

Óbudai Egyetem

Neumann János Informatikai Kar

Szoftvertechnológia Intézet



TUDOMÁNYOS DIÁKKÖRI DOLGOZAT

AutoNomus, autonóm autó navigációjának megvalósítása

Szerző:

Fugerth Máté

mérnök informatikus szak, IV. évf.

Konzulens:

Vámossy Zoltán

egyetemi docens

Tartalom

1.	Bevezető	4
2.	Célkitűzés.....	4
3.	Hasonló rendszerek	4
3.1.	Akadálydetektálás.....	4
3.1.1.	Akadálydetektálás és elkerülés mobil robotoknál.....	4
3.1.2.	Megjelenés alapú akadály felismerés színes egykamerás rendszereknél	5
3.1.3.	Optikai folyamon és mélység információ alapuló mozgó tárgyszegmentálás	6
3.1.4.	Akadálydetektálás egy kamerával és sztereo szenzorral.....	7
3.1.5.	Kerekes robotok látás alapú akadályérzékelése	8
3.1.6.	Autonóm robot valós idejű akadályelkerülése aktív szenzoros, vertikális lézervonal kivetítéssel	10
3.1.7.	3D objektumdetektálás országúti adatokból lézerszkennelést használva.....	10
3.2.	Térbeli szkennelés	11
3.2.1.	Automatikus 3D modellezés képszekvenciából.....	11
3.2.2.	MoBRo, háromdimenziós környezet leképező robot	12
4.	A rendszer modulvázlata	13
5.	A rendszer feladatai.....	13
5.1.	Képalkotás	13
5.2.	Akadálydetektálás.....	14
5.3.	Kikerülés irányának meghatározása	14
5.4.	Robotvezérlés	14
5.5.	Robotirányítás.....	14
5.6.	Lézerpontok detektálása	15
5.7.	Pontok visszatranszformálása a térbe	15
5.8.	Modellalkotás	15
6.	Lézerszkennelést megvalósítása	16
6.1.	Lézerdetektálás.....	16
6.1.1.	Színtéren és éleken alapuló detektálás	16
6.1.2.	Leíró alapú detektálás.....	18
6.1.3.	Leíró alapú detektálás továbbfejlesztése	19
6.1.4.	Szubpixeles középpont meghatározás.....	20

6.1.5.	Zajszűrés	21
6.2.	Kamerakalibrálás	22
6.2.1.	Kamera- és világ koordinátarendszerek kapcsolata	22
6.2.2.	Kalibrálás folyamata	22
6.3.	Rekonstrukció elve.....	23
6.4.	Lézerkalibrálás	24
6.4.1.	Kalibrálás elve	24
6.4.2.	Lézerkalibrálás folyamata	25
6.5.	Lézerszkennér értékelése	25
7.	Akadálydetektáló modul.....	26
7.1.	Felülnézeti kép elkészítése	26
7.1.1.	Felülnézeti kép kalibrálása.....	28
7.1.2.	Felülnézeti kép használata.....	29
7.2.	Akadálydetektálás.....	29
7.2.1.	Akadálydetektálás lézer segítségével	29
8.	Robotvezérlés interfésze	35
9.	Útvonaltervező modul.....	36
9.1.	Bogár algoritmus.....	36
9.2.	Javított bogár algoritmus.....	36
9.3.	Akadályelkerülés a legnagyobb térköz irányába	36
9.4.	Legközelebbi akadály kikerülése.....	36
9.5.	Az előbbi két modell ötvözése	37
9.6.	Exact cell decomposition algoritmus	37
9.7.	Értékelés	38
10.	Robot útvonalon tartása, irányítása	38
10.1.	Tesztelés	39
10.2.	Modul értékelése.....	39
11.	Értékelés	40
12.	Összegzés.....	40
13.	Továbbfejlesztési lehetőségek.....	40
14.	Felhasznált irodalom	42

1. Bevezető

Mindig is érdekelték az autonóm robotok, így egy ilyen szerkezet építését terveztem már régebb óta. A dolgozat keretein belül lehetőség adódott, hogy egy hardveres implementáció szülessen az elképzelésekből. A tervezett robotnak az autonómitását az érdeklődési körömbbe tartozó képfeldolgozás által próbálom megteremteni, így gondoskodva arról, hogy mind a természetes, mind a mesterséges környezetben helyt állhasson. A dolgozat a rendszer tervezését és kivitelezését tartalmazza.

2. Célkitűzés

Napjainkban a robotikában egyre inkább arra törekednek a fejlesztők, hogy robotjaik önállóan is megállhassák a helyüket, ne kelljen kívülről beavatkozni a működésükbe, azaz autonómok legyenek. Mindemellett rengeteg problémával találkozhatunk a kivitelezés közben. A robotnak érzékelnie kell az akadályokat, de ez nem elégséges feltétele az önállóságnak, csak szükséges, mivel ha csak ki tudja kerülni az akadályokat, akkor még sokszor átmehet a már „ismert” helyeken, és ezt nem veszi észre, azaz buta maradt a gép. Ennek a problémának a megoldásaként használhatunk valamilyen 3D modellezési eszközt, amivel a környezetet immáron digitális formába leképezhetjük, és ebben a digitális modellben, már fel tudja mérni a robotunk, hogy merre járt, és azt a többszöri áthaladás alatt részletesebbé is tudja tenni. Az így előállt rendszert már bátran nevezhetjük elégségesnek egy autonóm robot esetén. Ezt a két rendszert, azaz az akadály detektálót és a 3D szkennert ötvözve próbálunk megalkotni egy a robotok autonómitását támogató, megteremtő szoftveres és hardveres környezetet.

3. Hasonló rendszerek

3.1. Akadálydetektálás

3.1.1. Akadálydetektálás és elkerülés mobil robotoknál

Az [1] dolgozat megvizsgálta, hogy egy robotra helyezett számítógéppel hogyan lehet a leghatékonyabban akadályt felismerni, keresni. Az alaphardver egy mára már elavultnak tekinthető Pentium MMX-es processzorral szerelt konfiguráció volt, VGA felbontású kamerával kiegészítve. Az akadály érzékelésére az Optikai Folyam módszerét alkalmazták, melyet erre a konfigurációra optimalizáltak bizonyos megkötésekkel, hogy a kívánt sebességet el tudják érni. Majd erre épült egy képszegmentáció, így már a tárgyak mérete és helyzete is nagyjából meghatározhatóvá vált. Az ütközés érzékelésére a képet 32 darab 20 pixel széles oszlopra osztották és ezekben a sávokban határozták meg az ütközés kockázatát, az értékeket Kálmán filterrel szűrték, hogy az zajokat eltávolítsák a rendszerből, majd az így kapott diagramot elemezve határozták meg az akadályokat és az elkerülés irányát.

3.1.1.1. Értékelés

A módszer az előnye az egyszerűség, könnyű megvalósíthatóság, és a mai rendszereken való gyors futás. Kameraként lehet párhuzamosítani, így egy több kamerás rendszerrel az eredmények átlagolásából, valamilyen súllyal vett meghatározásából jól számolható egy ütközésmentes

továbbhaladási irány. Előnye még, hogy kalibrált kamerával akár mélységi adatokat is ki lehet nyerni ebből a modellből, így a saját rendszerbe is illeszkedik ez a módszer.

3.1.2. Megjelenés alapú akadály felismerés színes egykamerás rendszereknél

A [2] dolgozatban bemutatott robot, egy jó közelítéssel síknak mondható terepen az előtte lévő akadályokat nagy hatékonysággal detektálja, és azokat elkerüli. Ehhez számos korábban elért eredményt használt fel, és fejlesztett tovább. A legfőbb vonal, amin tovább mentek, a hisztogram alapú textúra elemzés volt, mellyel már egy 1997-es dolgozatban is elérték 64*64 pixeles képeknél a valós idejű feldolgozást, ami az adott időszakban igen jó eredménynek számított.

A kifejlesztett akadálydetektáló rendszernek az képezi az alapját, hogy a robot előtt található terület általánosan különbözik az akadályoktól, azaz a felület pixelei nagymértékben eltérnek a detektálandó akadályok pixeleitől. A rendszer három feltevésen alapul:

- Az akadályok megjelenése alapvetően különbözik a földtől.
- A robot előtt lévő talaj viszonylag lapos, sík.
- Nincsenek kiálló akadályok.

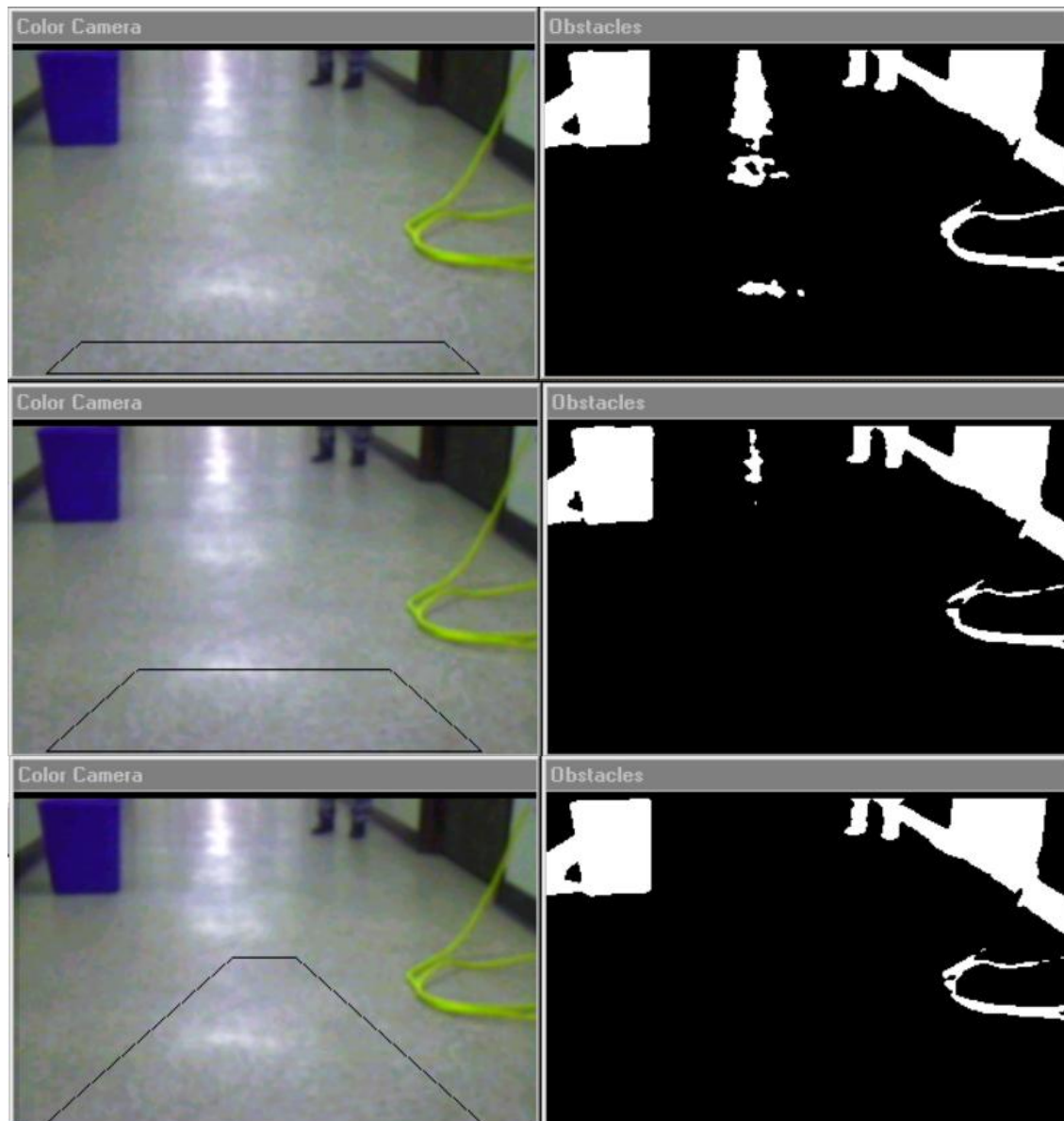
Ezek a feltételek eléggé bekorlátozzák a módszer alkalmazhatóságát, de belső tereknél jól használható, előnye az él és textúra felismerő rendszerekhez képest, hogy sokkal gyorsabban kalkulálható az eredmény a színek feldolgozásával, mint ha valamilyen éldetektáló, vagy textúra elemző algoritmust használnánk, mivel a színek kinyerése pont művelet, az éldetektálás és a textúra elemzés meg tipikusan maszk művelet.

Ahhoz, hogy az akadályok távolságát meg tudjuk becsülni a fenti három feltétel kiemelkedően fontos. Ha ezeknek a feltételeknek eleget tesz a környezet, akkor a távolságot egy jól bekalibrált eszközzel aránylag pontosan meg tudjuk határozni.

Az algoritmus menete az alábbi:

- 5*5-ös Gauss szűrővel zajt szűrünk a képen.
- Áttranszformáljuk az RGB színteret HSI színtérbe (hue, saturation, intensity).
- Csak akkor vesszük figyelembe a szín információt, ha az intenzitás elért egy bizonyos küszöböt. A színárnyalatot csak akkor vesszük figyelembe, ha a szaturáció egy bizonyos küszöb felett van.
- Majd a robot előtti térben egy trapéz alakú területet véve a színárnyalat és intenzitás értékekből készítünk egy hisztogramot.
- Ezt a két hisztogramot egy alul-áteresztő szűrővel szűrjük az átlag alapján.
- Az összes pixelre megnézzük, hogy egy küszöb érték alatt van-e, amennyiben igen, akkor akadály.

Ez az algoritmus az említett dolgozatban különböző távolságokat vizsgálva az 1-es ábrán látható eredményt szolgáltatja (baloldalon a vizsgált színes képpel és a vizsgált trapéz területtel, jobb oldalt a detektált akadályokkal).



1. ábra Hisztogram alapú algoritmus eredménye. Forrás: [2]

3.1.2.1. Értékelés

A módszer előnye, hogy változó fényviszonyok között is igen jól használható, az akadályokat jól detektálja a minták alapján, ám a Gauss szűrő, a HSI színtérbe való átkonvertálás igen időigényes, emiatt az algoritmus az akkori körülmények között nem ért el túl jó eredményt. Továbbá az algoritmust az esetleges árnyékok nagymértékben befolyásolhatják, mind pozitív, mind negatív irányba, így ezt a rendszert önmagában nem tartom alkalmazhatónak. Azonban az előbbi rendszerhez képest megvan az az előnye, hogy álló robot esetén is tudja detektálni az akadályokat, így az alkalmazása ebben a szituációban számunkra is hasznos lehet.

3.1.3. Optikai folyamaton és mélység információn alapuló mozgó tárgyszegmentálás

Egy optikai folyamaton alapuló mozgó-objektum szegmentáló algoritmust mutat be a [3] dolgozat. Ez az algoritmus lehetővé teszi, hogy a mozgó tárgyakat meghatározhassuk és kijelölhessük egy adott képen. Ezt kétféle rendszerrel valósították meg. Az egyik egykamerás, a másik egy sztereo kamerás

felállásra lett kidolgozva. Mivel a tervezett rendszerünk egy kamerával kell, hogy érzékelje az akadályokat, ezért részletesebben ezt fogjuk vizsgálni. Az egykamerás rendszer hátránya, hogy nem lehet vele egyszerűen mélység információt előállítani. Valamilyen módszer alkalmazásával meg kell határozni a kamera saját mozgását, ami történhet például optikai folyam vizsgálattal. A térreprezentációhoz szükséges a különböző képeken megjelenő azonos pontok párosítása, melyet csak nagy hiba aránnyal lehet meghatározni, így ennek a módszernek nagyobb a hibaráta, mint a sztereo látásnak.

Az egykamerás rendszernél az optikai folyam több jellemzőjét is kihasználhatjuk, hogy a statikus és mozgó tárgyakat el tudjuk különíteni. A mozgó tárgy a képen olyan formán jelenhet meg, hogy a mozgás vektorai nem egy középpont felé mutatnak, mint a robot egyenes haladásánál a vektorok általában, hanem egy ettől eltérő konstans irányba. Így már el tudjuk különíteni a mozgó és statikus elemeket a képen. A háromdimenziós pontok előállításához először két képet használtak fel ebben a munkában. Itt vizsgálták, hogy a két képen azonos pontok mozgása egy egyenesre esik-e, majd egy harmadik képet is hozzávettek a vizsgálathoz, amennyiben ennek a mozgása is a kiszámolt egyenesre esik, akkor az a pont egy statikus pont, és ha a kamera helyzetét ismerjük, akkor háromszögeléssel kiszámolható a tárgy távolsága.

A szegmentálásnál két részre osztjuk a képet, a mozgó előtérre, és a statikus háttérre. Erre egy küszöböt alkalmazunk, ami a zajokat elfedi, és azokat is háttérnek tekintjük, míg a többi elemet a mozgó tárgyaknak. Ezt a küszöböt 0,1 pixelnek vették ebben a dolgozatban. Ezután alkalmaztak egy több felbontású gráf vágó algoritmust a gyorsabb eredmény érdekében.

