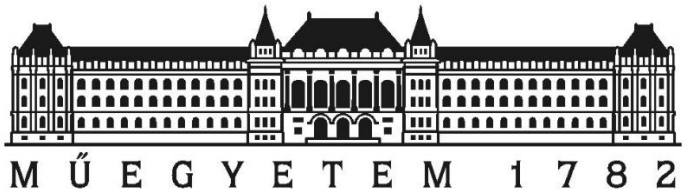


Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Irányítástechnika és Informatika Tanszék

TDK 2019

Haladó jármű környezetének adaptív maszkolása optical flow technikák használatával

Mészégető Tamás, H6LGMQ

meszeget.tamas@gmail.com

Konzulens: Szántó Mátyás

Tartalom

Abstract

This project aims to develop a sub-system of the cloud-based CrowdMapping social 3D mapping platform. The main objective of the subsystem is the realization of a 3D reconstruction of the environment of commercial vehicles (roads, parking lots, etc.), using inputs from a single camera and a GPS-IMU unit or other sensor providing displacement data. As an output, the subsystem provides a 3D point cloud which is uploaded to a cloud-based database for further processing.

The reconstruction combines multiple monocular depth prediction and 3D reconstruction methods to improve the quality of the output data. To help the subsequent reconstruction, based on an initial depth prediction - generated by a deep neural network – and displacement data, an expected optical flow field is calculated for subsequent frames. By comparing this to the actual optical flow field, the dynamic and static elements of the environment are separated.

The subsequent Structure from Motion algorithm considers only the static segments of the environment, fulfilling the basic condition for an unbiased reconstruction, potentially improving the accuracy and reliability of the output, with possible improvements in the execution time. The partial reconstructions are then sent to the cloud, where they are joined with the corresponding parts of the map, providing up-to-date segments. This map is then potentially used by autonomous vehicles for route or even for trajectory planning.

The current project realizes and integrates different elements of this processing scheme, evaluates the optical flow based static-dynamic segmentation and investigates the actual usability and characteristics of such a system.

For the development and evaluation of the above system the synchronised video, position and 3D point cloud data, as well as certain development kits contained in the „KITTI Vision Benchmark Suite” (provided by the a Karlsruhe Institut für Technologie and the Toyota Technological Institute at Chicago) were used, along with results from other projects related to the CrowdMapping architecture.

Bevezetés

A közeljövő közúti közlekedését illető várakozások alapvetően két területen, az alternatív meghajtások és az egyre teljesebb autonóm közlekedést illetően számítanak az iparágat alapjaiban átformáló változásokra. Míg az előbbit illetően lassan körvonalazódni látszik az akkumulátorokat használó elektromos személy- és tehergépjárművek elterjedése, illetve a kapcsolódó infrastruktúra jövőbeli kinézete, a különböző mértékű autonóm közlekedést lehetővé tevő funkciók épp csak megjelenőben vannak az sériagyártású autókban, míg a teljesen autonóm gépjárműveket egyelőre a fejlesztés kihívásai mellett tisztázatlan jogi kérdések is távol tartják az utaktól.

Az autonóm közlekedés megvalósítása során a legfontosabb részfeladatok közé tartoznak a gépjármű környezetének felmérése, érzékelése, a - bizonyos esetekben etikai kérdéseket is felvető - döntéshozási szituációk kezelése, a pályatervezés, valamint a megtervezett pályán való mozgáshoz szükséges szabályozások megvalósítása. A fentiek közül a dolgozatban az elsővel kapcsolatos kérdésekkel foglalkozom.

A jármű környezetérzékelésének elemei a környezet struktúrájának felmérése (mapping) illetve abban saját helyének meghatározása (localization). Alapvetően két csoportba sorolhatjuk az erről szerzett információkat: a GPS (Global Positioning System) segítségével meghatározott abszolút pozíció és egy megfelelő, járművön tárolt vagy online lekérdezett térkép segítségével a jármű az úthálózaton való lokalizációt végezheti el, ami valamilyen ismert, statikus környezetben való elhelyezkedést ad kimenetként. A térképek reprezentációtól függő részletességgel szolgáltathatnak információt a jármű úthálózaton elfoglalt pozíciójáról, az elérhető közúti sávokról, közlekedés irányokról és egyéb vonatkozó szabályozásokról, az utak pontos geometriájáról, az úthibákról, illetve az utak környezetéről. Ez utóbbi szintén számos releváns információt tartalmazhat, úgy, mint a kialakított parkolóhelyek elhelyezkedése, mérete, száma, vagy például az útpadka magassága, szélessége, amely valamilyen balesetelkerülő manőver esetén bírhat nagy jelentőséggel.

