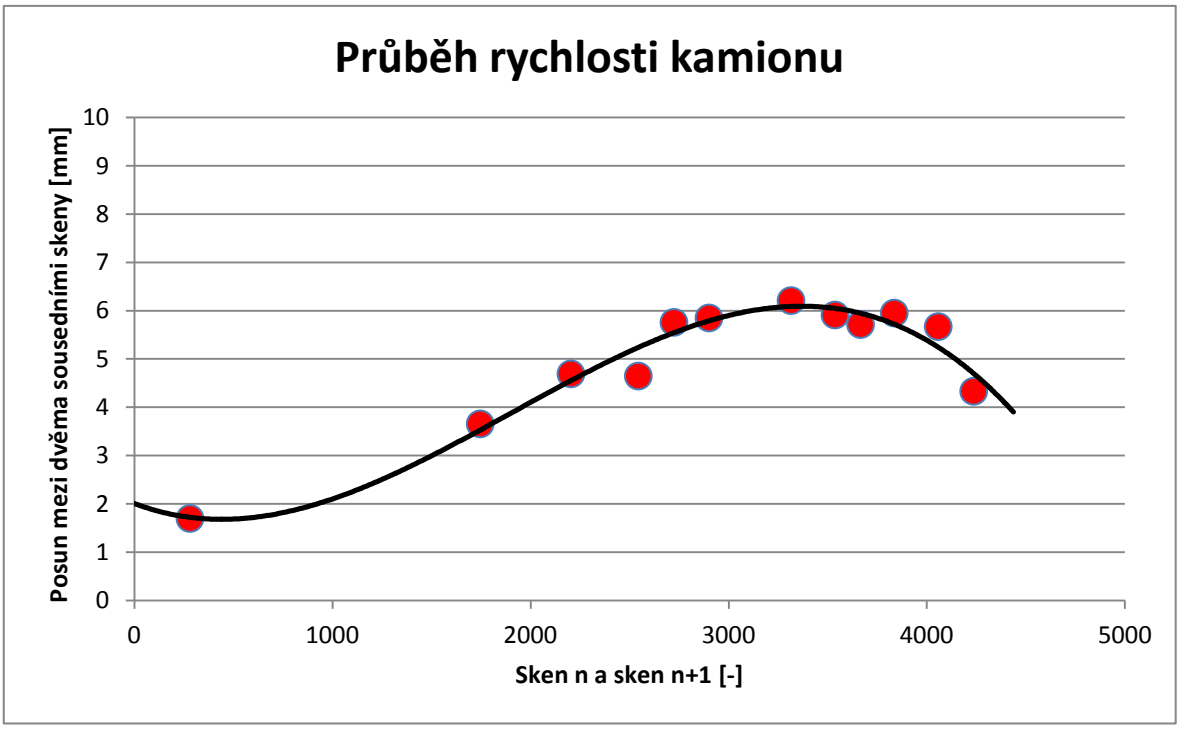


Obr. 8.10 Posuvy mezi skeny získané pomocí metody segmentů

Pro částí kamionu, které mají pestrý podélný profil, potažmo je u nich možno vypočíst posuv mezi sousední dvojicí skenu, je vypočítán jejích průměrný posuv reprezentováný červenými body. Těmito body je následné proložená kubická křivka, díky které je možné získat spojitý průběh rychlosti kamionu projíždějícího skenovací bránou.

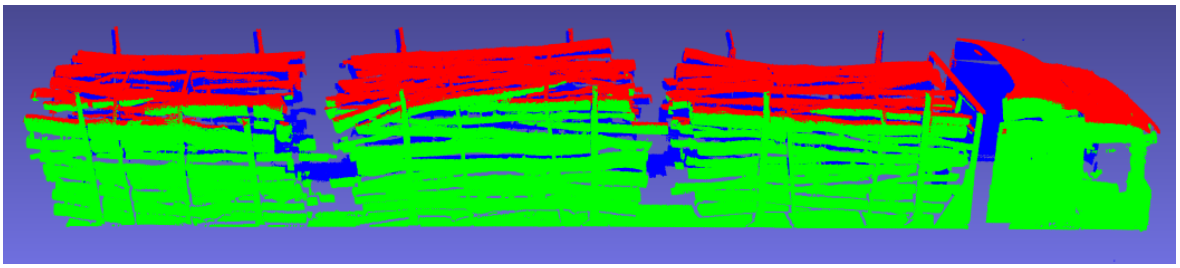
Rovnice kubické křivky vypočítaná pomocí metody segmentů:

$$y = -3{,}514 \cdot 10^{-10} \cdot x^3 + 2{,}007 \cdot 10^{-6} \cdot x^2 - 1{,}562 \cdot 10^{-03} \cdot x + 2{,}007$$



Obr. 8.11 Průběh rychlosti kamionu vypočítaný pomocí metody segmentů

Pomocí této kubické křivky lze zjistit skutečný posun kamionu mezi jednotlivými skeny. Na obrázku 8.12 je zobrazen 3D model kamionu, který byl získán pomocí metody segmentů.