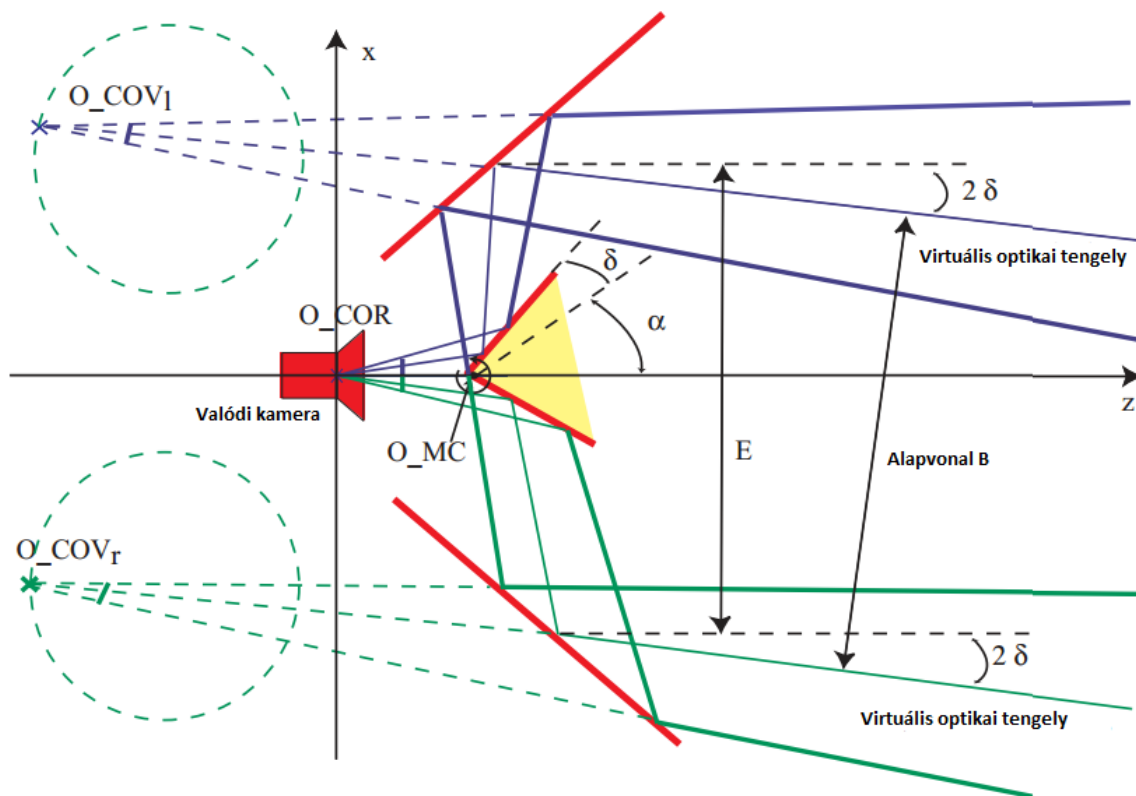
3.1.3.1. Értékelés

Ez a módszer kétkamerás rendszereknél jobb eredményt ad, mint egykamerás rendszereknél, így esetünkben a használhatósága megkérdőjelezhető. Mivel a módszer igen számítási igényes, ezért a továbbiakban nem fogjuk alkalmazni.

3.1.4. Akadálydetektálás egy kamerával és sztereo szenzorral

Katadioptrikus sztereólátáson alapuló rendszert használnak a [4] dolgozatban, ami egy kamerát használ, és arra tükörrel vetítik rá a két képet, így megfeleltetve a szenzort. A 2-es ábrán látható egy ilyen tükrös elrendezés felépítése. Az alap installáció egy mozgatható tükörrel van felszerelve középen, ami forgatható a tengelye körül a jobb kalibrálhatóság érdekében. Ez a tengely az ábrán O_MC-vel van jelölve. Az alábbi konfigurációban is igaz a pinhole modell a kamerára, azzal a megkötéssel, hogy nem biztos, hogy a két félkép ugyanakkora vízszintes felbontású lesz, ezt lehet korrigálni utólag a képek kivágásával.

Az így létrejött megvalósításban a középső tükör mozgásával az optikai tengelyek mindvégig párhuzamosak maradnak. Kísérletet végeztek arra, hogy milyen hatása van a valós és virtuális kamera tengelyei által bezárt szögnek. Arra jutottak, kis hibáknál a rendszer még stabilan, jól működik. Ezek után következett a diszparitás térkép elkészítése és végül annak meghatározása, hogy melyek azok a területek, ahol nagy intenzitásváltozás van. Ezen területek mellett összefüggő, azonos intenzitású területeket kerestek, így szegmentálva a képet, és határozták meg az akadályok helyzetét. Miután azokat megjelölték, megtervezték a kikerülés irányát.



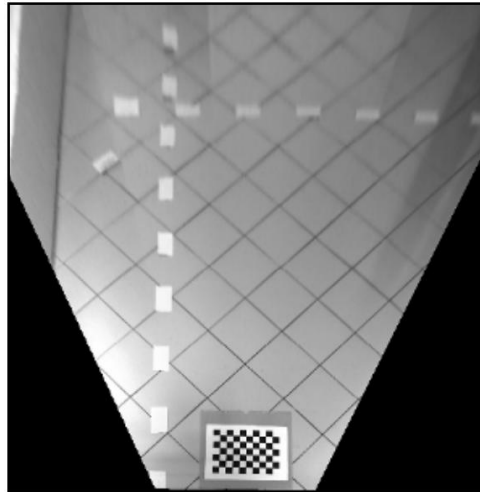
2. ábra Katadioptrikus „optika” elméleti ábrája. Forrás: [4]

3.1.4.1. Értékelés

Ez a felállás ugyanúgy működik, mint egy sztereo látáson alapuló konstrukció, azzal az előnnyel, hogy nem kell két kamera képét feldolgozni, és kompaktabb kivitel lehet vele elérni, mint sztereó látásos installációnál. Előnye még, hogy egy szenzorról kapjuk a képi információkat, így nem kell a képek intenzitás, szintér különbségével foglalkoznunk, és egyszerűbb a képek megfeleltetése is kalibrálás után. Hátránya viszont, hogy pontos konfigurálást igényel.

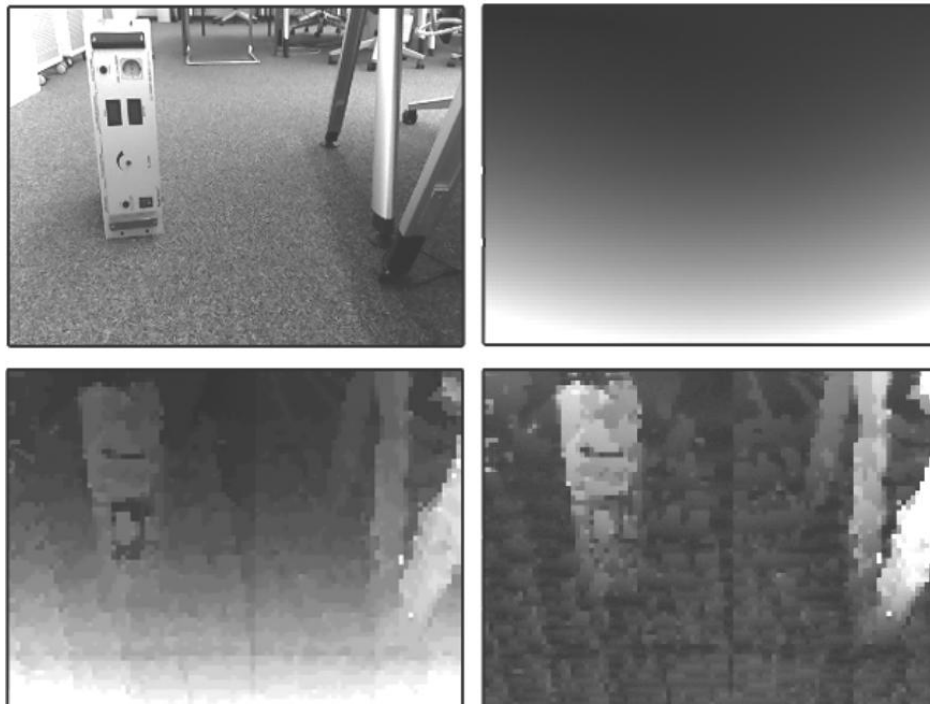
3.1.5. Kerekes robotok látás alapú akadályérzékelése

Ezt az akadályfelismerő rendszert [5] beltéri környezetre találták ki. Lényege, hogy a kamera képét egy kétdimenziós felülnézeti képpé transzformálja, ami a 3-as ábrán látható, így adva „térképet” a robot kezébe, megkönnyítve az akadályok kikerüléséhez szükséges irány- és útvonalválasztást.



3. ábra Felülnézetté transzformált kép. *Forrás: [5]*

A rendszer arra a feltevésre épül, hogy a robot csak síktalajon mozog, nem tud felfelé és lefelé mozogni a kamera. Ha bekalibráljuk a kamerát, vesszük annak a külső paramétereit, akkor ezek állandóak lesznek, és az a kalibráció meghatározza a föld síkját. Innen az inverz mátrixot képezzük, ezáltal tudjuk visszatorzítani a képet a talaj síkjának megfelelő alakúra, azaz így kapunk felülnézeti képet. Ezután, a kamerával egymás után készített két képből sztereo kameramodellt építünk fel. Ebből meghatározhatók a felülnézeti torzított képen a távolságok és fel tudjuk építeni a távolságtérképet az általános síkra. Majd a végén a két képből készített diszparitás térképet kivonjuk a talajra létrehozott modellből (4. ábra), és egy bizonyos küszöb alkalmazásával meghatározzuk az akadályokat, ezeket átranszformáljuk felülnézeti képpé, így kapjuk meg a végleges akadálytérképet felülnézetben.



4. ábra Az akadály detektálás folyamata. *Forrás: [5]*

3.1.5.1. Értékelés

Ez a módszer igen jónak mondható, mivel mozgás közben azonosítja az akadályokat. Előnye, hogy pontos detektálást tesz lehetővé, és viszonylag gyors is. Hátránya, jócskán maradnak zajok a rendszerben, amit elég műveletigényes kiszűrni, tekintve azt, hogy nem lehet egy jól meghatározott küszöböt alkalmazni, mivel ebben az esetben az akadályok hozzánk legközelebb eső pontja, azaz az alja a küszöbölésnek áldozatul eshet, és így a jármű irányítása kissé bizonytalanná válhat. Emiatt ebből az akadálydetektálást nem fogjuk alkalmazni, de a felülnézeti kép létrehozását igen.

3.1.6. Autonóm robot valós idejű akadályelkerülése aktív szenzoros, vertikális lézervonal kivetítéssel

A [6] projektben egy lézeres aktív szenzorral támogatott roboton valós idejű akadálydetektálást és elkerülést valósítanak meg.

A roboton két kamerát helyeztek el, de ebből csak az egyiket használták a lézerdetektálásra, mivel ezzel lehetett megfelelő pontosságot elérni. Az alkalmazott installációban a másik kamera a lézerforrás mellett volt, így azzal nem tudott elég nagy szöget bezárni. A lézerrel függőleges vonalat vetítettek a robot elé, így téve lehetővé az előtte elhelyezkedő akadályok detektálását. Azért volt fontos a függőleges elhelyezés, mert a robot igen magas volt, és egy esetleges magasan lévő akadályt fel nem ismerve, ha nekimegy, akkor felborulhatott volna. Mivel a lézer nagy intenzitással jelenik meg a képen abban az esetben, ha egy akadály kerül a robot elé, ezért az intenzitásból már könnyen detektálták azt. Azonban fontos volt, hogy a kép közepén jelenjen meg, mivel úgy volt kalibrálva a rendszer, hogy a kép közepének bizonyos részén talált lézervonal a robothoz közeli akadályokat jelenti. Ennek a rendszernek nem volt szüksége az akadályok térbeli pozíciójának pontos kiszámítására, mivel az előzetes kalibrálásoknál már meghatározták, hogy mi az a minimális távolság, ahol még biztosan el tudja kerülni azokat, és eszerint jelölték ki a kép közepén az akadályt detektáló keretet. Az elkerülési útvonal kiszámításához azonban célszerű volt háromszögeléssel távolságokat is mérni. Az útvonalat úgy határozta meg, hogy amennyiben a robot akadályt érzékelt maga előtt, akkor körbefordult és megnézte, merre tudna tovább haladni. Ha talál ilyen irányt, akkor az eddigi haladási pályától eltérő akadály mentes irányban folytatja tovább az útját.

3.1.6.1. Értékelés

Ez a rendszer nagy hatékonysággal működik. Előnye, hogy az akadályokat igen pontosan detektálja, és meg tudja adni a tőlük lévő távolságot is. Mindemellett biztosítja, hogy a robot ne ütközzön neki semmilyen magasan elhelyezkedő akadálynak. Mivel igen gyorsan képes a lézer helyzetét meghatározni, így valós idejű feldolgozásra alkalmas. Hátránya, hogy az akadály elkerülő algoritmus igen kezdetleges és túl sok heurisztikát tartalmaz.

3.1.7. 3D objektumdetektálás országúti adatokból lézerszkennerek használatával

A [7] dolgozatban egy akadálydetektáló rendszert hoznak létre autonóm autók számára, ám itt nem kisebb robotokról, hanem egy a forgalomban is használható személygépkocsiról van szó. Három szenzor információiból állítja elő a szükséges adatokat. Egy videó kamera képéből, LiDAR-ből (Light Detection and Range radar) és egy ultrahangos szenzorból. A kamerával kiszűrjük az útra jellemző színeket, így már csak azon a térrészen dolgoznak tovább. A lézer szkennerek 90 fokos látószögű, soronként 180 ponttal rendelkező 75Hz-es szenzor. Az érzékelők által begyűjtött adatokat az alábbi módszerrel elemezték.

A háromdimenziós pontokat Descartes koordináta rendszerben ábrázolták. Az x tengelyen megmaradtak az x értékek, az y tengelyen a z értékeket tüntették fel. Ebből az ábrázolásból egy hisztogram jön ki, ahol lesznek kiemelkedő csúcsok, és ezek a csúcsok fogják reprezentálni az akadályokat. Normalizálják a „hisztogramot” és így még jobban kiemelkednek a csúcsok, amiket ezután már egyértelműen meg tudunk határozni.

3.1.7.1. Értékelés

A módszer előnye, hogy eléggé pontosan meghatározza az x koordináták mentén az akadályok pozícióját, és a többi szenzorral ezeket tovább tudják finomítani. Hátránya, hogy az alkalmazott LiDAR szenzor igen drága, így ezt a saját projektünkben nem tudjuk alkalmazni.

3.2. Térbeli szkennelés

3.2.1. Automatikus 3D modellezés képszekvenciából

A [8] rendszer alapja a „projektív rekonstrukció”, az „önkalibráció”, és a „sűrű mélységbecslés”.

3.2.1.1. A folyamat leírása

- Összekapcsoljuk a különböző képeket.
- Páronként megkeressük az epipoláris geometriát.
- A kezdeti rekonstrukció az első két képpel kezdődik.
- Az alszekvencia képei számára a kamera pozíció megbecsülésre kerül a projektív keretben, amit az első két kamerakép definiál.
- Minden további képen megkeressük a jellemző pontokat és az új pontokat hozzáadjuk a rekonstruált képhez, majd újradefiniáljuk és korrigáljuk azokat, ezért nem fontos, hogy az első pont végig rajta maradjon a képen.

A folyamat eredménye a párezer felismert pont és a kamera pozíciók.

Ezután következik a mérési bizonytalanság csökkentése mind a kulcspontok pontosításával, mind a kamera általi torzítás kiszűrésével, melyhez a kamera mátrixát használják. Ekkor már ismertek a kamera pozíciók, a kamera mátrixok, és a detektált pontok, így már csak egy sztereolátásos algoritmust kell lefuttatnunk ahhoz, hogy vissza tudjuk állítani a térbeli helyzeteket. A kulcspontok ismerete abban segít, hogy az egyes ismétlődő elemeket kiszűrjük, vagy pontosítsuk a lokalizációt.

3.2.1.2. A projektív rekonstrukció

Először a képek teljesen függetlenek, azaz a képen látható alakzatok jellemzői máshol jelennek meg a két képen azzal a kikötéssel, hogy az adott képek nem térnek el túlságosan egymástól. A szomszédos képekből származó pontok hasonlóknak látszanak, így könnyű automatikusan megtalálni az egyezéseket.

3.2.1.3. A képek kapcsolatai

A két kép megfeleltetése nem valószínűsíthető meg pixel szinten, mivel ez túl nagy műveleti igényű és nem biztosított, hogy ugyanazok a részek látszanak mindkét esetben: valami az egyik képen takarásban lehet, a másik képen viszont nem. A jellemző pontokat a Harris sarok detektálóval célszerű megkeresni, majd ezeket egyeztetni a két kép között. Mivel a két kép nem nagyon tér el egymástól, ezért a két pontot egymás közelében keressük, és nézzük az intenzitás értékét. Mivel ez sem nagyon változhatott, ennek megfelelően adjuk meg a pontpárokat. Az így kapott eredmény

nagymértékben eltérő részeket is tartalmazhat, ezért célszerű szűrni az adathalmazt, amire a RANSAC algoritmust alkalmazták.

3.2.1.4. A kezdeti rekonstrukció

Az első két képet használjuk a referencia váz meghatározásához. Ezt a vázat először az első kamerához igazítjuk. Ezután meghatározzuk a másik kamera állását a jellemző pontokból és azok elmozdulásaiból, majd az így kapott két kameraállásból háromszögeléssel számítjuk ki a pontok térbeli helyzetét, és eltávolítjuk azt. Ez adja meg az előzetes rekonstrukciót. Ezt folytatva elvégezzük a további képekre is, így minden egyes kameraállást meg fogunk kapni a kiindulási ponthoz viszonyítva.

3.2.1.5. A rekonstrukció metrikussá konvertálása

Ebben a pontban a kamera belső mátrixával (amit a kamera kalibrálása által kaphatunk) korrigáljuk az eddigi értékeket, így történik meg a pontok valós helyzetének meghatározása.