Ugyanakkor a környezet dinamikus és változó elemeit, elsősorban a többi közlekedőt, legyen az jármű vagy gyalogos, a parkoló autókat, valamint a váratlan útakadályokat ez az előre készített térkép nem tartalmazza, így a járműnek képesnek kell lennie saját közvetlen környezetének, mindenekelőtt az úttestnek és az azon elhelyezkedő vagy az a felé tartó objektumoknak az érzékelésére, felmérésére. Ehhez az járművek a GPS egységen túl további

szenzorokat, mint kamerákat, radarokat, ultrahangos szenzorokat vagy LIDAR-okat alkalmaznak. Ezek közül a kamerák kiemelt fontosságúak, hiszen a jelenlegi közlekedési infrastruktúrát is vizuális alapú tájékozódásra való tekintettel alakították ki, így például a közlekedési táblák, lámpák és útburkolati jelek érzékelése és értelmezése is elsősorban kamerák segítségével történik. Ennek köszönhetően kamerák gyakorlatilag minden, részben vagy egészben autonóm közlekedésre törekvő járművön megtalálhatóak.

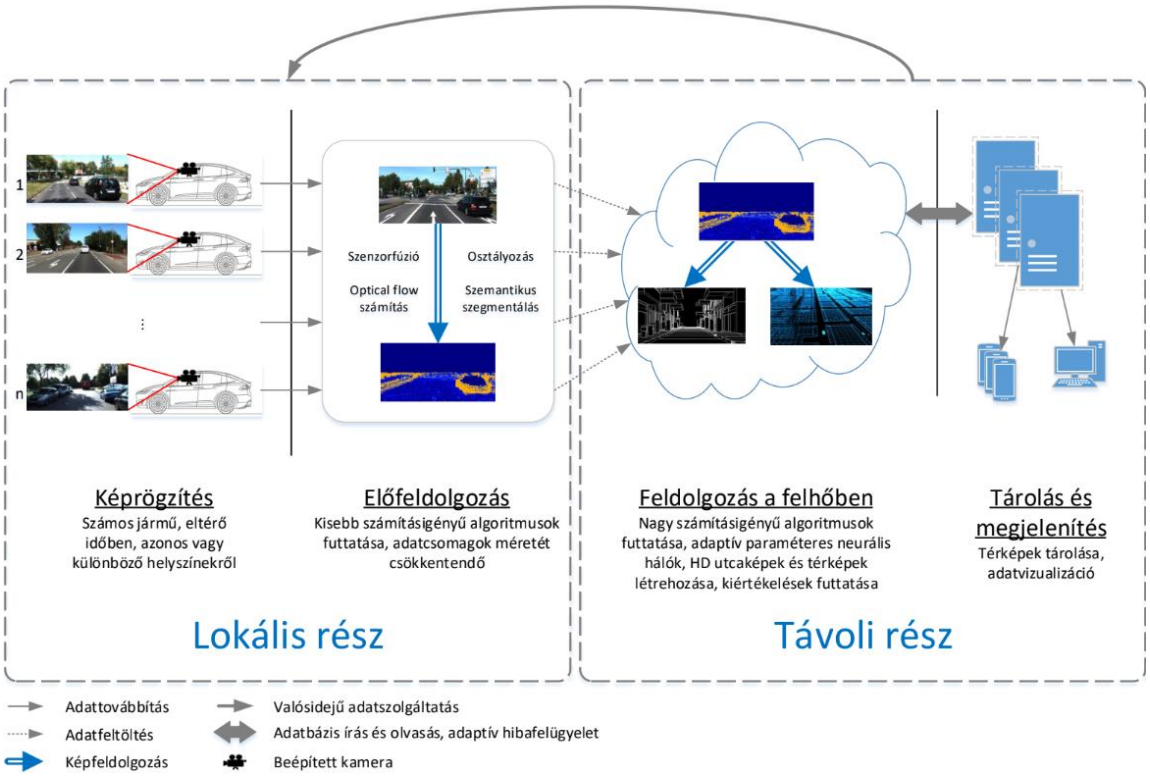
Az autonóm járművek közlekedését segítő részletesebb térképek szükségességét jól bizonyítja, hogy a területen olyan cégek aktívak, mint a korábban is navigációs eszközöket és térképeket készítő Tom-Tom, a német autógyártókkal együttműködő HERE Technologies, vagy a Google tulajdonában lévő Waymo. Jelen dolgozat egy hasonló célkitűzésű, az adatok gyűjtését közösségi közreműködés segítségével realizáló architektúra megvalósulását célzó, a Budapesti Műszaki és Gazdaságtudományi Egyetem Villamosmérnöki és Informatikai Karának Irányítástechnika és Informatika tanszékén futó projekt, a CrowdMapping keretei közt készült el. A következőkben röviden bemutatásra kerül a projekt és az annak keretei közt kijelölt feladat, a kapcsolódó irodalom és a felhasznált adatbázis, majd a konkrét megvalósítás, végül az eredmények.