Obr. 8.12 3D model kamionu získaný pomocí metody segmentů

3D model kamionu získaný pomocí metody segmentů již představuje skutečné rozměry kamionu. Kabina modelu kamionu již není protáhlá, tak jak tomu bylo u metody odhadu posuvu a jeho kontroly. Nákladový prostor kamionu taky splňuje reálné rozměry.

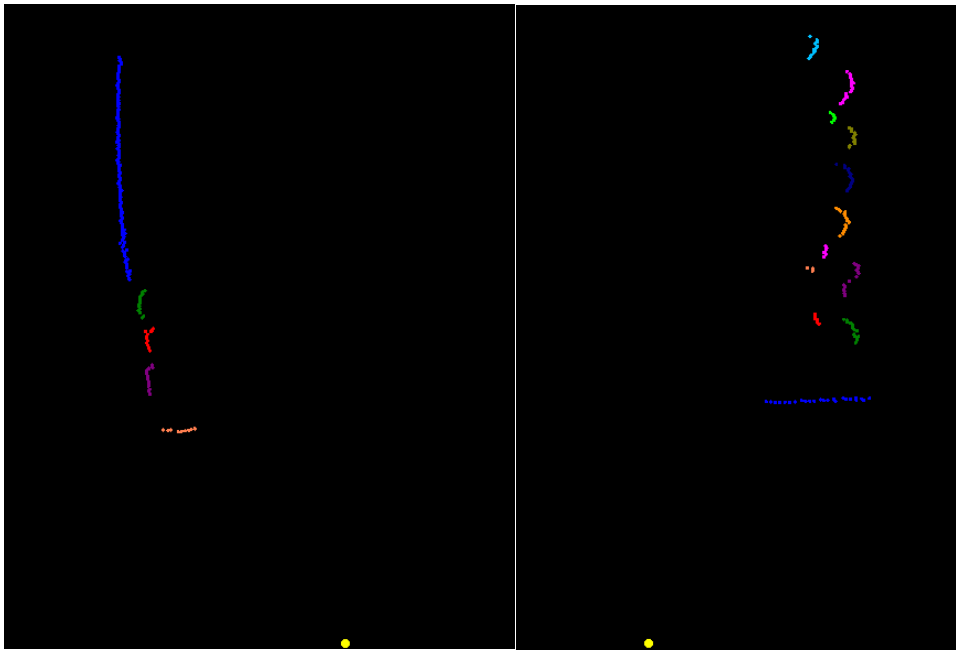
V příloze C se nachází ukázka kódu vytvořeného v programovacím jazyce C#, jehož úkolem je vytvoření a vykreslení na obrazovku realistický 3D model kamionu získaný pomocí obou zmíněných metod.

9 SEGREGACE NÁKLADU DŘEVA OD KAMIONU

Objem dřeva přiváženého na kamionech do podniku papíren je odhadován z jeho hmotnosti. Hmotnost suchého a mokrého dřeva stejného objemu se výrazně liší. Hmotnost mokrého dřeva u některých druhů dřevin přesahuje i dvojnásobek jeho hmotnosti v suchém stavu. K přesnějšímu určení objemu převáženého dřeva byla navržena metoda, která počítá objem dřeva z povrchu nasnímaného kamionu. (5)

Před výpočtem objemu dřeva převáženého na návěsu kamionu je zapotřebí rozdělit realistický 3D kamionu na dvě části. První 3D model bude obsahovat pouze povrch přepravovaného dřeva. Druhý 3D model bude obsahovat povrch tahače kamionu s návěsem, klanic a popruhů upevňujícími dřevo na návěsu. K segregaci byl vybrán realistický 3D model kamionu získaný pomocí metody segmentů, jehož rozměry jsou více autentické v porovnání s rozměry 3D modelu kamionu získaného pomocí metody odhadu posuvu a jeho kontroly.

Segregace nákladu dřeva od kamionu byla prováděna na jednotlivých skenech z pravého, levého a horního skeneru, reprezentujících příčný řez kamionu. Před samotnou segregací byly blízké body seskupeny do skupin. Na obrázku 9.1 jsou vykresleny náhodně vybrané skeny levého a pravého skeneru, u kterých bylo provedeno seskupení blízkých bodů. Každá skupina je pro názornost vykreslena jinou barvou. Velká žlutá tečka reprezentuje počáteční bod, respektive nulovou vzdálenost jak na vodorovné, tak na svislé ose.