3.2.1.6. Sűrű mélység becslése

Az előző lépésekben csak a kulcspontok helyzetét határoztuk meg, ami eléggé hiányos modellt eredményez, a modell részletességének növeléséhez két út vezet, az egyik az interpoláció, ami nem mondható jó megoldásnak, a másik a két kép minél több pontjának megfeleltetése, és azok térbeli pozíciójának kiszámítása.

3.2.1.7. Modell felépítése

Az így kapott pontokból a modell már felépíthető és textúrázható. A modellalkotáshoz a Delaunay háromszögelést alkalmazhatjuk.

3.2.1.8. Értékelés

Ez a módszer jól használhatónak tűnik, bár eléggé számítás igényes a valós idejű futtatáshoz, de jól optimalizált kód esetén lehetőség van rá. Előnye, hogy nemcsak a tér 3D-s vázát építi fel, hanem a kamera pozíciók által útvonalat is rögzít, így ezt is fel lehet használni utólagosan az ellenőrzéshez, és metrikus rendszerben kapjuk meg a pontok távolságát, azaz lehet kalkulálni belőle tárgytávolságot is.

3.2.2. MoBRo, háromdimenziós környezet leképző robot

A MoBRo [9] projektben egy háromdimenziós környezet leképző robotot hoztak létre, vizsgálva az egyes térleképző algoritmusok hibáit, lehetőségeit. A dolgozat tartalmazza a hardveres implementáció részleteit is, külön kitérve a mikrokontroller tervezésre és a kommunikáció megvalósítására. A térleképző algoritmusokat mélyebben vizsgálja, és részekre bontja az egész folyamatot az alábbi lépések szerint:

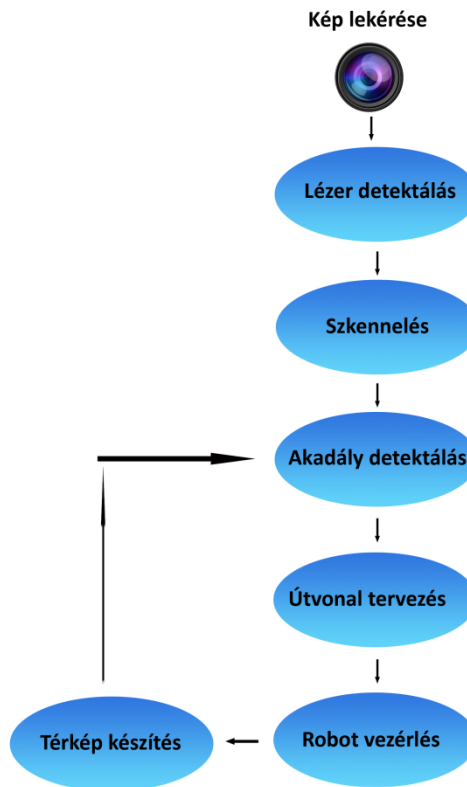
1. Jellemző pontok detektálása
2. Pontok követése
3. Pontpárok meghatározása
4. Kamera kalibráció
5. Pontok térbeli elhelyezkedésének kiszámítása

Megállapítható, hogy a [6]-ban említett módszer és ez a dolgozat ugyanazon a gondolatmeneten haladt végig, azaz nem térnek el sokban egymástól. A végeredmény szempontjából pedig csak a robot fejlesztése, megtervezése és kivitelezése tekinthető különbségnek. Azonban az így elérhető térképezési pontosság igen kicsi, így a lézerszkenneres módszerek felé érdemes elmenni, mivel azok valósidejű feldolgozás mellett igen nagy pontossággal kecsegtetnek.

3.2.2.1. Értékelés

A robotot távolról irányították, így nem volt autonóm, nem kellett akadályokat detektálnia, ellenben a 3D környezet leképezésével foglalkozott. Ezen a területen ért el eredményeket a rendszer.

4. A rendszer modulvázlata



5. ábra A rendszer modulvázlata

A megismert rendszerekből levont következtetéseink alapján saját alkalmazásunk két főbb részre bontható, lézerszkennelő modulra, ami az 5-ös ábrán a kép lekérését, lézerdetektálást és a szkennelést foglalja magába. A másik modul a robot irányításáért felelős, a többi részegység ide tartozik. Ez a rész a legfontosabb a robot mozgásának szempontjából, mivel itt kerül detektálásra minden akadály, itt történik az útvonaltervezés és a továbbfejlesztés során a térképkészítés is.

5. A rendszer feladatai

5.1. Képalkotás

A képek lekérése az OpenCV könyvtár segítségével történik. Ezzel a könyvtárral lehetőségünk van többféle kamera beállításra, a képvétel és a kamera inicializálása is meg van valósítva. A képek lekérésénél fontos, hogy milyen kamerát használunk, mivel annak minősége nagyban befolyásolja a képfeldolgozás további fázisaiban elérhető eredményeket. Az általunk épített rendszerben a

költséghatékonyság miatt csak egy ipari kamerát alkalmazunk. A kamerán változtatható rekeszállású és fókuszálható objektívet használunk a jobb eredmény elérése érdekében. A kamera CMOS típusú érzékelővel rendelkezik, és rolling shutter rendszerrel, azaz nem szinkronizált a képlekérés, vagyis nem minden esetben az első pixeltől kezdődik a szenzor kiolvasása. Ez hátrány lehet, amennyiben hirtelen fényváltozás történik. Mivel a kamera 30 képre képes másodpercenként (fps), és mi csak 25 fps-sel használjuk, jó esetben, ez nem befolyásolja negatívan a rendszerünket. A használt objektív nagy fényerejű f2-es, és kis optikai torzítással rendelkezik. A kamera 1600*1200-as felbontású 30 fps mellett. Az auto gain-t ki lehet kapcsolni, így nem változik az egyes képkockák között az érzékelő globális intenzitás értéke.

5.2. Akadálydetektálás

Az akadályok detektálását a 3.1.6 fejezetben megismert rendszerhez hasonló módon, lézer segítségével próbálom megoldani, azzal a különbséggel, hogy a lézert vízszintes vonal kivetítésére használom, mivel a robot nem túl magas, nem kell foglalkoznom a magasban elhelyezkedő tárgyak detektálásával. Az akadályokat többféle módon lehet lézerrel detektálni. Az egyik megközelítés a képen megtalált lézerpontok alapján detektál, míg a másik, a térbe „visszatorzított” pontokat használja fel erre. A dolgozatban mindkét lehetőséget meg fogom vizsgálni.

5.3. Kikerülés irányának meghatározása

Az adott irány meghatározására több módszer létezik. Vehetjük a feldolgozott kép közepét a haladási iránynak, ez egy jól kalibrált rendszerben igaz is lehet, de kalibrálás nélkül rossz eredményt adhat. Ettől a középponttól való eltérést határozzuk meg az akadályok esetében. Ha a robot a kép középpontjánál akadályt detektál, akkor megnézzük, hogy milyen irányban van eltérés, és a további akadályokat is megvizsgálva meghatározzuk a kitérés irányát. Ezután az akadály mozgását folyamatosan figyelemmel kísérjük egészen addig, míg el nem haladtunk mellette. Itt szétválik a feldolgozás menete. Két szemlélet is lehetséges. Az egyik, hogy folyamatosan figyeljük a tárgyat és kikerüljük. A másik, hogy megpróbálunk egy útvonalat becsülni a kikerüléshez. Az előbbi pontosabb eredményt ad, az utóbbi viszont gyorsabb feldolgozást tesz lehetővé, viszont a navigálást nehezítheti, ha már több akadályt is detektált a rendszer. Ilyenkor célszerű a kettő ötvöztetését használni, azaz megbecsülni az útvonalat, és amikor jön egy újabb akadály, valamint még nem hagytuk el az előzőt, akkor a becsült útvonalat az új akadálynak megfelelően módosítani.

5.4. Robotvezérlés

A robot vezérlése soros porton keresztül történik. A megvalósításhoz szükségünk van soros porti kapcsolatra. A kapcsolat kiépülése után küldhetjük ki az üzeneteket a robot számára. A roboton mindkét tengely külön forgatható, így nagyobb kanyarodási ívet tudunk elérni, és a meghajtás is tengelyenként történik, ami lehetőséget adhat az erő elosztásra. Mivel kísérleti fázisban egyes parancsoknak végzetes hatása lehet, például a túl nagy sebességgel történő megindulásnak, ezért külön be kell kapcsolni a meghajtás hídját, e-nélkül a motorok nem kapnak áramot. A robot induláskor mindig alapállapotba állítja a kerekeket. Lehetőség van az eszköztől lekérdezni a megtett távolságot, amit cm-es pontossággal képes visszaadni. Ezt fel lehet használni a saját pozíció meghatározására. Ilyen lehetőség a kormányzásnál azonban nincsen.

5.5. Robotirányítás

Az irányításért egy külön modul felelős, ami több egység adatait dolgozza fel, és ezekből kalkulálja a megfelelő értékeket. Fő feladata, hogy a robot pozícióját a térben figyelemmel kövesse, a robot

mozgását meghatározza, a készülő térképet összefűzze, a térbeli ponthalmazt a megfelelő állásba transzformálja, és abból információt nyerjen ki. Ezt az információt felhasználva pedig pontosabban tudjon akadályokat detektálni és útvonalat tervezni. A modul feladata még, hogy térképet készítsen a robot környezetéről, ezen a térképen kövesse annak mozgását számon tartsa helyzetét és folyamatosan frissítse a rajta lévő akadályokat.

5.6. Lézerpontok detektálása

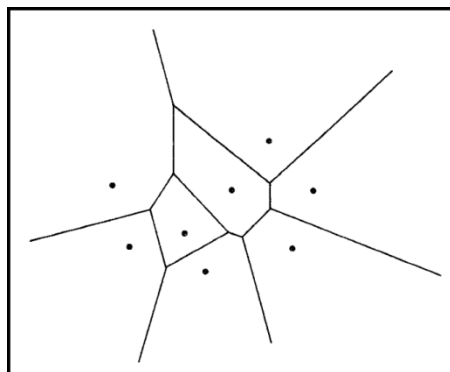
A lézerdetektálást is külön modul végzi, aminek nincs más feladata, csak a lézervonal megtalálása, és az esetleges hibák kiszűrése, azaz a lézer minél pontosabb azonosítása. Itt szükségünk lehet szubpixeles keresésre is, hogy minél pontosabban meg tudjuk határozni a detektált elemek térbeli elhelyezkedését. A detektálást három módszerrel teszteljük, ezek közül a legjobbat fogjuk alkalmazni. A három módszer a különbségkép, szintér alapú és a leíró alapú.

5.7. Pontok visszatranszformálása a térbe

A pontok visszatranszformálása bonyolult művelet, és az elméleti háttérét a 6.3-as fejezet tartalmazza. A transzformáció során feltételezzük, hogy ismerjük a kamera belső paramétereit [10] és a lézer síkját a kamera koordináta rendszerében. Ezt felhasználva a képen található lézer pontokról már egyértelműen megmondható, hogy a térben hol helyezkednek el. A transzformáció során ügyelünk arra, hogy a kamera torzítását is kezeljük, így növelve a pontosságot. A pontok térbeli helyzetének meghatározása arra az elvre épül, hogy a lézernek ismerjük a kamera koordináta rendszerében vett sík egyenletét. Erre az ismeretre alapozva történik a transzformáció. Ehhez szükséges a kamera kalibrációja és a lézer síkjának a meghatározása.

5.8. Modellalkotás

A modellalkotásra több algoritmust lehet használni. Esetünkben érdemes a Delaunay háromszögelést alkalmazni, mivel az ponthalmazokra hatékony. A ponthalmazok háromszögelése részben megoldott probléma, nehézséget a lyukak jelentenek. A probléma onnan fakad, hogy az algoritmus optimalizálja a ponthalmazt, és befedi a lyukakat is az optimalizálás következtében. Erre megoldás lehet, ha annak megfelelően paraméterezzük az algoritmust, hogy mekkora méretűek lehetnek a kitöltetlen részek. A Delaunay [11] háromszögelésnek létezik háromdimenziós kiterjesztése is. Ez a módszer a Voronoi [12] diagramokra épít. A Voronoi diagramot kétdimenziós esetben úgy kell elképzelni, hogy minden egyes pont és a közvetlen szomszédja között húzunk egy egyenest, ami a két pontot összekötő egyenesre merőleges és azoktól egyenlő távolságra van. Utána az egyenesek azon részeit tartjuk meg, ami a metszések által valóban megmarad, azaz azokat az élrészeket, ami a két pont közé esik. Egy ilyen diagramot láthatunk a 6-os ábrán.



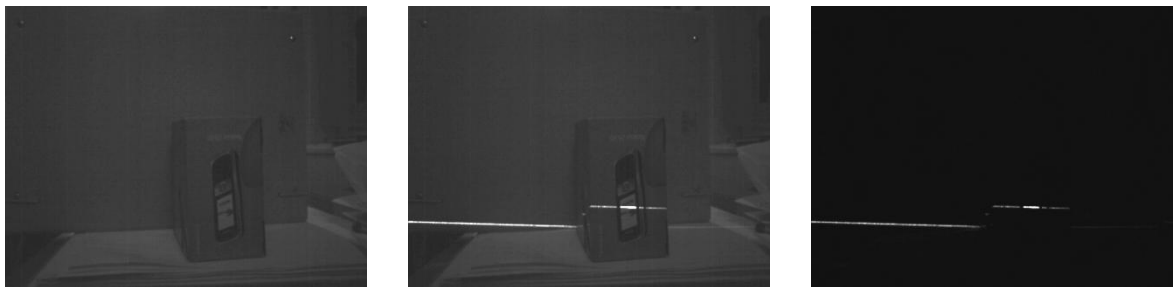
6. ábra Voronoi diagram Forrás:[12]

A textúrázás külön probléma. Mivel több képből áll elő a modell, lehetőség van egy szuperrezolúti textúra felépítésére, vagy akár a jellemző pontok alapján, a pozíciókat ismerve, az adott térrészre a legjobban illeszkedő textúrát is ki tudjuk választani. Ennek hátránya, hogy erőforrás igényes, így a valós idejű feldolgozásnál nem alkalmazható.

6. Lézerszkennerek megvalósítása

6.1. Lézerdetektálás

Alapvető probléma a képen megjelenő lézervonal detektálása. Ennek deformitásából a szenzor előtt elhelyezkedő alakzatokra és azok térbeli helyzetére tudunk következtetni. Erre a feladatra három megoldást találtunk: különbségképzéses módszert, szintéren és éldetektáláson alapuló megközelítést, valamint leíró alkalmazó technikát. Az első valós időben csak külön elektronika segítségével alkalmazható, mivel a különbségképek alapján határozza meg a lézert. Ez a lehető legrobosztusabb algoritmus, de ehhez fel kell gyűjtani a lézert, készíteni egy képet, majd a lézer kioltása után szintén képet kell venni. A két kép különbsége adja meg a lézer helyzetét, amin belül még meg kell határozni a lézervonal középpontját. Tehát itt egy detektáláshoz két kép elkészítésére van szükség. Maga a kivonás gyors művelet, de a kamera sebessége miatt ezt nem alkalmazzuk, mivel a valós időben való működéshez minimum 50fps sebességgel rendelkező kamerára lenne szükség, ami nem áll a rendelkezésünkre. A másik probléma, hogy a robot mozgatását „szaggatottan” kéne tervezni, hogy a két kép készítése között ne legyen elmozdulás. A 7-es ábrán látható a megközelítés eredménye. Baloldalon a kikapcsolt, középen a bekapcsolt lézerral készült kép látható, jobboldalon a két kép különbsége.

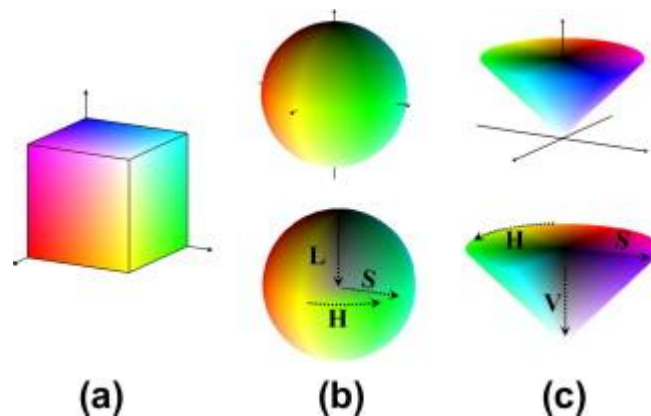


7. ábra Két kép lézerral, lézer nélkül és azok különbsége

6.1.1. Szintéren és éleken alapuló detektálás

Ennél a detektálásnál először a színre szűrünk. Itt lehet zöld, piros, kék lézert használni, és ennek megfelelően az adott színt vizsgáljuk. Egyik lehetőség, a gyorsabb módszer, ha vesszük külön a lézer színének megfelelő színcsatornákat és ezeket értékeljük ki. Ennek előnye, hogy itt csak intenzitás képet kell elemezni, és a megadott értékek közötti eredményt kell már csak vizsgálni. Hátránya, hogy nem túl pontos a módszer, mivel nem tudjuk tökéletesen meghatározni a színt ténylegesen, és a többi színcsatorna információit eldobjuk. Megoldás lehet a HSV [13], vagy HSL szintér alkalmazása is [8. ábra]. Ezek a szinterek az emberi észlelésnek megfelelően határozzák meg a színeket, a világosságot, a telítettség, és maga a szín alapján. Így például a vörös szín nagy pontossággal kiszűrhető. Hátránya azonban, hogy ez a konverzió számításigényes, rengeteg benne a szinusz és a koszinusz hívás, ami

elégge lelassítja a folyamatot. Ugyan ezt javítani lehet lookup table-lel, de a sok érték előzetes kiszámítása jelentős memóriát foglal, így annyira nem jó megoldás. Lehetséges még az egyes értékek kerekítésére, így kisebb táblát is alkalmazhatunk, ám ekkor sok információt veszünk, és a színtartományok határai torzulhatnak.