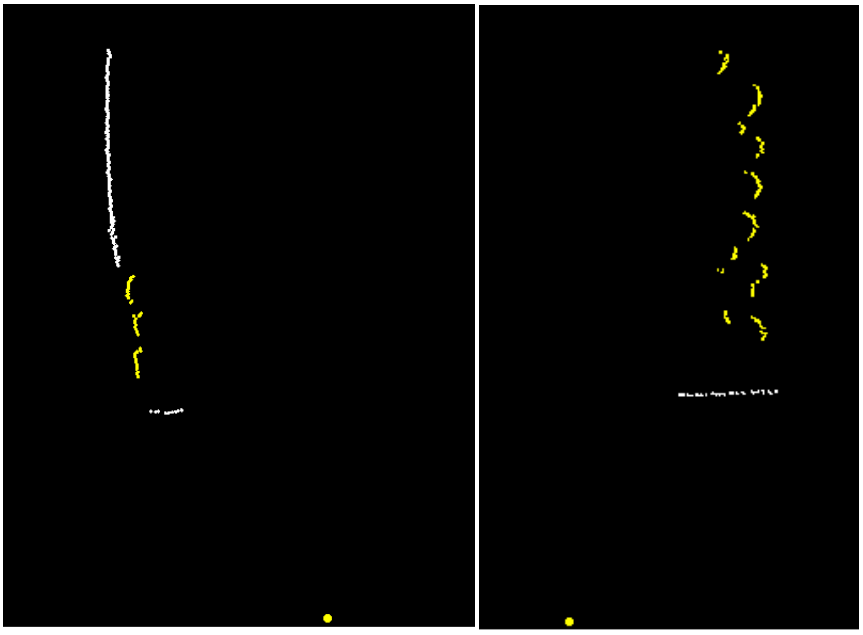
Obr. 9.1 Náhodný sken levého a pravého skeneru se seskupenými body

Algoritmus samotné segregace je poměrně komplikovaný. Nejprve byly větší skupiny bodů testovány, zdáli patří povrchu nákladu dřeva, či nikoliv. Jednotlivé skupiny bodů byly proloženy parabolou, pomocí metody nejmenších čtverců, a jestliže parabola splňovala předem definované podmínky pro její zakřivení a polohu vrcholu, pak byla celá tato skupina přiřazená ke dřevu. Kritérium pro zakřivení paraboly je definováno tak aby byly rozpoznány klády dřeva s max. ø1 m. Toto kritérium je dostačující pro dřevo naších zeměpisných šířek.

Větší skupiny bodů, které ne splnily podmínky definované pro dřevo, jsou testovány, zdáli jsou součástí kamionu, či nikoliv. U tohoto testu se využívá faktu, že všechny částí kamiony na příčném řezu vypadají jako rovné úseky. Proto se daná skupina bodů proloží přímkou, pomocí metody nejmenších čtverců, a sleduje se jak její úhel a poloha, tak průměrná vzdálenost bodů této skupiny od přímky. Jestliže body leží v těsné blízkostí aproximované přímky a její úhel a poloha splňuje určité parametry, pak je tento úsek přiřazen ke kamionu.

Po segregaci větších skupin jsou roztříděny i ty menší s využitím informací získaných při segregaci těch větších. Například skupiny tvořené velmi nízkým počtem bodů, které se nacházejí v rozmezí větších úseků, jež byly přiřazeny již dříve ke dřevu, a zároveň se nacházejí hlouběji směrem ke středu nákladu, jsou rovněž přiřazeny ke dřevu. Tyto skupiny reprezentují klády dřeva, které jsou uloženy hlouběji, a skener je naskenoval pouze pár měřícími paprsky, které se vešly mezi klády dřeva uloženého na povrchu.

Skupiny bodů, u kterých nelze jednoznačně určit, zda patří ke dřevu či ke kamionu, nejsou přiděleny k žádnému 3D modelu. Takové případy jsou ovšem ojedinělé.



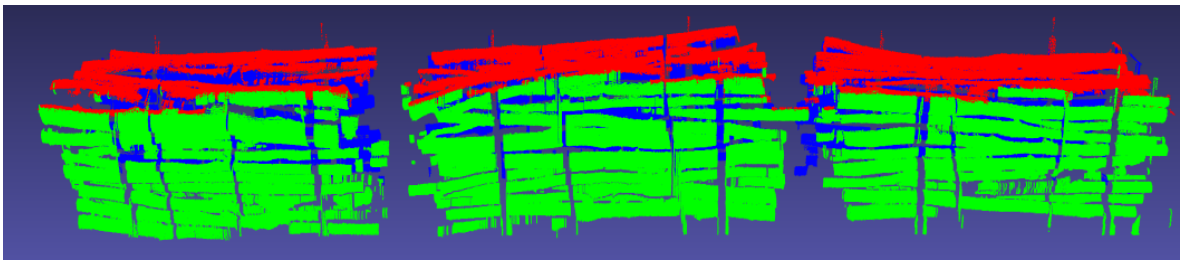
Obr. 9.2 Náhodný sken levého a pravého skeneru po segregaci

Na obrázku 9.2 jsou znázorněny dva náhodně vybrané skeny po segregaci. Bílou barvou jsou vyznačeny úseky kamionu, kdežto úseky reprezentující náklad dřeva jsou vyznačeny barvou žlutou. Na levém skenu se nachází návěs kamionu, část klanice, a dřevo. Na pravém skenu lze pozorovat pouze náklad dřeva a návěs.