8. ábra RGB, HSL, HSV színterek Forrás: [14]

Ha sikerült meghatározni a lézer színtartományába eső képrészeket, akkor minden vörös színt, ami a képen megjelenik, lézernek detektálnánk, akár egy tűzoltó készüléket is. Ezért elkészítjük a feldolgozandó képünk éltérképét, mivel ezzel tudjuk meghatározni, hogy hol lehet a lézer valójában. Ez többféle módszerrel is kivitelezhető. Végig mehetünk Sobel operátorral a képen, így nyerve ki az adott orientáltságú éleket, vagy csinálhatunk Laplace éldetektálást. Ez utóbbinál akár fel is cserélhetjük a műveletek sorrendjét. Azaz előbb végezzük el a Laplace éldetektálást, és utána az így kapott képen keressük a piros színt. Ez sokat gyorsíthat a rendszeren, mivel az élt nem tartalmazó pixelek feketék, vagy ahhoz közeli értékűek, ezért csak a világosabb pixeleket kell transzformálni az adott színtérbe, amit gyorsabban el lehet végezni.

Amennyiben a Sobel operátort használtuk élkeresésre, ha végeztünk a lehetséges színek és az élek detektálásával, már csak a két eredményhalmaz metszetét kell képezni és feltételezhetjük, hogy ez az eredmény a lézer pontokat fogja magában tartalmazni. A I. táblázatban látható a két módszer a futási ideje. A táblázatban szereplő értékek az egy kép elemzésére fordított időt mutatják.

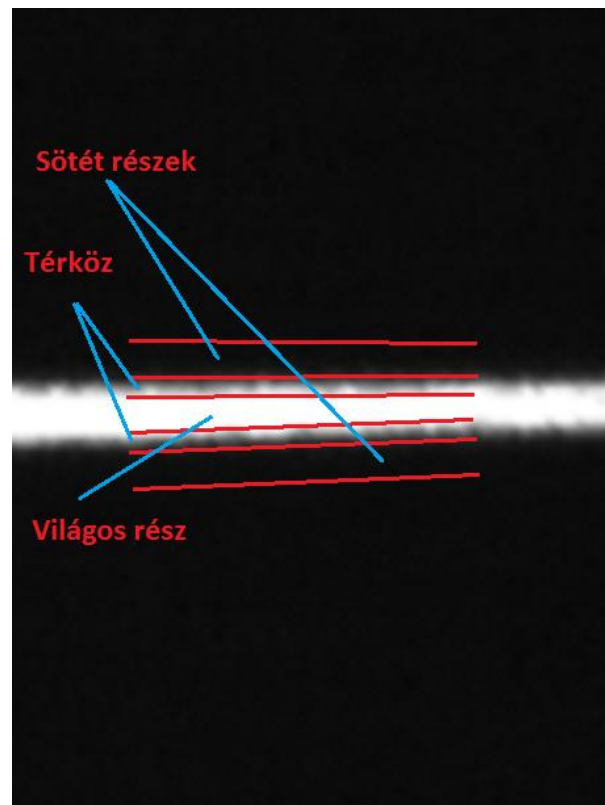
Képszám	1	5	10	15
Laplace	71ms	69ms	68ms	68ms
Sobel	320ms	306ms	300ms	301ms

I. táblázat Laplace és Sobel módszer összehasonlítása

Természetesen a módszer könnyen hibázhat, mivel ha a kameraképen szerepel egy piros tárgy, akkor ott is él keletkezik. Ezt úgy lehet javítani, hogy minden oszlopban csak egy lézerpontot engedünk meg, és az esetleges több találatból kiválasztjuk a leginkább megfelelőt. Ellenben látható, hogy a kezdeti képen elég sok számításigényes algoritmust kell végrehajtani. Ebből a színtér transzformáció a leglassabb a már említett szinusz és koszinusz műveletek miatt, az éldetektálás sokkal gyorsabb, ám az összesített eredmény futási ideje még így sem közelíti meg a kitűzött 50ms-os határt. Emiatt a továbbiakban nem használjuk ezt a módszert, mivel a tervezett rendszerünknel a futási idő elsődleges korlátot jelent.

6.1.2. Leíró alapú detektálás

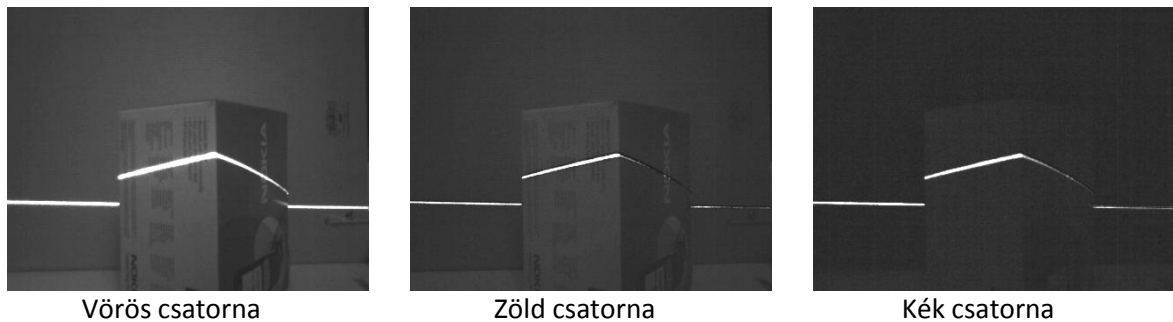
A leíró alapú módszer azt tételezi fel, hogy a lézer megjelenése jellegzetes a kamera képén. Ez a lézer fizikai tulajdonságaiból fakad. A közepén nagyobb az intenzitása, ami a szélek felé közelítve csökken, így a fényesebb részek lesznek a lézervonal középpontjai. Ez a jellemző a 9-es ábrán kerül szemléltetésre.



9. ábra Lézervonal profilképe

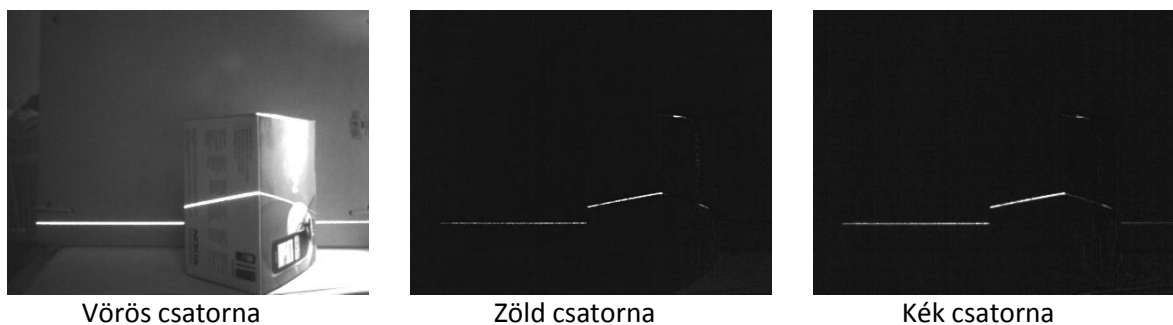
A lézer képe itt három részre bontható, egy sötét, egy világos és egy közbülső, térköz részre. A három rész lézercsíkra merőleges intenzitás értékei egy görbével jellemezhetők, de mivel ezek elemzése és illesztése lassabb, nagyobb számítási kapacitást igényel, ezért egy négyszögjellel „közelítjük” Ez a négyszögjel a 9-es ábrán látható részeket vonja össze úgy, hogy kihagyja a térközt, ami az átmenet lenne a görbében, és így csak közelítő „négyszögjel” jön létre. Ebből úgy kapjuk meg a legjobb illeszkedést, hogy vesszük a világos rész intenzitásainak az összegét, abból kivonjuk a sötét részek intenzitás összegének átlagát és a két sötét rész intenzitás átlagának a különbségét, majd az így kapott értékek maximumát vesszük. Ahol a maximum található a képoszlopban, ott van a lézer közepe. Az így kapott eredményeket még szűrni kell az alapján, hogy a maximum helyen mekkora intenzitás értéket vesz fel a lézer. Ha ez az érték túl kicsi, akkor arra következtethetünk, hogy valószínűleg az adott oszlopban nem is volt lézervonal, így ott eldobhatjuk a talált pontot. Ezzel tovább növelhetjük a helyes találatok számát. A módszer hátránya, hogy az összes fényes élt, ami lézerszerű, azt lézernek fogja detektálni. Azonban több lehetőségünk van a találatok pontosítására. Az egyik, ha színes képen vizsgálva a részletet csak a vörös csatornát használjuk fel. Mivel a lézerünk vörös színű, ezért feltételezhetjük, hogy az intenzitása ezen a csatornán lesz a legnagyobb. Ebből kiindulva már jobb eredményt érhetünk el, ám még mindig maradnak olyan élek, amik a vörös csatornán elég erősek és a lézervonal jellegzetességével rendelkeznek. A 10-es ábrán láthatók az egyes színcsatornák

intenzitásképként ábrázolva külön-külön. Kivehető, hogy a vörös csatorna hordozza a legtöbb információt a lézerről, ám a kép többi részéről is ez a csatorna rendelkezik a legtöbb információval.



10. ábra Lézerkép színcsatornái intenzitásképként ábrázolva

A többi szín kiküszöbölésére vörös szűrőt használhatunk, hogy a többi csatornán is csökkentsük az egyéb színek megjelenését. Ez a színszűrő képességitől függően hardveres támogatással kezelhető, ami gyorsítja a lézertetkálás folyamatát, mivel nincs szükség szoftveres lépésekre, azaz a számítási kapacitást nem kell terhelni. A vörös színszűrő előnye, hogy elnyomja a fény nem vörös tartományába eső sugarait, és erősíti az adott spektrumot. A 11-es ábrán megtekinthetjük a különböző csatornákat a vörös szűrő alkalmazása esetén. A képeken látható, hogy a lézer azért a többi képen is megjelenik, de csak az erősen megvilágított részeken. Ez azért lehetséges, mert a színszűrőn átjutó fény olyan nagy intenzitású, hogy fehérnek érzékeli a szenzor, ami tartalmaz zöld és kék összetevőket.



11. ábra Lézerkép színcsatornái vörös szűrő használata esetén

Szinte már csak a vörös csatorna hordoz információt, a zöld és a kék csatornákon csak a lézer, illetve az erősebb fényforrások jelennek meg. Ezt ki lehet használni a lézer pozíciójának pontosítására, de a képeken látható az is, hogy a kisebb intenzitású lézervonal rész a jobb oldalon csak a vörös csatornán található meg. Ebből következik, hogy ezt csak a nagy intenzitású lézervonal részeken tudjuk használni, amiket az algoritmus amúgy is jól detektál.

6.1.3. Leíró alapú detektálás továbbfejlesztése

A robot tesztelésénél felmerült az a probléma, hogy a vörös színtartomány igen nagy, sok számunkra szükségtelen információt, fényt enged be a szűrő. Ez a lézertetkáló modult zavarta, mivel a színszűrő következtében minden vörös színben látszott, ami a lézerhez hasonló eloszlású részeken jelentett gondot, mivel az algoritmus maximumkeresésen alapul. Így a fényesebb részek

„győztek” a valódi lézercsíkkal szemben. Ennek megoldására el kellett távolítani a színszűrőt. A tesztek azt az eredményt adták, hogy ha vettem a vörös csatornán lévő érték kétszeresét és kivontam belőle a kék és zöld értékeket, akkor jó közelítéssel csak a pirosas árnyalatú színek maradnak. Ekkor küszöbölést végezve kapjuk a lézerpontokat. A 12-es ábrán szemléltetjük a javított és eredeti algoritmusok eredményét. A detektált lézerpontokat mindkét esetben zöld szín jelöli.



Eredeti algoritmus



Javított változat

12. ábra Javított lézerdetektáló és az előző verzió eredménye

Az új algoritmusnál látható, hogy csak az ablak alatt téveszti el a lézert a neoncső miatt, a többi helyen viszont jobban detektálja a lézervonalat.

6.1.4. Szubpixeles középpont meghatározás

A lézer közepének detektálását már sikerült megoldani az előbbi modulokban, de ez csak pixel pontosan történt. Feltételezve azt, hogy a lézer tengelye a Föld felé néz ferdén, fentről lefelé, a kamera képsíkja pedig merőleges a Föld síkjára, akkor a kameraképen a lézer is fentről lefelé egyre távolabbi pontokat jelent. Azonban mivel a képen a lézer síkja nem a kamera képsíkjával párhuzamos, ezért egy-egy képpontra más távolság esik. A fenti pixelek esetében a távolságok változása lényegesen kisebb, mint a lent elhelyezkedő képpontoknál. Ebből fakad, hogy a pontosság távolabbi objektumoknál lényegesen csökken. Ezért van szükségünk a lézer közepének szubpixeles detektálására.

6.1.4.1. Meghatározás élek alapján

A módszer elve, hogy a lézer két szélén élek keletkeznek, ezeket az éleket könnyen detektálni tudjuk. Ha a kezdeti éltől visszább lépünk két pixellel és onnan egy $[-1, -1, 0, 1, 1]$ -et tartalmazó maszkkal végighaladunk minden pixelen a végső él utáni két pixelig, akkor megkapjuk a kép adott irányú elsőrendű deriváltjának közelítését minden „lézerpontban”, ami közelíthető egy görbével. Ha a görbe legnagyobb értékű pontját összekötjük a minimum pontjával, akkor az így kapott egyenes, és vízszintes tengely metszéspontjában lesz a lézer középpontja.

6.1.4.2. Meghatározás környezet alapján

A [14] szerint a lézer eloszlása valamilyen „csúcsos” függvénnyel írható le. Ennek köszönhetően feltételezhetjük, hogy a legnagyobb értékeknél lesz a középpont. Ha a középpontnak bizonyos környezetét vizsgáljuk, és kiszámoljuk a súlypontot, az a lézer közepét adja meg. A súlypontszámítás igen gyors, és itt ki tudjuk használni, hogy már megvan a pixeles középpont.

6.1.4.3. Meghatározás Gauss görbe illesztéssel

A lézer eloszlása jól közelíthető Gauss függvényvel. Így lehetőségünk van arra, hogy a középpont adott környezetében lévő intenzitás értékekre Gauss függvényt illesszünk [15]. Ebben az esetben a Gauss függvény csúcspontja határozza meg a lézer szubpixeles középpontját. Az irodalom és saját tapasztalataink alapján ez a módszer tekinthető a legjobbnak.

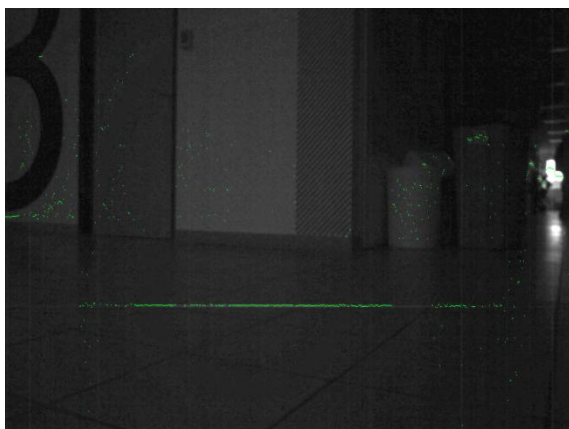
6.1.4.4. Értékelés

A fenti módszerek közül a Gauss-illesztés adja a legjobb eredményt, ám számításigényes, ami jelen esetben szűk keresztmetszet. Az él alapú módszer a második legjobb, mivel ennek a pontossága közel van a Gauss függvényéhez, azonban itt szükséges egy maszk művelet lefuttatása, és két egyenes metszéspontjának a meghatározása, ami megint elég sok számítást jelent. A súlypont alapú módszer hátránya, ha rossz helyre van detektálva a középső pixel, akkor az eredmény annak a környezetében lesz érvényes, azaz a sima lézerközeppon egy pixeles ingadozása az eredményben is közel egy pixeles ingadozást jelent, ami az előbbi módszereknél nincs. Ennek ellenére mi ezt használjuk, mivel a sebessége messze jobb a többi módszernél, és jó középpont találat mellett közel ugyanazt a pontosságot hozza.

6.1.5. Zajszűrés

A detektált pontok között még mindig vannak olyanok, amik nem tartoznak a lézerhez, hanem valamilyen becsillanást talált a detektáló. A probléma enyhítésének érdekében szűrjük az adatokat. Első lépésben teszteltük a medián filtert, ám azt tapasztaltuk, hogy ez sok esetben fals értéket határoz meg a pontok helyeként, és nem csökkenti a zajosságot, mivel az esetek nagy részében a zajos részek 5-30 pixel nagyságúak. Ilyen maszkmérettel használva pedig elmossa az eredeti középpontokat, és teljesen rossz eredményt ad. Kisebb maszkokat alkalmazva pedig másféle zajt generál. Az átlagoló szűrő sem hozott jobb eredményt, mivel a pontokat eltolta.