Pomocí dat získaných z podélného skeneru lze přesně určit polohu zadní stěny kabiny. Všechny body ze skenů, které se vyskytují před zadní stěnou kabiny, jsou přiřazeny ke kamionu. Tímto je přiřazená celá kabina ke kamionu.

Po provedení separace kamionu od nákladu dřeva na jednotlivých skenech, byly tyto skeny složené za sebou a tím byly získány dva 3D modely.

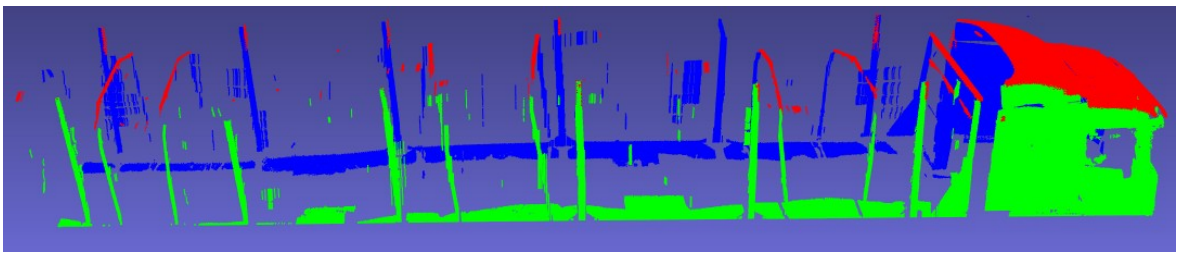
První 3D model obsahuje pouze čistý náklad dřeva.



Obr. 9.3 3D model čistého nákladu dřeva

Tento 3D model je připraven k výpočtu objemu dřeva převáženého na kamionu.

Druhý 3D model obsahuje tahač kamionu s návěsem, klanicemi a popruhy upevňující dřevo na návěsu.



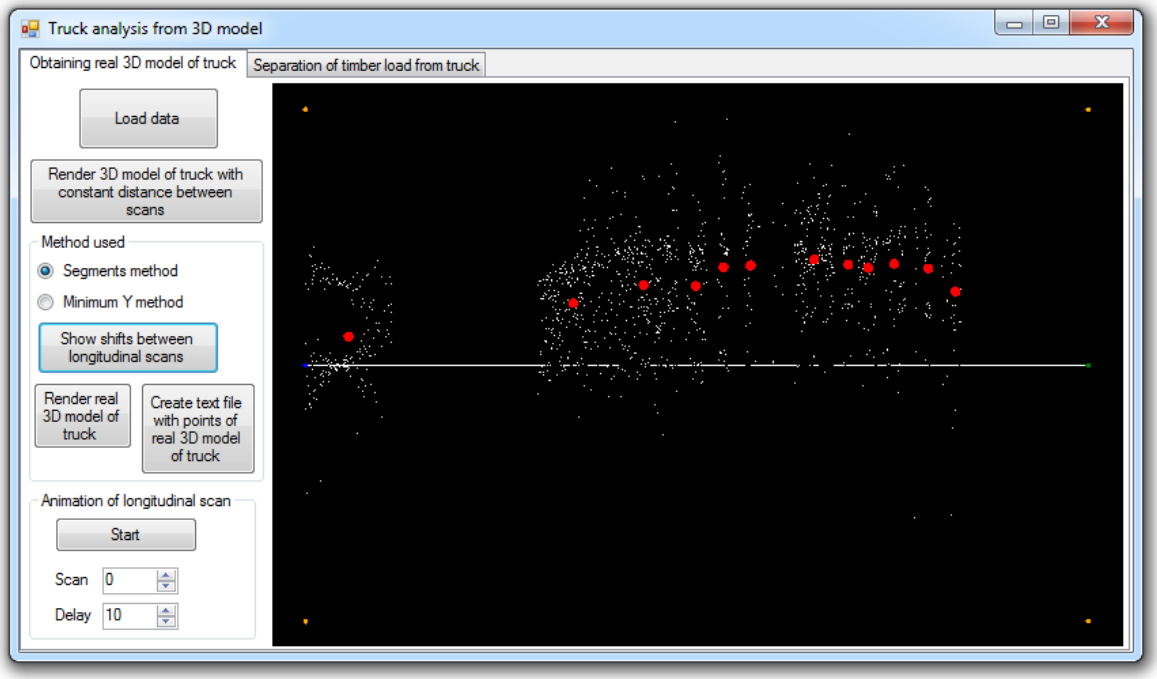
Obr. 9.4 3D model prázdného kamionu

10 WINDOWS APLIKACE

Windows desktopová aplikace je rozdělena na dvě části. Obě částí aplikace se vyznačují tím, že jejich levá strana obsahuje pestrou škálu tlačítek s příkazy a pravá část okna aplikace slouží k vykreslování různých 2D schémat a jednotlivých nasnímaných skenů. Pro zobrazování 3D modelů je využíváno samostatné okno. Tyto 3D modely jsou pro analýzu obrazu velmi důležité, protože můžeme pozorovat kompletní model ze všech stran.

10.1 Vytvoření realistického 3D modelu kamionu

V první záložce aplikace jsou umístěny funkcionality, které byly navrženy pro zjednodušení práce při určování realistického 3D modelu kamionu.



Obr. 10.1 Záložka aplikace určena k získání realistického 3D modelu kamionu

Po spuštění aplikace je nejprve nutno načíst data, získaná naskenováním kamionu, do aplikace. Textový soubor s daty je jednoduše vybrán z adresáře aktuálního PC. Po vložení dat lze využívat ostatní funkcionality aplikace.

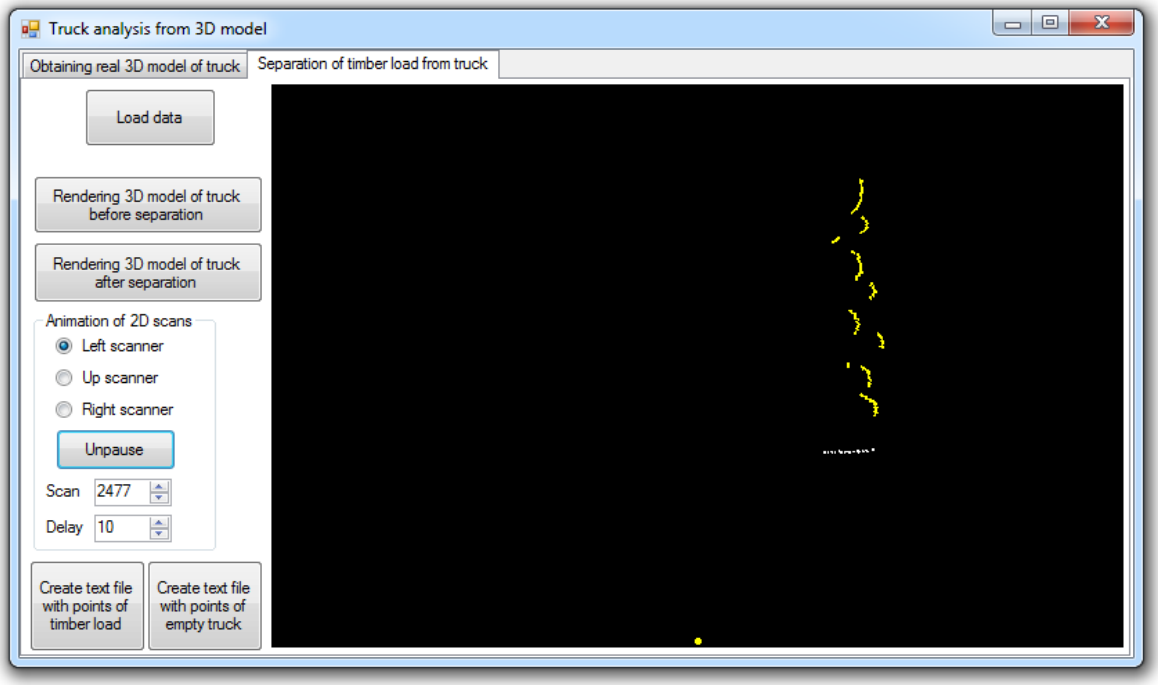
První z nich slouží k vykreslení 3D modelu kamionu s konstantními posuvy mezi jednotlivými skeny na obrazovku.

Další funkce aplikace slouží k znázornění velikostí všech posuvů, které kamion urazil mezi jednotlivými skeny. Dále je možno vykreslit realistický 3D model kamionu na obrazovku anebo vytvořit textový dokument, který bude obsahovat všechny body realistického modelu kamionu definované v trojrozměrném kartézském systému. Všechny tyto tři metody mohou být použity pro realistický model kamionu získány jak pomocí metody segmentů, tak pomocí metody odhadu posuvu a jeho kontroly.

Poslední funkce slouží k animaci skenů nasnímaných podélným skenerem i s vyznačenými segmenty, které jsou stěžejní při určování realistického 3D modelu kamionu pomocí metody segmentů. Animaci je možno kdykoliv pozastavit, pozměnit její rychlost a dokonce přeskočit na zvolený sken.

10.2 Segregace nákladu dřeva od kamionu

Druhá záložka desktopové aplikace obsahuje funkcionality, které byly navrženy pro zjednodušení práce při segregaci nákladu dřeva od kamionu.



Obr. 10.2 Záložka aplikace určená k separaci nákladu dřeva od kamionu

Obdobně jak tomu bylo u první záložky i tady je nutno nejprve načíst data, získaná naskenováním kamionu, do aplikace. Textový soubor s daty je jednoduše vybrán z adresáře aktuálního PC. Ostatní funkcionality této záložky je možno využít až po načtení dat.