A jelenleg használt algoritmus a szóráson alapszik. Három egymás mellett lévő pont függőleges távolságának veszem a szórását, és ha ez bizonyos szint alatt van, akkor a középső pontnak ténylegesen lézer, ha felette, akkor zajnak minősül. Az így kapott információt egy külön tömbben tárolom, a végén a zajnak minősített pontokat megjelölöm. Ez a módszer a tényleges zajokat kiszűrte, az eredeti adatokat nem változtatta meg és futási ideje gyors. A 13-as ábrán látható a zajszűrés előtti és utáni kép. A lézernek detektált pontokat zölddel jelöltük mindkét képen.



Zajszűrés nélkül



Zajszűréssel

13. ábra Zajszűrő algoritmus eredménye

6.2. Kamerakalibrálás

A kamera kalibrálása által kapcsolatot tudunk teremteni a kamera pixelkoordinátája és a világkoordináták között. Enélkül nem lenne lehetséges a képen megtalált pontok térbe visszatranszformálása, bár ez önmagában nem elégséges, csak szükséges feltétel. A kalibrálás másik előnye, hogy megkapjuk a kamerán használt objektív torzítását, és lehetőségünk van ennek a kiküszöbölésére, így csökkentve a pontatlanságot.

6.2.1. Kamera- és világ koordinátarendszerek kapcsolata

A kamera és a világ között négy koordinátarendszer írja le a kapcsolatot. Ha az ezeket összekötő transzformációkat elvégezzük, akkor kapjuk meg az egyik rendszerből a másikba számított koordináta értékeket. A világkoordináták háromdimenziósak, míg a pixelkoordináták kétdimenziósak, így a kapcsolat nem kölcsönösen egyértelmű, mivel dimenzióvesztéssel jár.

A négy koordinátarendszert két részre oszthatjuk, külső kamera paramétereket és a belső kamera paramétereket meghatározókra [16]. A külső paramétereket az objektum- és a világkoordináta közötti, valamint a világ- és a kamerakoordináta közötti leképezés adja meg. A belső paramétereket a képsík és a pixel koordináta közötti kapcsolat írja le. A külső és a belső paraméterek között a kamera és a képsík közötti transzformáció teremt kapcsolatot.

6.2.1.1. Belső paraméter mátrix

A belső paraméter mátrix a kamera paramétereit tartalmazza. A mátrix kiszámításához [10] több képet veszünk a kamerával, és ezeken a képeken különböző helyzetben megjelenik egy kalibráló objektum. Az objektumon jól definiált pontok vannak, amiknek valós elhelyezkedését ismerjük. Így a képen látható pozíciókból, és az ismert valós helyzetekből már kiszámolhatók a paraméterek. A mátrix a 14-es ábrán látható. Az α_x , α_y a fókusztávolságot, az u_0, v_0 pedig az optikai tengelyt jelölik, γ pedig a pixel kétirányú méretének hányadosa.

$$K = \begin{bmatrix} \alpha_x & \gamma & u_0 \\ 0 & \alpha_y & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

14. ábra Belső paraméter mátrix. Forrás: [16]

6.2.1.2. Külső paraméter mátrix

A külső paraméter mátrix két részt tartalmaz, egy forgatási mátrixot és egy eltolás vektort. A külső paraméter mátrixot a belső ismeretében lehetséges csak kiszámolni, mivel a forgatási mátrix és az eltolási vektor csak az optikai tengely és a fókuszpont ismeretében határozható meg. A 15-ös ábrán r betűvel jelöli a forgatási mátrix elemeit és t betű az eltolási vektort.

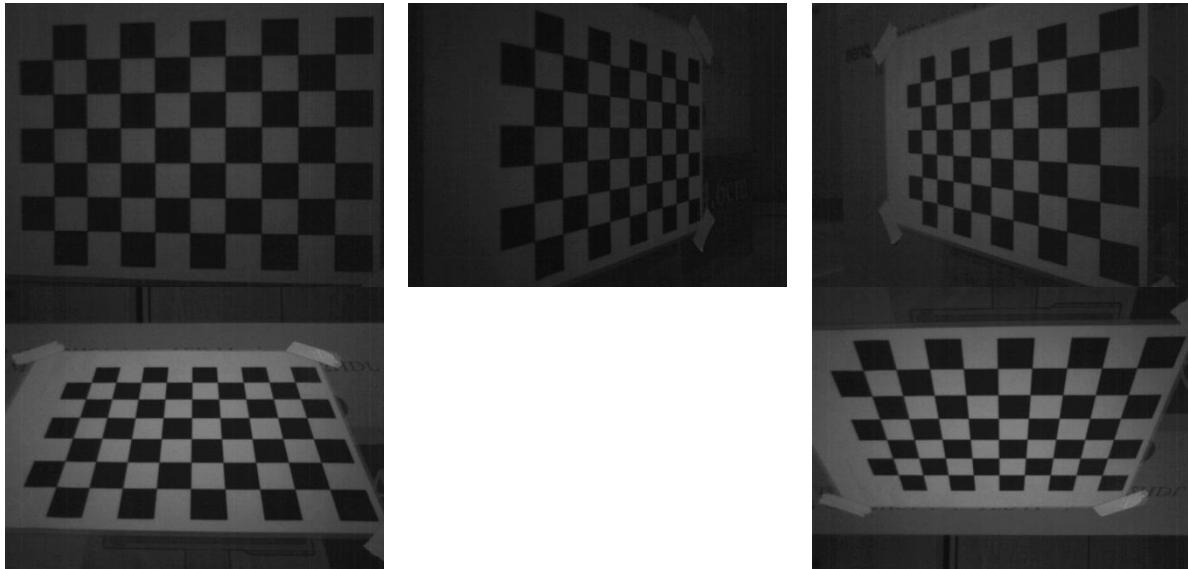
$$M_{extr} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix}$$

15. ábra Külső paraméter mátrix. Forrás: [16]

6.2.2. Kalibrálás folyamata

A kalibrálás folyamán egy sík 10*7-es sakktábláról készítettünk különböző orientációban képeket. Ezen képekből meghatározzuk a belső paraméter mátrixot, és ezt használjuk a továbbiakban.

Vizsgálataink azt mutatták, hogy a kalibráció pontos megvalósításához minimum 5 képet kell készíteni. Ezeknek a képeknek az elhelyezkedése a kamera képén nagymértékben különböző legyen. Ellenkező esetben az optikai tengely és a fókuszpont meghatározása nem megfelelően pontos. A 16-os ábrán láthatóak az általunk használt elrendezések.



16. ábra Kalibráló tábla orientációi

6.3. Rekonstrukció elve

A rekonstrukció arra az alapelvre épül, hogy tudjuk a lézer síkját a kamera koordináta rendszerében leírva. Ennek és a kamera mátrix ismeretében a képen talált lézer pontokat térbeli helyzetét kiszámolhatjuk, ami a rekonstrukciójukat jelenti.

A rekonstrukcióhoz egy rekonstrukciós mátrixot alkalmazunk, amivel megszorozva a talált lézerpontok pixelkoordinátáit, megkapjuk a pontok eredeti térbeli helyzetét. Ahhoz hogy ezt megtehessek, szükségünk van egy P mátrixra, ami felépítését tekintve megegyezik a kamera külső paraméter mátrixával, azaz tartalmaz egy forgatási mátrixot és egy eltolási vektort. Ahhoz, hogy a P mátrix forgatási vektor része a kamera szemszögébe képezze le a modellt, a főátlóban lévő elemeket csupa egyesnek kell venni, a többi elemet pedig nullának. Az eltolási vektor elemei pedig mind nullák lesznek. A rekonstrukciós mátrix egy négyszer négyes mátrix, ami magába foglalja a P mátrixot és a sík egyenletének paramétereit a 17-es ábrán látható módon. Itt p betűvel vannak jelölve a P mátrix elemei és A, B, C, D-vel a sík paramétereit.

$$M_{\text{rekonstrukciós}} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \\ A & B & C & D \end{bmatrix}$$

17. ábra Rekonstrukciós mátrix. Forrás: [16]

A rekonstrukcióhoz azonban ennek a mátrixnak az inverze szükséges. Ezen kívül szükségünk van még egy forgatási mátrixra, hogy ne a kamera síkjára képezzük le ezeket az adatokat, hanem a lézer síkjára.

A végső lépésben egy négyszer négyes mátrixba bele kell helyoznunk a lézer pontokat, ez a 18-as ábrán látható módon történik meg, majd ezt megszorozzuk a rekonstrukciós mátrixszal, és utána össze kell szorozni a forgatási mátrixszal, és így megkapjuk a lézerpontok térbeli helyzetét. A P_x , P_y jelenti a pont kamerakoordinátáit.

$$\begin{bmatrix} P_x & 0 & 0 & 0 \\ P_y & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

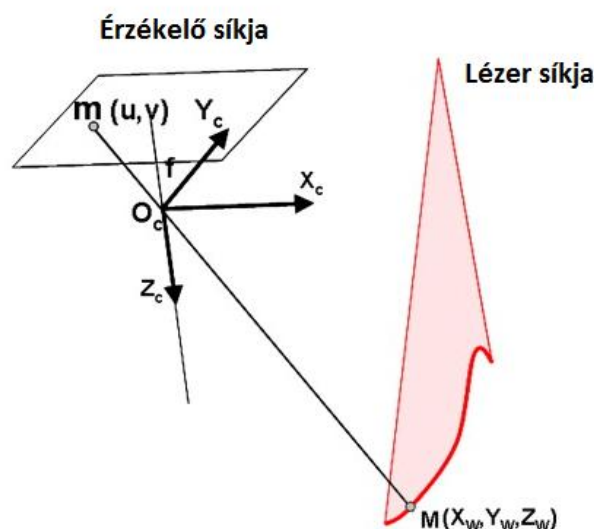
18. ábra Lézer pontok elhelyezkedése a 4*4-es mátrixban

6.4. Lézerkalibrálás

A használt lézer egy vonalat vetít ki, így lehetőségünk van arra, hogy a vonal és a forrás által meghatározott síkot a kamera koordináta rendszerében is leírjuk, majd felhasználjuk a rekonstrukcióhoz. A kalibráció célja, hogy a lézer síkját meghatározzuk, és annak paramétereit megkapjuk.

6.4.1. Kalibrálás elve

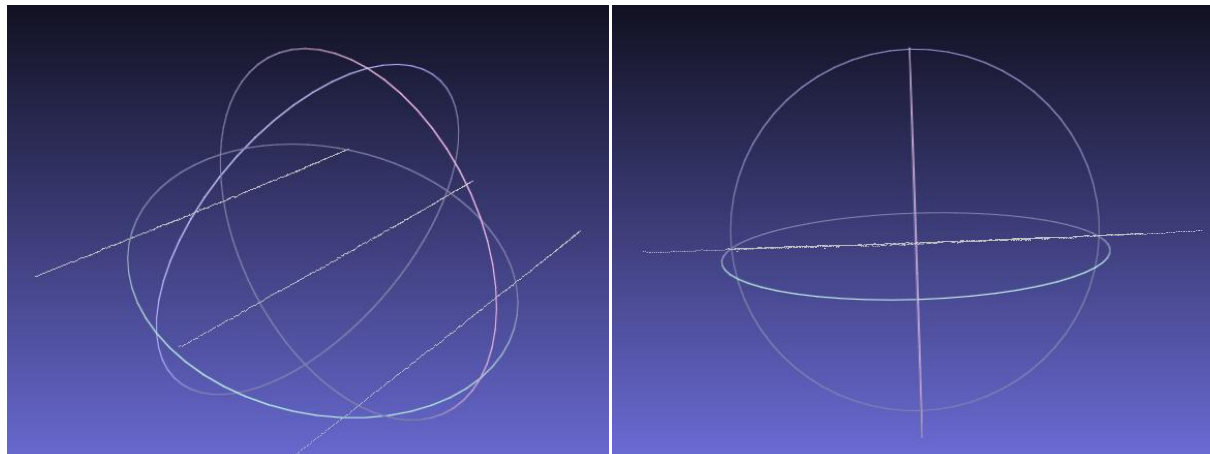
A lézer kalibrálásánál kihasználjuk, hogy egy adott sík egyenletét ismerve, a 6.3-as pontban bemutatott módon, a sík pontjait térbe tudjuk visszatranszformálni. A lézer síkját a kamerához képest a 19-es ábrán szemléltetjük.



19. ábra A lézer síkja a kamerához képest *Forrás: [17]*

Ha a kameránk már kalibrált, akkor egy sík ismert pontjai és a képen lévő pontjai alapján külső paraméter kalibrációt hajtunk végre. Az ismert sík a kalibráló tábla, ami esetünkben egy nyolcszor hetes vörös sakktábla, hogy ne nyelje el a lézert, hanem a legnagyobb mértékben visszaverje. A sakktábla külső paraméter mátrixából nyert rotációs almatríxából kinyerhető annak egyenlete. Ennek ismeretében rekonstruálnunk kell a sakktáblán megtalált pontokat, és az így kapott térbeli pontokat

eltávoljuk. Ezt több különböző távolságra megismételjük, így kapunk térbeli vonalakat. A térbeli vonalak elvileg egy síkra esnek, ez a 20-as ábrán látható. A baloldali képen látszik, hogy a vonalak párhuzamosak, a jobb oldalin pedig a vonalak egymásra vetítve láthatóak, amiből leszűrhető, hogy ténylegesen egy síkot alkotnak az egyenesek.



A detektált párhuzamos lézervonalak

Beforgatás után a párhuzamos vonalak egymásra vetítve látszódnak

20. ábra A detektált lézervonalak a tér dimenzióit szemléltető körökkel

A vonalakra regressziós síkot illesztünk [18], így kapjuk meg a lézer síkjának egyenletét. Ezt az egyenletet felhasználva rekonstruáljuk későbbiekben a lézerszkennert által detektált pontokat.

6.4.2. Lézerkalibrálás folyamata

A kalibrálásnál minden pozícióban két különböző képet kell készíteni, egy olyat, amin van lézer vonal, és egy olyat, amin nincs. A két képre azért van szükség, mert a lézeres képen nem tudjuk detektálni a sakktábla sarokpontokat. A két kép még abban is segít, hogy a lézer pozícióját biztosan meg tudjuk adni különbséget képezve belőlük. Ezután kiszámításra kerülnek a lézerpontok térbeli koordinátái. Ezt a műveletet összesen háromszor végezzük el és az így kapott pontokra regressziós síkot illesztünk, hogy megkapjuk a lézer síkját (20. ábra).

6.5. Lézerszkennert értékelése

A lézer szkennert több szempontból is lehet értékelni. Mi két szempontot veszünk alapul, mivel a jármű irányításánál és az akadályok elkerülésénél ezek a döntőek: a sebesség és a pontosság. A kitűzött cél egy valós időben dolgozó, azaz megközelítőleg 25 fps-t tudó rendszer volt, ami milliméter pontosan detektálja a tárgyakat. A kész modul mért értékei a II-es táblázatban tekinthetők meg.

Mérés száma	1	2	3	4	5	6
Kézzel mért távolság	61mm	127mm	206mm	378mm	536mm	942mm
Lézerrel mért távolság	61,285mm	127,195mm	206,775mm	378,972mm	536,894mm	941.319mm

II. táblázat Lézer szkennert mérési eredményei

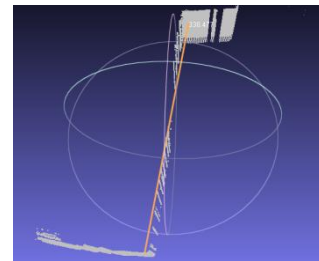
A pontosság meghatározása érdekében a lézerszkennert úgy teszteltem, hogy két téglatest alakú tárgyat egymás mellé helyeztem, és folyamatosan eltoltam őket egymáshoz képest, így adódtak a különbségek, amit mértem. A mérési pontok a találkozási élek voltak, itt mértem meg kézzel a két tárgy homlokfalának a távolságát. Mivel a mérés kézi részét mérőszalaggal végeztem, ezért ott számolni kell némi pontatlansággal. A lézer által mért értékeket a programom által készített wavefront objektum formátumba mentettem ki, és a kiértékelést a Meshlab nevű szoftverrel végeztem. A kapott értékek a kalibrációnak köszönhetően milliméter pontossággal adódtak. Ezzel már másfél méteres akadályérzékelési távolságot is jól lehet kezelni, mivel pontosan meghatározható az akadály távolsága a robottól. A lézer detektálása és a rekonstrukció futási ideje zajszűréssel együtt 16ms lett, ami azt jelenti, hogy másodpercenként 62.5-ször képes a rendszer a környezetét detektálni. A 21-es ábrán szemléltetjük a mérési módszert. A jobb oldalon a sárga vonal jelöli a mért távolságot.



Lézeres mérés



Kézi mérés



Lézer távolság meghatározása

21. ábra Mérés menetének szemléltetése

Eredményeiben a lézerszkennер modul kimagaslóan jól sikerült az eredeti elvártakhoz képest, így annak alkalmazása sok lehetőséget rejt magában, és hagy még időt az akadályok detektálására, valamint az elkerülési útvonal megtervezésére.

7. Akadálydetektáló modul

A robotunk számára az akadályok pontos detektálása elengedhetetlen, mivel csak azok ismeretében lehet képes kikerülésre. Fontos kérdés, hogy mit tekintünk akadálynak. Akadály lehet például egy szék lába, egy fal, egy oszlop, vagy egy mozgó ember lába, de semmiképp nem tekinthető akadálynak, egy fűcsomó, vagy szőnyeg, mivel azokon át tud menni a robot. Így meg kell tudnia különböztetni az akadályok fajtáját, nem elég csak megmondani, hogy az adott objektum akadály-e vagy sem. Emellett a detektált részekről meg kell határozni a pontos térbeli helyzetét. Ezen problémákkal foglalkozunk ebben a fejezetben.