Pomocí prvního tlačítka je generován realistický 3D model kamionu v externím oknu. Obdobnou funkci má druhé tlačítko s tím rozdílem, že je vykreslen 3D model kamionu s nákladem dřeva po segregaci. 3D model nákladu dřeva je umístěn nad 3D modelem prázdného kamionu. Díky tomu je možno sledovat oba 3D modely v jednom oknu, což je výhodné pro přehlednost a rychlost práce.

Další funkce slouží k animaci skenů, které byly nasnímané pomocí pravého, levého nebo horního skeneru. Na jednotlivých skenech jsou barevně odlišeny body přiřazené během segregace k nákladu dřeva od bodů přiřazených k prázdnému kamionu. Animaci je možno kdykoliv pozastavit, pozměnit její rychlost, přeskočit na zvolený sken a změnit vybraný skener.

Dva tlačítka v dolní části okna slouží k vytvoření textového dokumentu do vybraného adresáře. První z nich zapíše do textového souboru všechny body reprezentující čistý náklad dřeva převážený na kamionu. Druhé tlačítko slouží k zápisu všech bodů, náležících prázdnému kamionu, do textového souboru. Body jsou definovány polohou X, Y a Z v kartézském systému.

10 ZÁVĚR

V první části diplomové práce proběhlo seznámení se s principem skenování a vytváření počítačového 3D modelů kamionů s nákladem dřeva přijíždějících do společnosti MONDI SPC, a.s. sídlícím v městě Ružomberok. Byl popsán jak způsob detekce kamionu na příjezdové cestě pomocí kamer CANTONK KIP300CK60A, tak samotný princip skenování nákladu pomocí sady laserových skenerů LMS 400 firmy SICK upevněných na příjezdové bráně. Následně proběhlo seznámení se s unikátními knihovnami, které slouží k synchronizovanému skenování objektů z více stran pomocí sady skenerů LMS 400.

Následující část diplomové práce je věnována stručnému popisu otevřené knihovny EmguCV obsahující velké množství algoritmů, které slouží k analýze a prácí jak se 2D obrazem tak s obrazem ve formátu 3D. Popsány byly jen funkce, struktury a nástroje z knihovny EmguCV, jež byly použity při práci na této diplomové práci.

Po vypracování teoretické části byla podrobně zdokumentována praktická část, jejímž cílem bylo vytvoření realistického 3D modelu dřeva převáženého na kamionech směřujících do papíren sídlících v Ružomberoku. Z takového 3D modelu je možno vypočíst objem převáženého dřeva.

Nejprve proběhla filtrace dat získaných naskenováním povrchu kamionu pomocí sady skenerů. Po filtraci byl vytvořen počítačový 3D model naloženého kamionu s konstantním posuvem mezi jednotlivými skeny.

Tento 3D model kamionu byl nedostačující pro dosažení cíle této úlohy, a proto byly navrženy dvě metody pro získání realistického 3D modelu kamionu. K získání takového modelu bylo zapotřebí vypočítat spojitý průběh rychlosti kamionu projíždějícího skrz skenovací bránu. K tomu byly využity data naměřená pomocí podélného skeneru, který zaznamenává podélný profil kamionu. První metoda odhaduje posuv mezi jednotlivými dvojicemi skenů a kontroluje jeho přesnost. 3D model kamionu získaný pomocí této metody nebyl tak věrný reálnému kamionu jako 3D model kamionu získaný pomocí druhé metody. Ta počítá posuv mezi dvojicemi sousedních skenů přímo, pomocí segmentů reprezentujících výrazné úseky kamionu na jeho podélném profilu.

Realistický 3D model kamionu tvořil základ pro další část diplomové práce, která se věnuje segregaci nákladu dřeva od kamionu. Tato segregace probíhala na úrovni 2D skenů tvořících příčný průřez kamionu. Po propojení roztříděných skenů vznikly dva 3D modely. První z nich obsahuje pouze čistý náklad dřeva. Druhý 3D model obsahuje tahač kamionu s návěsem, klanicemi a popruhy upevňující dřevo na návěsu.

I když je algoritmus pro segregaci náročný a komplikovaný, nebylo docíleno vytvoření perfektních 3D modelů čistého dřeva a prázdného kamionu. I přes veškerou snahu odladit

proces segregace před termínem odevzdání práce, ne bylo tohoto stavu docíleno. Na vytvořených 3D modelech se pořád vyskytují nedokonalosti, kde části dřeva jsou chybně přiřazeny ke kamionu, a obráceně. Nedokonalosti na 3D modelech jsou ovšem minimální a jejich vliv při případném výpočtu objemu dřeva ze 3D modelu je zanedbatelný.

V poslední části práce byla popsána desktopová aplikace, která byla vytvořena k získání jak realistického 3D modelu kamionu, tak 3D modelu dřeva odseparovaného od modelu prázdného kamionu. V obou částech byly vytvořeny pomocné funkce, které ulehčovaly práci díky rychlé vizuální kontrole výsledků jak na grafech nebo skenech ve formátu 2D, tak na 3D modelech.