7.1. Felülnézeti kép elkészítése

A robotnak két felhasználási környezete van, beltéri vagy kültéri. Beltéri felhasználás esetén feltételezhetjük, hogy a talaj sík, nincsenek emelkedők, csak minimális egyenletlenségek. Ezt felhasználhatjuk arra, hogy a 3.1.5-ös fejezetben megismert módon felülnézeti képet készítsünk a belső területekből, így térképet adva a robot kezébe. Ehhez szükségünk lesz a föld-kamera homográfiájának kiszámolására.

Homográfia mátrixnak nevezzük azt a háromszor hármass mátrixot, ami két kép közötti kapcsolatot ír le olyan módon, hogy a mindkét képen megjelenő jellegzetes pontok a transzformáció után egybe essenek. Ezen mátrix három alkotóelemből épül fel, egy kétszer kettes forgatási mátrixból, eltolási vektorból, és perspektív pontból (22. ábra). Az ábrán s a skálázási tényezőt, a t az eltolási vektort, a p a perspektív pontot, az a , b , paraméterek pedig a kép ferdítési paramétereit jelöli.

$$H = \begin{bmatrix} sa \cos(\theta) & -sb \sin(\theta) & t_x \\ sa \sin(\theta) & sb \cos(\theta) & t_y \\ p_0 & p_1 & 1 \end{bmatrix}$$

22. ábra Homográfia mátrix felépítése *Forrás: [19]*

Ennek a mátrixnak a kiszámítására [19] van szükségünk ahhoz, hogy a két kép közötti transzformációt megtudjuk. A számítás a következő lépésekből áll:

- Felírjuk az egyenletet az eredeti és a végső pontok között $\begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{bmatrix} * \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix}$
- Rendezzük az egyenletet $x'_2 * (H_{31}x_1 + H_{32}y_1 + H_{33}) = H_{11}x_1 + H_{12}y_1 + H_{13}$

$$y'_2 * (H_{31}x_1 + H_{32}y_1 + H_{33}) = H_{21}x_1 + H_{22}y_1 + H_{23}$$

- Mivel H -ra akarjuk megoldani, ezért át kell rendeznünk ennek megfelelően az előző pontban kapott egyenleteket így ezt kapjuk $a_x^T * h = 0$

$$a_y^T * h = 0$$

ahol az egyes értékek az alábbiak

$$h = (H_{11}, H_{12}, H_{13}, H_{21}, H_{22}, H_{23}, H_{31}, H_{32}, H_{33})^T$$

$$a_x = (-x_1, -y_1, -1, 0, 0, 0, x'_2 * x_1, x'_2 * y_1, x'_2)^T$$

$$a_y = (0, 0, 0, -x_1, -y_1, -1, y'_2 * x_1, y'_2 * y_1, y'_2)^T$$

Ebből megkapjuk a végső lineáris egyenletrendszert $A * h = 0$, ahol

$$A = \begin{pmatrix} a_{x_1}^T \\ a_{y_1}^T \\ \vdots \\ a_{x_N}^T \\ a_{y_N}^T \end{pmatrix}$$

Ezt a lineáris egyenletrendszert már ki tudjuk számítani az SVD [20] módszerrel (Singular Value Decomposition). Ebből a mátrixból úgy képezzük a forrás kép pixeleinek pozíciójából a cél kép pixeleinek pozícióját, hogy vesszük a p_0 , p_1 , és 1 összegét, ezt z -nek hívjuk a továbbiakban (a 22-es ábrán lévő értékekre hivatkozunk). Majd a mátrix elemeit megszorozzuk a pixel koordinátaival, az adott pixel z koordinátáját 1-nek véve, és ezt elosztjuk az előbbiekben kalkulált z értékkel. Így adódik az új pixel koordináta. A folyamat gyorsításához lookup table-t készítünk az egyes értékekhez, így a

számítást csak egyszer kell elvégezni, a továbbiakban, már csak tömbhivatkozással biztosítható egyes pixelek transzformációja.

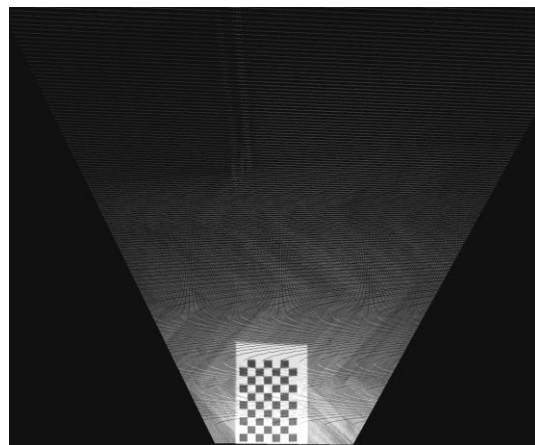
7.1.1. Felülnézeti kép kalibrálása

Számunkra a föld síkja és a kamera síkja közti homográfia mátrixot szükséges kiszámolni. Ehhez szükségünk van olyan pontokra, amelyeket egyértelműen tudunk, hogy hol helyezkedik el a képünkön, és hogy hol kéne elhelyezkednie a valóságban. Erre a feladatra is megfelel a sakktábla detektálás módszere, ahol a sakktábla négyzetei által meghatározott sarkokat detektáljuk. A sarkokról tudjuk, hogy vízszintes és függőleges irányban egységnyi távolságra helyezkednek el egymástól. Ezt kihasználva egy képzeletbeli sakktábla pontjait megadva, a kamera képre vetítve, és a detektált sakktábla képét véve, a két sakktábla homográfia mátrixa már könnyen kiszámolható az előző pontban leírt módszerrel, és elkészíthető a leképezés lookup table-je.

Az így kapott kép csak arra jó, hogy a robot előtti teret felülnézetben láthassuk, ám belátható, ha egy tárgy kilóg a föld síkjából, amire kalibráltuk a rendszert, akkor az tévesen jelenik meg. Ellenben a ténylegesen a föld síkjában lévő mintázatok, például parketta elemek már tényleg azonos méretűek lesznek. A 23-as ábrán a felülnézeti és az eredeti kép látható. A kamerához közeli részek részletesek, míg a kamerától távolabb eső részek kevesebb adatot tartalmaznak. Ebből következik, hogy a kamerától mért távolsággal arányos a számítás pontossága, azaz minél távolabb vagyunk a kamerától annál pontatlanabb az eredmény.



Eredeti kamerakép



Felülnézetivé transzformált kép

23. ábra Felülnézeti kép

A kapott felülnézeti képet még kalibrálni kell az ismert sakktábla paraméterekkel. A táblánkon lévő négyzetek 24,8*24,8mm-esek. Ezzel kalkulálva, a felülnézeti képen vesszük a kamerához legközelebbi vízszintes sort, ennek vesszük a hosszát, majd az egyik függőleges hosszát is lemérjük. A mérésekből az alábbi értékek adódtak, a vízszintes távolság 124mm a függőleges pedig 198,4mm. A kamera képen detektált sakktábla pontoknál a mértékek megegyezők között megmérjük a pixelben vett távolságokat, majd ezeket leosztjuk a valós távolsággal, az így kapott két értéket átlagoljuk, és ez fogja megadni, hogy egy pixel hány mm-nek felel meg a valóságban. A pontosság értékek a III-as táblázatban találhatóak. A távolságok nem a robottól való, hanem az egyes referencia pontok közötti távolságot jelentik. Az egyik referencia pont a kameraképen a kamerához legközelebb eső pont volt.

Mérés	1	2	3	4
Kézzel mért távolság	315mm	584mm	915mm	1560mm
Felülnézeti képen mért távolság	314,2mm	587,1mm	923mm	1575,6

III. táblázat Kalibrált képen mért távolságok pontossága

A mért értékekből látható, hogy a kamerához közel lévő pontok esetében az elért pontosság milliméter közeli. A távolabbi pontok esetében azonban ez romlik, másfél méteres távolságnál már a centiméteres pontosság sem érhető el. Azonban ez a centiméteres pontosság még mindig elegendő a robot szempontjából, mivel a robot elég nagy kiterjedésű, és nem képes mm pontos navigációra.

7.1.2. Felülnézeti kép használata

A kapott felülnézeti képünket több-mindenre használhatjuk. Egyrészt alkalmas a talaj síkjában lévő pontok távolságának a mérésére. Térképet lehet vele készíteni, amennyiben tudjuk, hogy az adott képen mely részek tekinthetők akadálnak. Amennyiben tudjuk az akadályok felülnézeti képre vetített helyzetét, akkor pontos útvonaltervezésre is lehetőséget ad a módszer.

7.2. Akadálydetektálás

Esetünkben az akadályok pontos detektálása elengedhetetlen, annak érdekében, hogy a robot ki tudja kerülni azokat, ne ütközzön beléjük. Erre a feladatra mindenképp egy stabil rendszert célszerű létrehozni, ami a lehető legnagyobb mintavételezésre képes adott időegység alatt, valós időben.

7.2.1. Akadálydetektálás lézer segítségével

Mivel a robot rendelkezik már egy nagy pontosságú lézerszkennerral, ezért ezt használjuk fel az akadályok megtalálására és azonosítására.

7.2.1.1. Akadálydetektálás a lézer talajra vetített helyzete alapján felülnézeti két segítségével

Az ebben a pontban ismertetett módszernél arra a feltételezésre építettünk, hogy a lézer síkra vetített helyzete állandó marad, csak akkor változik, ha akadályt érzékel. Ennek a nézőpontnak, az az előnye, hogy ha nem látszik jól a lézer, akkor az adott térrészt mindenképp akadálnak detektálja. Így elmondható, hogy azokon a részeken, ahol nem sikerült a lézert detektálni, a modul nem enged arra továbbhaladni. Ez hátrány is lehet akkor, ha olyan felületre kerül a robot, ahol elég kicsi a lézer visszaverődése ahhoz, hogy azt detektálja, mivel ekkor mindent akadálnak azonosít a rendszer, miközben a valóságban nincs előtte semmi. Felmerül a kérdés, hogy a robot szempontjából melyik a jobb megközelítés. A robot detektáljon mindent akadálnak, ha nem képes felismerni az adott részeket, azaz a lézer intenzitása a képen túl alacsony, vagy legyen az a kiinduló alap, hogy semmi nem akadály, csak az annak detektált részek. Ebben a modellben az első megközelítés jobbnak tekinthető, mert a robot biztonságosabb mozgását eredményezi, azonban sok hiba fakadhat belőle. A különböző felületeken végzett kísérleteinkre példát a 24-es ábra prezentál. Itt látható, hogy csak a sötétebb belterületeken látszik jól a lézer, kültéren gyenge a fényereje, és a talajnak nincs olyan visszaverő képessége.



Aszfalt



Laminált padló



Csempe

24. ábra Lézer képe különböző talajokon

Ha sikerült meghatározni, hogy melyik részek tekinthetők akadállyal, akkor azok térbeli pozíciójának megbecsléséhez használhatjuk a kalibrált felülnézeti képet. Mivel csak a földsíkon lévő lézerpontok adnak valós adatot a távolságról, ezért az akadályok távolságát csak közvetlenül az első megjelenésük pillanatában tudjuk detektálni, ekkor is csak pár centiméteres pontossággal. A módszernek ez gyenge pontja. Mivel csak a detektálás pillanatában tudunk következtetni a tárgyak távolságára, így az elkerülési útvonalat ekkor kell megterveznünk. A másik probléma, hogy a tárgyak kiterjedése is csak a felbukkanás pillanatában számolható, a továbbiakban nem, mivel a tárgy alakja változik, emiatt a tárgyak követésére még a nagyságukat sem tudjuk felhasználni. A felülnézeti képet tehát csak az útvonal durva megtervezéséhez tudjuk használni.

Ezt a módszert az előbbieken említett problémák miatt elvettem, és inkább a lézer pontok térbeli pozíciója alapján kezdtem el detektálni az akadályokat, mivel ott folyamatosan pontos távolság adatunk van.

7.2.1.2. Akadálydetektálás a lézer térbeli pozíciója alapján (első verzió)

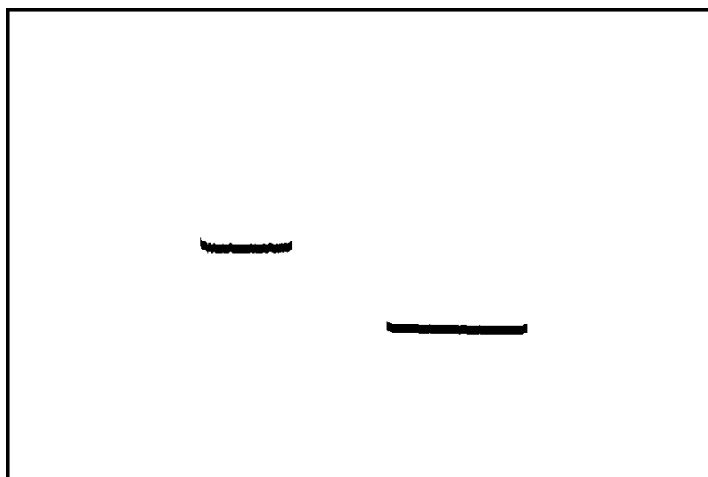
A lézerszkennerek modulnak köszönhetően a rekonstrukció után milliméter pontos adatunk van a megtalált akadályok távolságáról és helyzetéről. Ez a modul ráadásul valós időben végzi számításait. A térbeli pontok esetén azonban felmerül több probléma is. Ezek az alábbiak: mi alapján határozzuk meg, hogy mi akadály és mi nem; mit vegyünk összetartozó részeknek; milyen távolságot vegyünk a nagyobb akadályok tekintetében távolságnak.

Az akadályok meghatározásánál azt vettem alapul, hogy az akadályok a kamera képen általában elkülönülnek, azaz nem érnek össze. Így az akadályok szegmentálása már egyszerű. Tekintem a megtalált lézerpontokat, végigmegyek rajtuk, és az egybefüggő részeket keresem. Ezeket akkor jelölöm akadállyal, ha azok elérnek egy minimális méretet. A módszernek az az előnye, hogy ha nem épületben, hanem kint, füves talajon használjuk, akkor a fűszálakat nem tekintjük akadállyal, így rendesen tud tájékozódni ott is. Hátránya, hogy a vékonyabb akadályokat nem biztos, hogy kellő időben detektálja. Belátható, hogy mindenképp detektál, mivel egy adott tárgy a kamerához közelebb nagyobbak látszik, mint távolabb, azaz a minimális szélességet a kamera képen szinte minden esetben el fogja érni. Erre példát láthatunk a 25-ös ábrán. A képen két azonos méretű doboz található a kamerától különböző távolságokban. Jól kivehető, hogy a jobb oldalon lévő doboz közelebb van, és ezen a lézer is nagyobb területen jelenik meg, a bal oldali távolabbi dobozon a lézer már csak kétharmad akkora vízszintes kiterjedésű.



25. ábra Lézer képe azonos szélességű tárgyon különböző távolságokban

A detektált akadályoknál ezután meghatározásra került a robottól mért távolságuk olyan módon, hogy az észlelt akadály két szélén elhelyezkedő lézerpont távolságának az átlagát képeztük. Az útvonaltervezéshez azonban még nem kielégítő a módszer, mivel az egyes akadályok pixel szélessége lett az akadály szélességének meghatározva, így ha azok közelebb kerülnek a kamerához nagyobbak látszanak, így nem tudunk elkerülési útvonalat tervezni. Előny viszont, hogy a módszer igen gyors, és minden esetben pontos információval rendelkezünk a mélységi távolságok tekintetében. Ebből következik, hogy az akadályok követése már lehetséges. A 25-ös ábrához tartozó távolságkép látható a 26-on.



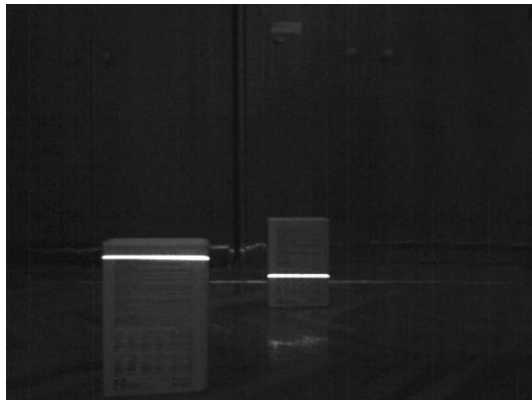
26. ábra Távolságkép

A képen a kamerától mért távolság lentől felfelé értendő, azaz a kép alja esik egybe a kamera síkjával, itt található a kamera. Minden pixel két millimétert jelent, a kép eredeti felbontásában. Az ábrán két akadály látható, az egymástól mért távolságuk nem mérvadó, és a szélességük sem, mivel bár a két objektum ugyanaz, de a perspektív leképzésű kamera miatt méretük mégis különbözőnek látszik.

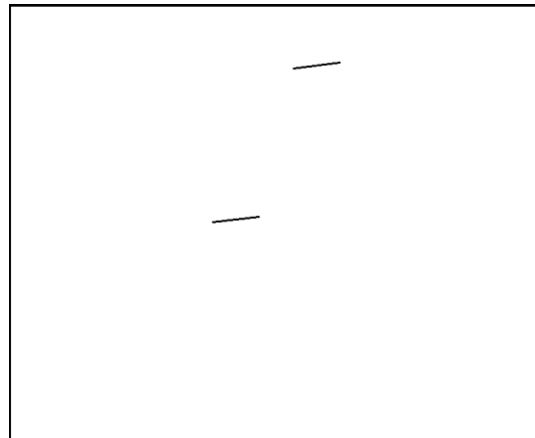
7.2.1.3. Akadálydetektálás a lézer térbeli pozíciója alapján (második verzió)

Az előző megoldás legnagyobb hibája, hogy nem ad információt a tárgyak tényleges elhelyezkedéséről, csak a távolságukról. Szélességük változik a közelségükkel fordítottan arányosan, így folyamatos útvonaltervezésre nincs lehetőség, valamint bizonytalanabb a tárgyak követése az elkerülés ideje alatt.