SEZNAM POUŽITÉ LITERATURY

1. Our history | Mondi Group. Object moved [online]. Dostupné z: https://www.mondigroup.com/en/about-mondi/our-history/

2. SICK AG. *LMS400 Laser Measurement Sensors: Operating Instructions*. Walkirch, Germany, ©2013-2017. Dostupné také z: https://www.sick.com/media/docs/8/98/698/Operating_instructions_LMS400_Laser_measurement_sensors_en_IM0010698.PDF

3. KIP-300CK60A/POE venkovní 3-MPX (Full HD) IP kamera s WDR, variobjektiv, IR LED (60m). *Bezpečnostní kamery, kamerové systémy, zabezpečení, CCTV, webové IP-kamery* [online]. Copyright © 2018 ESCAD Trade s.r.o. [cit. 10.05.2018]. Dostupné z: http://www.escadtrade.cz/kip-300ck60a-poe-venkovni-3-mpx-full-hd-ip-kamera-s-wdr-variobjektiv-ir-led-60m.html

4. Emgu CV: OpenCV in .NET (C#, VB, C++ and more). [online]. Dostupné z: http://www.emgu.com/wiki/index.php/Main_Page

5. Objemová hmotnost dřeva | Dřevorubec.cz | DEBUG MODE. *Rizikové kácení stromů, prořezávání, výškové práce, arboristika* | DEBUG MODE [online]. Copyright © 2007 [cit. 04.05.2018]. Dostupné z: http://drevorubec.cz/prodej-dreva/objemova-hmotnost-dreva

6. DEVDEPT. *Eyeshot Documentation*. Bologna, Italy, ©2006-2017. Dostupné také z: http://documentation.devdept.com/100/Common/webframe.html#topic2.html

7. FOJTÍK , David, Petr PODEŠVA, Miroslav MAHDAL a Milan MIHOLA. Scanning of trucks to produce 3D models for analysis of timber loads. In: *Proceedings of the 2016 17th International Corporation Control Conference, ICCC 2016*. Danvers: IEEE, 2016. s. 194-199. ISBN 978-1-4673-8606-7.

8. KAEHLER, Adrian a Gary R. BRADSKI. *Learning OpenCV 3: computer vision in C++ with the OpenCV library*. Sebastopol, CA: O'Reilly Media, 2017. ISBN 1491937998.

9. LAGANIÈRE, Robert. *OpenCV 2 computer vision application programming cookbook over 50 recipes to master this library of programming functions for real-time computer vision.* Birmingham, UK: Packt Pub, 2011. ISBN 1849513252.

10. SHI, Shin. *Emgu CV essentials: develop your own computer vision application using the power of Emgu CV.* Birmingham, UK: Packt Publishing, 2013. ISBN 1783559527.

11. Microsoft API a referenční katalog. *Learn to Develop with Microsoft Developer Network* | MSDN [online]. Copyright © 2018 Microsoft [cit. 10.05.2018]. Dostupné z: https://msdn.microsoft.com/cs-cz/library

12. BRADSKI, Gary R. *Learning OpenCV*. Sebastopol: O'Reilly, c2008. ISBN 978-0-596-51613-0.

13. Stack Overflow - Where Developers Learn, Share, & Build Careers. Stack Overflow - *Where Developers Learn, Share, & Build Careers*[online]. Dostupné z: https://stackoverflow.com/

14. RANdom Sample Consensus (RANSAC) in C# – César Souza. *César Souza – Sharing knowledge efficiently* [online]. Dostupné z: http://crsouza.com/2010/06/02/random-sample-consensus-ransac-in-c/

15. FARANA, Radim, Lubomír SMUTNÝ a Antonín VÍTEČEK. *Zpracování odborných textů z oblasti automatizace a informatiky*. 1. vyd. Ostrava: VŠB - Technická univerzita Ostrava, Strojní fakulta, 1999. ISBN 80-7078-737-6.