Erre megoldást ad, ha a rekonstruált lézerpontoknak nem csak a mélységi információját, hanem a szélességi koordinátáit is figyelembe vesszük (27. ábra).



Kameraképen az akadályok



Detektált akadályok a térben

27. ábra Azonos méretű akadályok megjelenése a kamera és akadály képen

Az akadály távolságtól függetlenül valós és azonos méretekké jelent meg a jobboldali képen. A modul jelenlegi hátránya az, hogy a megtalált akadályokat síkokkal határoltnak tételezi fel, és ennek megfelelően számolja a távolságokat, azaz a tárgy egy oldalon elhelyezkedő pontjainak térbeli koordinátájából kapott átlagát veszi. Ez nem síkfelületek esetén téves eredményt adhat. Például homorú tárgy esetén közelebbinek mondja, míg egy domború tárgy esetén távolabbinak, mint a valóság.

Ezt a modult előzetesen kalibrálni kell a szélesség irányú értékek helyes meghatározásához, azaz fel kell vele ismertetni egy tárgyat különböző távolságokban, és meg kell adni, hogy mekkora a tárgy fizikai mérete. A méréssorozatok által kapott szélességek átlagát vesszük, elosztjuk a valós fizikai szélességgel, és megkapjuk a lézer síkjában vett szélesség és a valós szélesség közötti kapcsolatot. Így a lézer síkjában lévő egy egységnyi távolság milliméteres szélességét határozzuk meg.

A kalibráció elvégzése után rendszerünk valamennyi irányban képes pontosan meghatározni az akadályok térbeli helyzetét nagy sebességgel, mivel az akadálydetektáló algoritmus 1ms-es számítási igényű az alkalmazott konfiguráción. A valós idejű akadálydetektáló modul így több mint hatvan mintavételezésre képes másodpercenként, a robot gyors mozgások esetén is biztonságosan irányítható marad.

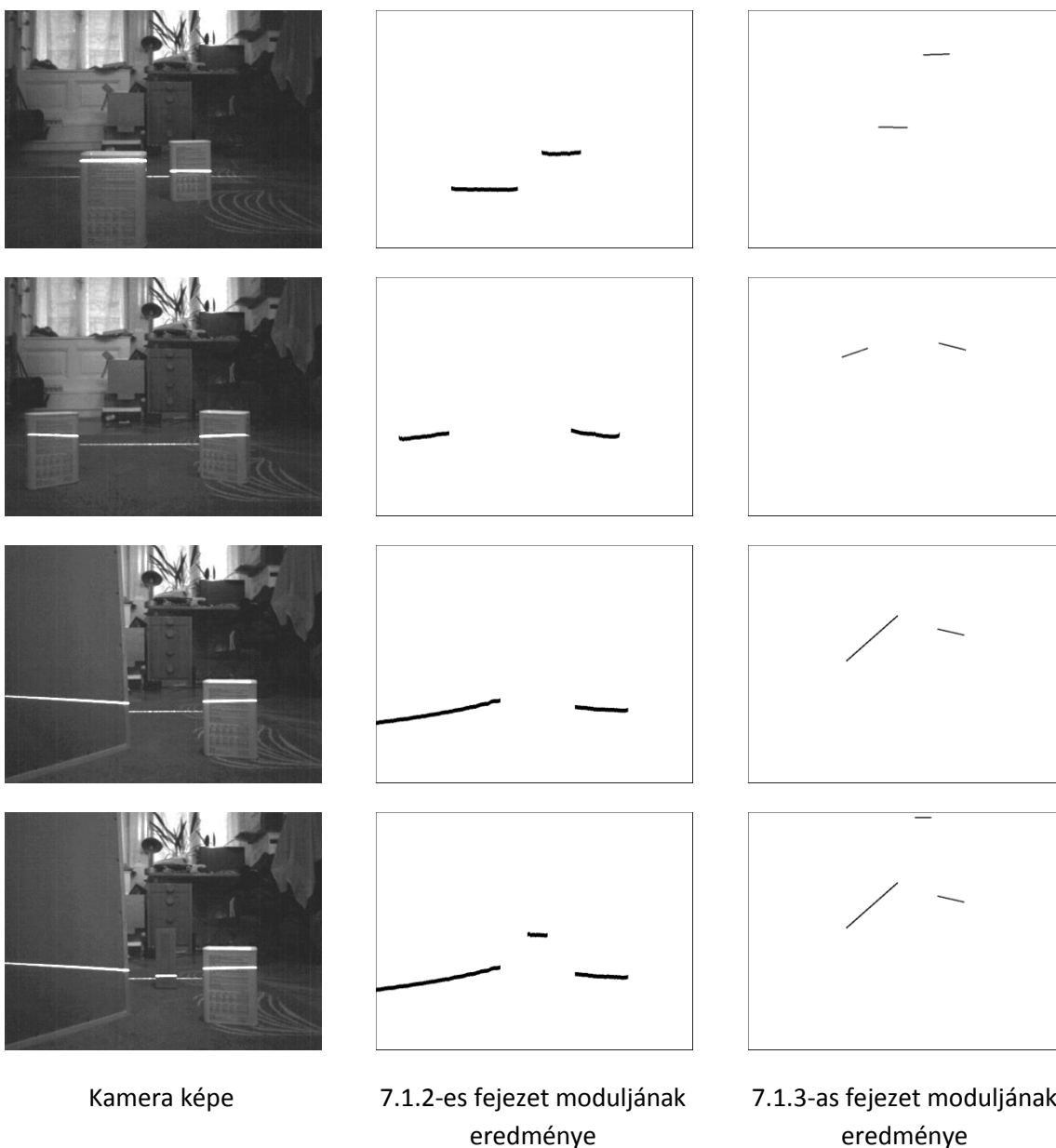
7.2.1.4. Összehasonlítás

A három akadálydetektáló modul közül az 7.1.1. fejezetben szereplőnek nagy hátránya volt, hogy az akadályokat csak az első megjelenés pillanatában képes távolság információval felruházni. Emiatt

az akadályok nyomon követése és a további akadályok elkerülése nehezkessé válhat. Emiatt nem alkalmazzuk a módszert a akadályelkerülés során, későbbi térképkészítéshez viszont hasznos lesz.

A 7.1.2. fejezetben taglalt akadálydetektáló az objektumok távolságát folyamatosan meg tudja határozni, és ezen távolságok alapján már lehetséges az akadályok követése. Hátrány azonban, hogy a tárgyak szélességéről nem rendelkezik valós információval, így az elkerülési útvonal nehezen tervezhetővé válik.

A legutolsó fejlesztésnél, ami a 7.1.3. fejezetben jelenik meg, sikerült a megelőző verziót úgy javítani kalibrációval, hogy a fő hiányosságot kiküszöböljük. Itt valós szélesség információt kapunk a tárgyak elhelyezkedésétől függetlenül. Ezzel a modullal már kiváltható a felülnézeti kép készítése, mivel meg tudjuk határozni az akadályok robotunkhoz viszonyított fizikai elhelyezkedését. A 28-as ábrán követhető a lézerszenzorra kifejlesztett két utóbbi módszer eredményének összehasonlítása.



28. ábra Összehasonlító táblázat

7.2.1.5. Tesztelés

Az akadálydetektáló modult nyílt és beltéri terepen is vizsgáltam. A vizsgálat során a rendszer csak az általam elé tett akadályokat láthatta, így minden esetben el lehetett dönteni, hogy megtalálta-e az objektumot, vagy sem, esetleg valamit tévesen akadálynak detektált. A mérési környezet a 29-es ábrán látható.



29. ábra A tesztelés körülményei

A bal oldali képen a robot kültéri viszonyok között egy fallal szemben helyezkedett el. Jobb oldalon két akadály, és egy fénylő alumínium szekrényláb jelent meg a robot mozgása során. A mérési eredmények a környezetben vett különböző állásokból adódtak eltérő számú akadállyal. A IV-es táblázatban kerültek összefoglalásra eredmények.

Épületen belül	Helyes detektálások száma (volt akadály és detektált)	Hibás detektálások száma (nem volt akadály, és detektált)
Összes detektálható akadály száma 112	108	21
Hibasázalék	3,57%	17,2%
Hirtelen ütközési távolságon belülre juttatott akadályok száma 52	52	-
Hibasázalék	0%	-
Épületen kívül	Helyes detektálások száma (volt akadály és detektált)	Hibás detektálások száma (nem volt akadály, és detektált)
Összes detektálható akadály száma 115	107	45
Hibasázalék	4,4%	28,1%
Hirtelen ütközési távolságon belülre juttatott akadályok száma 47	46	-
Hibasázalék	2,1%	-

IV. táblázat A mért eredmények összefoglaló táblázata

Az eredményekből kitűnik, hogy a rendszer az akadályokat több mint 95%-os pontossággal detektálta, de sokszor, az esetek több mint 25%-ában detektál fals akadályt. Ez nem okoz gondot a

robot mozgása során, mivel a hibás detektáció nem egymás utáni képsorozaton jelenik meg, és a másodpercenkénti 60 mintavételezés gyors javítási lehetőséget biztosít, amivel a hibás detektálások száma csökkenhet. A táblázatból látszik, hogy a robot kint és bent egyaránt 95%-ot meghaladó pontossággal detektálja a valós akadályokat, de kültérben több mint 10%-kal nagyobb a hibásan akadálnak detektált elemek száma. A hirtelen, ütközési távolságon belül elé kerülő akadályokat pedig egy eset kivételével mindig jól detektálta.

7.2.1.6. Összegzés

Végeredményben elmondható, hogy az akadályok detektálásánál átlagban közel 95%-os pontosságot sikerült elérni, a megtalált akadályok elhelyezkedése megfelelt a valóságnak. A legutolsó fejlesztés sikeresen meg tudta adni az akadályok térbeli távolságát és szélességét, így az elkerülési útvonal már tervezhetővé válik.

8. Robotvezérlés interfésze

A rendelkezésünkre álló robotot soros porton keresztül lehet vezérelni. Minden parancs után választ küld a saját állapotáról, így azt folyamatosan figyelemmel kísérhetjük. A kapcsolat 38400 bit/sec-es átviteli sebességgel rendelkezik, ahol az adatméret 8 bit, dupla záró karakter van és nincs paritás bit.

Egy parancs hat alapvető részből épül fel. Az egyes elemeket ASCII karakterpárokkal lehet megadni, az értelmező ezeket bájtokká alakítja. A parancs mindig x karakterrel kezdődik.

A parancsok az V. táblázatban találhatók.

Üzenet típusa	Cím1	Cím2	Cím3	Üzenet-számláló	Üzenet hossza	Adat1	Adat2	Adat3	Adat4
Első híd ki-bekapcsolás	00	02	07	0-65535 két bájton	01	00: ki 01: be			
Hátsó híd ki-bekapcsolás	00	02	08	0-65535 két bájton	01	00: ki 01: be			
Kormányzás	00	02	04	0-65535 két bájton	02	Irány felső byte-ja	Irány alsó byte-ja		
Sebesség	00	02	05	0-65535 két bájton	02	Sebesség felső byte-ja	Sebesség alsó byte-ja		
Megtett távolság	00	02	30	0-65535 két bájton	00	Első tengely felső byte-ja	Első tengely alsó byte-ja	Hátsó tengely felső byte-ja	Hátsó tengely alsó byte-ja

V. táblázat Robot vezérlési parancsainak felépítése

9. Útvonaltervező modul

Az akadályok detektálása után még egy fontos dolgunk maradt, ez pedig az akadályok elkerülésének megtervezése, azaz útvonaltervezés és a robot útvonalon tartása. A fejezetben a vizsgált technikákat vesszük számba.

9.1. Bogár algoritmus

A bogár algoritmus [21] a bogarak akadály elkerülését modellezi. A bogarak, ha találkoznak egy nagy akadállyal, akkor fogják és körbejárják azt, majd ha teljesen körbejárták, akkor megkeresik az akadály körül azt a pontot, ami a célhoz a legközelebb van. Ha több akadály van az útjukban, akkor akadályról akadályra járva mennek végig ugyanezt a logikát követve. Ezt a folyamatot valósítja meg a bogár algoritmus is.

A megközelítés hátránya, hogy a robotnak ezen algoritmus alapján többször körbe kell járnia egy adott testet, míg tovább halad onnan. A módszer javítását tartalmazza a tovább fejlesztett bogár algoritmus.

9.2. Javított bogár algoritmus

Ez a módszer annyiban tér el az előzőtől, hogy az első adandó alkalommal a cél felé folytatja az utat, nem kerüli meg többször az akadályt.

A bogár algoritmusok előnye, hogy igen egyszerűek, így könnyen implementálhatók. Hátrányuk, hogy kicsi robotokhoz, illetve konfigurációs térben alkalmazhatóak, mivel a robotok kiterjedésével nem számolnak. Mivel nem mehetünk a szabad úton, hanem az akadály körüli pályán kell mozognunk, ezért rendszerünkben más megközelítést alkalmazunk.

9.3. Akadályelkerülés a legnagyobb térköz irányába

Az első útvonaltervező modul, azon az elgondoláson alapult, hogy a robotnak minél biztonságosabb utat kell találnia hogy, semmiképp se ütközzön. Az alapötlet szerint megkeressük az akadályok közötti legnagyobb átjárót, és ha van ilyen, akkor arra haladunk tovább. Az algoritmus a 7.2.1.2-es fejezetben lévő akadálydetektálót alkalmazta. Először megnézte, hogy hány darab akadály van az útjában, ha nem volt semmi, akkor egyenesen ment, ha talált akadályt, akkor vizsgálta az akadályok számát. Amennyiben egyet talált, akkor a szabad nagyobb hely irányába indult el. Ha több akadállyal találkozott, akkor a legnagyobb térköz irányába fordult.

9.4. Legközelebbi akadály kikerülése

Az előző módszer hibáit figyelembe véve, a javított bogár algoritmust fejleszti tovább ez a módszer. Az akadályok követése megoldható volt az akadályoktól mért távolságok alapján. Mivel a robot eléggé kis sebességgel halad, ezért két detektálás között kicsi elmozdulás történik a képen. A követés módszere a következő. A robot detektálja az akadályt, eltárolja minden akadálnak a távolságát, és az akadálykép koordinátában vett vízszintes tengelyen lévő középpontját. Az eltárolt értékeket összehasonlítjuk a frissen detektált akadályokkal, ha az egyes akadálykoordináták nem térnek el egymástól túlságosan, akkor a két akadályt ugyanannak tekintjük. Ennél a módszernél lett implementálva az a funkció, ami biztosította, hogy a robot, ha egy akadályhoz túl közel kerül, akkor ne csak megálljon, hanem törekedjen annak kikerülésére. Amennyiben a robot hirtelen akadályhoz ért, akkor előbb megállt öt másodpercre, mivel feltételeztük, hogy az akadály hirtelen került oda, és

egy mozgó objektumhoz tartozik. Ha öt másodperc múlva még mindig ott volt az akadály, akkor hátrátolatott az akadálytól egy méter távolsáig, és az addigi iránytól eltérően indult tovább.

A módszer beltéri tesztelése pozitív eredményt mutatott. Hátránya, hogy sok kicsi, közel lévő akadály esetében, nem lehet eldönteni a továbbhaladás irányát.

9.5. Az előbbi két modell ötvözése

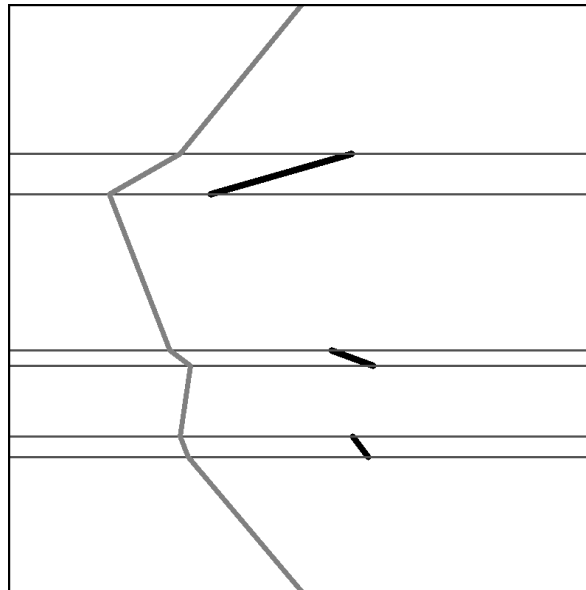
Ahhoz, hogy a két módszert össze lehessen gyúrni, szükséges volt a 7.2.1.3-as fejezetben kifejlesztett akadálydetektáló javításra, azaz, hogy az egyes akadályok a távolságtól függetlenül azonos mérettel látszódnak, és azok tényleges szélességét le lehessen kérdezni kalibráció után. Ezt felhasználva dolgoztam ki a harmadik akadályelkerülő modult. Az akadályok közötti távolság ekkor már ismert volt, ezt kihasználva úgy lett megtervezve az útvonal, hogy a legközelebbi akadályt a legnagyobb szabad hely felé kerülje ki, de csak akkor menjen erre, ha ez a térköz elég nagy ahhoz, hogy a robot átférjen rajta.