Příloha A: Seznam obrázků

Příloha B: Seznam tabulek

Příloha C: Ukázka kódu

```
//Vytvoření realistického 3D modelu kamionu a jeho vykreslení na obrazovku.

//korekce kamionu podle metody segmentů
if (rbSM.Checked)
{
	//zjistí posuvy mezi každou dvojicí skenů získané pomocí metody segmentů
	differencesBetweenScans = Functions.FindShiftsBySegmentsMethod
	(LongitudinalPoints  //vektor obsahující všechny 2D body nasnímané podélným
	skenerem
	, qLo  //počet skenů podélného skeneru
	, LongitudinalScanner.NumberMeasureValues  //počet nasnímaných 2D bodů
	obsažených v jednom skenu podélného skeneru
	, 35);  //minimální počet bodů k vytvoření segmentu

	//výpočet pro získání pár bodů potřebných k zjištění spojitého průběhu
	rychlostí kamionu projíždějícího skrz skenovací bránu pro metodu segmentů
	pointsForCurve = Functions.FindFewImportantPointsForSegmentsMethod
	(LongitudinalPoints  //vektor obsahující všechny body nasnímané podélným
	skenerem
	, LongitudinalScanner.NumberMeasureValues  //počet nasnímaných 2D bodů
	obsažených v jednom skenu podélného skeneru
	, differencesBetweenScans  //posuvy mezi každou dvojicí skenů získané pomocí
	metody segmentů
	, 35);  //minimální počet bodů k vytvoření segmentu
}
//korekce kamionu podle metody odhadu posuvu a jeho kontroly
else if (rbMYM.Checked)
{
	//zjistí posuvy mezi každou dvojicí skenů získané pomocí metody odhadu posuvu
	a jeho kontroly
	differencesBetweenScans = Functions.FindShiftsByMinimumYMethod
	(LongitudinalPoints  //vektor obsahující všechny body nasnímané podélným
	skenerem
	, qLo  //počet skenů podélného skeneru
	, LongitudinalScanner.NumberMeasureValues);  //počet nasnímaných 2D bodů
	obsažených v jednom skenu podélného skeneru

	//výpočet pro získání pár bodů potřebných k zjištění spojitého průběhu
	rychlostí kamionu projíždějícího skrz skenovací bránu pro metodu odhadu
	posuvu a jeho kontroly
	pointsForCurve = Functions.FindFewImportantPointsForMinimumYMethod
	(differencesBetweenScans);  //posuvy mezi každou dvojicí skenů získané pomocí
	metody odhadu posuvu a jeho kontroly
}

//nová kolekce pro 2D body jehož souřadnice jsou ve formátu double
List<PointD> doublePointsForCurve = new List<PointD>();

//převede body potřebné pro vytvoření průběhu rychlostí kamionu do formátu double,
který je mnohem přesnější pro výpočty
foreach (var fpfc in pointsForCurve)
{
	doublePointsForCurve.Add(new PointD((double)fpfc.X, (double)fpfc.Y));
}

//zjistí koeficienty algebraické rovnice
List<double> coefficients = CurveFunctions.FindPolynomialLeastSquaresFit
(doublePointsForCurve  //body pro proložení křivkou ve formátu double
, 3);  //stupeň rovnice (kubická křivka = 3)
```

```
//korekce Z-ové souřadnice všech bodů nasnímaných levým skenerem
VectorOfPoint3D32F leftPoints = Functions.CorrectZCoordinate
(LeftPoints  //vektor obsahující všechny 3D body nasnímané levým skenerem
, LeftScanner.NumberMeasureValues  //počet nasnímaných 2D bodů obsažených v jednom
skenu levého skeneru
, coefficients);  //koeficienty kubické křivky

//korekce Z-ové souřadnice všech bodů nasnímaných horním skenerem
VectorOfPoint3D32F upPoints = Functions.CorrectZCoordinate
(UpPoints  //vektor obsahující všechny 3D body nasnímané horním skenerem
, UpScanner.NumberMeasureValues  //počet nasnímaných 2D bodů obsažených v jednom
skenu horního skeneru
, coefficients);  //koeficienty kubické křivky

//korekce Z-ové souřadnice všech bodů nasnímaných pravým skenerem
VectorOfPoint3D32F rightPoints = Functions.CorrectZCoordinate
(RightPoints  //vektor obsahující všechny 3D body nasnímané pravým skenerem
, RightScanner.NumberMeasureValues  //počet nasnímaných 2D bodů obsažených v jednom
skenu pravého skeneru
, coefficients);  //koeficienty kubické křivky

//definice barvy pro body nasnímané horním skenerem
MCvScalar UpColor = new MCvScalar(0, 0, 255, 255);      //červená
//definice barvy pro body nasnímané levým skenerem
MCvScalar LeftColor = new MCvScalar(0, 255, 0, 255);    //zelená
//definice barvy pro body nasnímané pravým skenerem
MCvScalar RightColor = new MCvScalar(255, 0, 0, 255);   //modrá

//definice mračna bodů nasnímaných pomocí horního skeneru
WCloud UpCloud = new WCloud(upPoints, UpColor);
//definice mračna bodů nasnímaných pomocí levého skeneru
WCloud LeftCloud = new WCloud(leftPoints, LeftColor);
//definice mračna bodů nasnímaných pomocí pravého skeneru
WCloud RightCloud = new WCloud(rightPoints, RightColor);

//definice okna pro vizualizaci 3D oběktů
Viz3d v = new Viz3d("model");

//nastavení pozadí vizualizačního okna
v.SetBackgroundMeshLab();

//do vizualizačního okna přidá mračno bodů, které byly nasnímané pomocí horního
skeneru
v.ShowWidget("UpCloud", UpCloud);
//do vizualizačního okna přidá mračno bodů, které byly nasnímané pomocí levého
skeneru
v.ShowWidget("LeftCloud", LeftCloud);
//do vizualizačního okna přidá mračno bodů, které byly nasnímané pomocí pravého
skeneru
v.ShowWidget("RightCloud", RightCloud);


//zahájí cyklus pro vykreslení všech součástí ve vizualizačním okně
v.Spin();
```