A megközelítés csak a hozzá legközelebb eső akadállyal számolt, hiszen ezek az útvonaltervező algoritmusok lényegében a bogár mintára épültek. Ennek megfelelően több akadály esetén nem az összes akadályra határoznak meg elkerülési útvonalat, csak a hozzá legközelebb esőt veszik figyelembe. A területek bejárása és az akadályok kikerülése így nem hatékony.

9.6. Exact cell decomposition algoritmus

A [22] módszer lényege, hogy a munkateret az akadályok segítségével több részre bontják fel, és ezekben a térrészekben keresi az algoritmus az optimális utat. A módszerben célszerű az egyes cellák határainak középpontját venni az optimális útvonal részének. Az egymás után következő cellaközéppontok, és ezek távolsága egy súlyozott gráfot adnak eredményül, amiben már könnyen megkereshetjük a célt. Az útvonal biztonságossága növelhető úgy, hogy a gráf egyes csúcspontjaihoz is rendelünk súlyokat, méghozzá olyan módon, hogy minél szélesebb az akadálymentes rész, annál nagyobb értéket kap a csúcspont. A gráf két külön súlyrendszert tartalmaz, melyek látszólag ellentmondanak egymásnak, azonban az egyes élek a távolságot, a csúcsok pedig a biztonságot jelentik. Megfelelő súlyozással a legrövidebb és legbiztonságosabb utat tudja megtervezni a robot. A 30-as ábrán szemléltetjük az implementált algoritmus eredményét. A vízszintes vonalak jelentik az egyes cellák határait, a fekete vastag vonal mutatja a detektált akadályt, a vastag szürke vonal pedig a tervezett útvonalat. Az ábrán a robot lentről felfelé halad végig a tervezett útvonalon.

A megtalált útvonal elég távol elkerüli az akadályokat. Ezt bővíteni lehet a szélességek figyelembevételével. Az algoritmus bejárható útvonalat ad, kicsi a számítási igénye, így alkalmazása kézenfekvő. Nagy előnye még, hogy minden detektált akadállyal számol és térképet ad a robot kezébe. A módszer nagy előnye, hogy az akadályok detektálása pontos, és követésükre képes a robot, így az összes akadály mért távolsága haladás közben ismert. Ha közeli akadályt detektál a robot, akkor visszatolat, és innen folytatja útját az ideálisnak gondolt pályán.



30. ábra Exact cell decomposition algoritmus eredménye

9.7. Értékelés

A fenti módszerekből kitűnik, hogy a bogár algoritmusok és a rájuk épülő megoldások jók lehetnek kis kiterjedésű robot esetén, azonban az általam használt robotnál nem váltak be igazán, mivel ezek leginkább pontszerű mérettel kalkulálnak. A 9.6-os fejezetben használt módszer azonban minden kitételnek megfelel, és szem előtt tartja a robot méreteit. Az egyetlen hátránya, hogy bizonyos esetekben a megtervezett útvonal elég éles kanyarodási szögekkel rendelkezik, amit nem biztos, hogy a robot biztonságosan be tud venni, de erre megoldást találunk a következő fejezetben.

10. Robot útvonalon tartása, irányítása

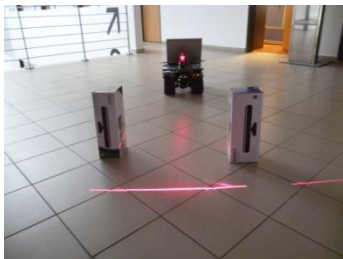
A robot irányítási modulja felelős azért, hogy a megtervezett útvonalon haladjon végig, figyelemmel kövesse a változásokat, és az akadályokat elkerülje. A modul feladata, hogy a térkép alapján felmérje, milyen irányban, mekkora sebességgel induljon el. A sebesség tíz fokozatban szabályozható irányonként.

A kerekek tengelyenként képesek külön-külön irányba állni, amit kihasználunk arra, hogy minél nagyobb kanyarodási szöget tudjunk elérni a robottal. Szükség esetén a kerekek ellentétes állásúak, és az így elérhető legnagyobb fordulási szög negyvenöt fok. A vezérlő parancsok kiadásánál (lásd 8. fejezet) a középállást, azaz az egyenesen való haladást a 900-as értékkel lehet elérni. Maximális jobbra fordulást a 600-as, balra fordulást az 1200-as érték képviseli. Ezen paraméterek között arányosan került felosztásra a két intervallum. A robot saját, és a rajta lévő laptop súlya alatt túl nagy terhelés jut a szervó motorokra, így azok nem tudnak azonnal a parancs kiadása után irányba állni, szükségük van pár centiméter megtételére a teljes beállásig. Másik tapasztalatunk az, hogy nem mindig egyezik meg a kerekek által bezárt szög, néha kicsit eltér, és a kormány fejek is lötyögnek, azaz nem precíz a mechanika. Ebből fakad, hogy a kanyarodásnál nagy hibával kell számolnunk a becsült fordulási szög esetében, és ezt is figyelembe kell vennünk a fordulások tervezésekor.

A robot útvonalon tartását a következő algoritmus valósítja meg. Megtervezzük az akadályok ismeretében az útvonalat, majd megnézzük, hogy az útvonal első szakasza milyen szöget zár be az egyenes haladási iránnyal. Ennek megfelelően állítjuk be a kormányt. A rendszer valós idejű, így az egyes akadályok kikerülési útvonala folyamatosan feltérképezésre kerül, és a változásokat pontosan tudja követni a robot. Ha a robot 30cm-nél közelebbi akadályt érzékel, akkor vár egy másodpercet, majd ha az akadály még mindig előtte van, akkor hátramenetbe kapcsol, és az akadálytól egy méteres távolságig eltávolodik. Innen ismét megkísérel annak kikerülését. Ha ez háromszor történik egymás után, akkor másfél méterre megy hátra, és az eredetileg tervezett útvonaltól eltérő irányba indul tovább. Ezzel biztosítjuk, hogy a robot ne kerülhessen végtelen ciklusba egy akadálynál.

10.1. Tesztelés

Alapos tesztelésnek vetettük alá ezt a modult is, mivel a helyes működés kulcsfontosságú a robotrendszerben. A modult vizsgáltuk beltérben, ahol a robotot letettük egy véletlen helyre, elindítottuk és néztük, hogyan reagál az egyes akadályokra. Az akadályok típusai eltérőek voltak (31. ábra). Egyedüli nehézséget a kanyarodás pontatlansága jelentette, mert párszor a sarkánál hozzáért az akadályhoz a robot. Egy másik tesztsorozat során hirtelen akadály került a robot elé. Erre a robot minden esetben jól reagált, azonnal megállt, és ha az akadály nem távozott el onnan, akkor tolatott, majd megpróbálta kikerülni azt. A harmadik esetben célzottan akadályokat helyeztünk el egy nagyobb sík területen. A robot sikeresen el tudta kerülni azokat, ha néhány esetben fals akadályt jelzett, akkor a gyors feldolgozásnak köszönhetően valós időben korrigálni tudta a hibát, mivel a következő képeken már helyes volt a detektáció.



31. ábra A robot tesztelés közben

Kültéri, füves terepen is tesztelésre került a robot (a 7.2.1.5-ös fejezet IV-es táblázatában találhatóak meg a pontos eredmények). Hibát csak az okozott, ha a megvilágítás miatt nehéz volt megtalálni a lézert, ekkor a robot fals akadályt érzékelt általában és megállt.

10.2. Modul értékelése

A robot ezzel az útvonalon tartási módszerrel stabilan tud közlekedni. Sikertelenül elérni, hogy az akadályokat közel 95%-os biztonsággal detektálja, és az esetleges ütközéseket mindig elkerülje, ha túl közel kerül egy tárgyhöz, akkor is. A robot mind kint, mind bent tud navigálni, bár a kinti környezetben a többletfény miatt több mint 10%-al többször detektál fals akadályt. A tényleges akadályokat 95%-os pontossággal detektálta, és sikeresen ki is kerülte. A hirtelen ütközéseket a IV-es táblázatban lévő értékek alapján egy kivétellel mindig sikeresen elkerülte.

11. Értékelés

Az egyes modulok külön-külön is tesztelésre kerültek a 6.1.1, 7.2.1.5 és a 10.1-es fejezetekben. Ezen tesztek alapján elmondható, hogy a lézerszkennerral sikerült milliméteres pontosságot elérni másfél méteres távolságon belül. Az erre épülő akadálydetektáló modul épületen belül 95% fölötti detektálási arányt ért el, de kültéren a robot már elég sokszor kényszerült megállásra a hibásan detektált akadályok miatt. Egyes esetekben a robot a kinti környezetben beragadt, mivel a kép olyan világos volt, hogy nem látszódott a lézervonal, és ütközési távolságon belül észlelt fals akadályt. Amikor több különböző akadályt helyeztünk el egy nagy területen, és véletlen helyen indítottuk el a robotot, több mint három percig ütközés nélkül sikeresen navigált az akadályok között.

12. Összegzés

A dolgozat célja egy olyan irányítási rendszer fejlesztése volt, ami lehetővé teszi egy robot autonóm navigációját, biztosítva ezt gyors és pontos akadálydetektálással és útvonaltervezéssel. A kitűzött célokat sikerült elérni. A saját fejlesztésű lézerszkennert, a kifejlesztett akadálydetektációs, útvonaltervezési és navigációs algoritmusok beltéri körülmények között autonóm robotműködést eredményeznek. A lézer szkennerral sikerült elérni a milliméteres pontosságot a szubpixeles lézerkeresésnek köszönhetően. A kinti teszteredményekre alapozva továbbfejlesztésre került a lézerdetektáló algoritmus. A hamisan detektált lézerpontok szűrésére szóráson alapuló algoritmust fejlesztettem. Az akadálydetektáló modul a lézerszkennernak köszönhetően valós idejű feldolgozást ért el a 16ms-os futási idejével. Elmondható, hogy az akadályok valós térbeli pozíciója és szélessége is meghatározásra került, és az egyes akadályokat 95%-os pontossággal sikerült azonosítani kültéren és beltéren is. Az útvonal tervező algoritmus segítségével a robot az akadályokat elkerülte, csak néhány esetben érte a robot sarka azokat. Az egész rendszer futási ideje 32ms lett, ami másodpercenkénti 33 mintavételezést tesz lehetővé. A robotot tesztelni tudtam különböző fényviszonyok és kinti, valamint benti körülmények között is. Kinti esetben a fényviszonyok miatt nagyobb teljesítményű lézer szolgáltatna stabilabb navigációt, mivel több esetben érzékelt fals akadályt a rendszer, azonban az ütközések egy kivételével, minden esetben elkerülhetők voltak.

Összességében elmondható, hogy sikerült egy működő rendszert megtervezni és elkészíteni, aminek alkalmazása több területen is elképzelhető. Lehetőség van az akadálydetektáló modul külön alkalmazására autonóm autók, vagy robotok fejlesztésénél, esetleg a járműiparban a ráfutásos balesetek megelőzésére, parkolás segítésére, mivel az autóktól valós távolság adatokat kaphatunk. A lézerszkennernak még több alkalmazási területe is lehet, mivel milliméter pontos mérést tesz lehetővé.

13. Továbbfejlesztési lehetőségek

A rendszer leggyöngébb része a lézerkereső algoritmus, mivel a túl világos képeken kevésbé hatékony a lézer megtalálása. Ennek ellenére mindenképp a lézeres módszernél maradok, mivel ezzel lehet a legnagyobb mérési pontosságot elérni. Egyik fejlesztési lehetőség a lézer teljesítményének növelése. Egy másik fejlesztési irány két kamera képe alapján való lézerdetektálás abból az alapelvből kiindulva, hogy a lézer polarizált fény, lehetőségünk van polárszűrő alkalmazására, és ezzel

elérhetjük, hogy az egyik képen látható legyen a lézer fény, míg a másikon nem. A két kamera képének homográfiáját kalibrálva, az egyik kamera által készített kép egyszerűen leképezhető a másikéra. A két kép különbségét véve és utána a megmaradó részeken vörös színre való szűrés által elvileg tökéletes eredmény kapható valós idejű futás mellett.

A robot saját pozíciójának (odometria adatainak) meghatározását szeretném még megoldani, hogy a folyamatos és nagyobb környezetet tároló térképezési funkció is implementálására kerüljön. Így belső területek feltérképezésére lehetne használni a rendszert, akár centiméteresnél is jobb pontossággal.

14. Felhasznált irodalom

- [1] M. Sarcinelli-Filho, H. A. Schneebeli, E. M. O. Caldeira, C. M. Soria „Optical Flow-based Obstacle Detection And Avoidance in Mobile Robot Navigation”. Mediterranean Conference on Control and Automation – MED 2002, Lisbon, Portugal, July 9-12, 2002. (CD issue) p. 7. URL: med.ee.nd.edu/MED10/pdf/340.pdf, utolsó látogatás: 2012. október 30.
- [2] Iwan Ulrich, Illah Nourbakhsh „Appearance-Based Obstacle Detection with Monocular Color Vision” Proceedings of the AAAI National Conference of Artificial Intelligence, Austin, TX, July/August 2000, pp. 866–871.
- [3] Jens Klappstein, Tobi Vaudrey, Clemens Rabe, Andreas Wedel, and Reinhard Klette „Moving Object Segmentation Using Optical Flow and Depth Information” Lecture Notes in Computer Science, 2009, Volume 5414, pp. 611-623.
- [4] Luc Duvieubourg, Sébastien Ambellouis, Sébastien Lefebvre and Francois Cabestaing „Obstacle detection using a single camera stereo sensor” Third International IEEE Conference on Signal-Image Technologies and Internet-Based System, SITIS’07 Shanghai: China, 16-18 Dec. 2007, pp. 979 – 986.
- [5] Tilman Wekel, Olaf Kroll-Peters, Sahin Albayrak „Vision Based Obstacle Detection for Wheeled Robots” International Conference on Control, Automation and Systems, Berlin, 14-17 Oct. 2008, pp. 1587 - 1592
- [6] Seydou Soumare, Akihisa Ohya and Shinichi Yuta „Real-Time Obstacle Avoidance by an Autonomous Mobile Robot using an Active Vision Sensor and a Vertically Emitted Laser Slit” University of Tsukuba, Intelligent Autonomous Systems, 2002.
- [7] Jimmy Tang, Avihed Zakhori „3D object detection from roadside data using laser scanner”, Proceedings of SPIE, Three-Dimensional Imaging, Interaction, and Measurement, 2011, p. 16. doi:10.1117/12.872620
- [8] Marc Pollefeys, Maarten Vergauwen and Luc Van Gool „Automatic 3D Modelling From Image Sequences” Proc. of. the XIX ISPRS Congress, Vol. XXXIII, Amsterdam, 2000, pp. 619–626.
- [9] Bónus Zoltán, Kelemen Bálint „MoBRo Háromdimenziós környezetleképző robot” Óbudai Egyetem, NIK TDK, 2011,
- [10] Richard Hartley, Andrew Zisserman „Multiple View Geometry in Computer Vision” Second Edition, Cambridge University Press, 2004
- [11] Frédéric Cazals and Joachim Giesen, “Delaunay Triangulation Based Surface Reconstruction: Ideas and Algorithms”, Téma SYM – Systèmes symboliques Project Geometria, Rapport de recherche n 5393 – November 2004- 42 pages
- [12] Franz Aurenhammer, “Voronoi Diagrams – A Survey of a Fundamental Geometric Data Structure”, Institute für Informationsverarbeitung Technische Universität Graz, 1991
- [13] Adrian Ford and Alan Roberts „Color Space Conversions” Technical report, Westminster University, London,. August 1998, p. 31. URL: <http://www.poynton.com/PDFs/coloureq.pdf>, utolsó látogatás: 2012. október 30.

- [14] Srikanth Patala, Jeremy K. Mason, Christopher A. Schuh, "Improved Representation of Misorientation Information for Grain Boundary Science and Engineering", Progress in Materials Science, 2012, pages 1383-1425
- [15] Stephen M. Anthony and Steve Granick, „Image Analysis with Rapid and Accurate Two-Dimensional Gaussian Fitting”, University of Illinois, Urbana, Illinois 61801, , Langmuir, 2009, Vol. 25 (14), pp. 8152–8160
- [16] Emanuele Trucco, Alessandro Verri „Introductory techniques for 3D computer vision” Prentice Hall, 1998,
- [17] Jorge Santolaria, Juan – José Aguilar, David Guillomía, Carlos Cajal, “A Crenellated – target – based Calibration Method for Laser Triangulation Sensors Integration in Articulated Measurement Arms”, Robotics and Computer-Integrated Manufacturing Vol 27, April 2011, Pages 282-291
- [18] David Eberly „Least Squares Fitting of Data” Geometric Tools, LLC, 2008 pp. 1–10. URL: <http://www.geometrictools.com/> utolsó látogatás: 2012. október 30.
- [19] Elan Dubrofsky, “Homography Estimation”, MSc. diplomamunka, University of British Columbia, 2009, p. 32.
- [20] Dan Kalman, “A Singularly Valuable Decomposition: The SVD of a Matrix”, The American University Washington, DC 20016, February 13, 2002, p. 27.
- [21] Alpaslan Yufka and Osman Parlaktuna, “Performance Comparison of Bug Algorithms for Mobile Robots”, Eskisehir Osmangazi University, Turkey, 5th International Advanced Technologies Symposium, May 13-15, 2009, Karabuk, Turkey, p. 1-5.
- [22] Nora H Sleumer, Nadine Tschichold Grünman, “Exact Cell Decomposition of Arrangements used for Path Planning in Robotics”, Technical Report, Swiss Federal Institute of Technology, Institute of Theoretical Computer Science, September 29, 1999, p. 14.