1) CrowdMapping

Egy részletes, az út és az út környezetének dimenzióit is jellemző térkép hasznos lehet egy autonóm jármű számára, többek között segítheti, hogy előzetesen részletes pályatervezést végezzen, könnyebben parkolóhelyet találjon, vagy hogy felkészüljön a megfelelő manőverekre az esetleges veszélyzónákban. A részletes térképek elkészítése azonban az úthálózat rendkívüli kiterjedésére és folyamatos átalakulására való tekintettel nehéz és költséges feladat. Amennyiben ezt egy speciális szenzorrendszerrel ellátott, saját feltérképező flottával valósítanánk meg, úgy ez mind a városokban az alacsony átlagsebesség, mind az országutakon a nagy távolságok mellett igen jelentős flottát és magas üzemeltetési költségeket jelentene, ha kellően magas frissítési frekvenciát kívánunk elérni.

A CrowdMapping architektúra (1. ábra - CrowdMapping architektúra) motivációja egy olyan, lehető legkevésbé platformfüggő, minél több jármű bevonására alkalmas adatgyűjtés és -feldolgozás megvalósítása, amely a nagy számban bevont járművek révén egy mindig naprakész adatokat tartalmazó 3D-s térkép készítésére, tárolására és annak a felhasználókhoz való eljuttatására alkalmas. A lehető legnagyobb számú jármű bevonása a minél kevesebb

megkövetelt, minél elterjedtebb szenzor adataira való hagyatkozás által valósítható meg. Ezek a jelen esetben a korábban taglalt GPS – potenciálisan a megfelelő nagyfrekvenciás pontosság érdekében egy IMU (Inertial Measurement Unit) eszközzel kiegészítve, kamera, valamint hálózati kapcsolat. A bevont járművek az összegyűjtött adatokat részben helyben dolgozzák fel, majd ezt az előzetes feldolgozás során előállt, a konkrét képsorozatoknál lehetőleg kisebb adatcsomagot jelentő rész rekonstrukciót a rendszer felhőalapú részébe továbbítják. Itt az egyes rész rekonstrukciók a GPS adatok és más illesztési eljárások segítségével állnak össze az úthálózat környezetét jellemző térképpé.



1. ábra - CrowdMapping architektúra [25]

A TDK dolgozat keretei közt a járműveken történő előfeldolgozás megvalósítását célzó rendszer prototípusa készült el, amely a fenti sajátosságok miatt egyetlen kamera és egy pozíció-érzékelő szenzor jeleire hagyatkozva kísérel meg megfelelő bemenetet előállítani a további feldolgozáshoz. Az egykamerás, monokuláris 3D rekonstrukció dinamikus környezetben történő végrehajtásának korlátjai miatt ezt a mozgó objektumok maszkolásával, csak a statikus térrészekre vonatkozóan kerül végrehajtásra, így potenciálisan javítva és gyorsítva a működést. A kisebb térrészről nyert információ a rendszerbe potenciálisan nagy számban integrált adatgyűjtő jármű miatt nem okoz jelentős problémát, plusz feldolgozási feladatokat pedig elsősorban a távoli részben, offline körülmények közt eredményez.

2) Szakirodalmi áttekintés

Autonóm közlekedés

A vezetéssegítő-rendszerek (Driver Assistance System – DAS) fejlődése több évtizedre tekint vissza, egyik első példaként az 1978 óta sériagyártásba kerülő blokkolásgátló (ABS) említhető. A DAS rendszerek eddigi és potenciális jövőbeli fejlődését áttekintő munkájukban Bengler et al. (2014) [1] kitérnek mind a szenzorrendszerek, mind a célok és a megvalósított funkciók fejlődésére. Tagolásuk szerint míg az első vezetéstámogató rendszerek elsősorban a jármű belső diagnosztikájára, állapotát figyelő szenzorokra (kerékelfordulás szenzor, gyorsulásmérők) hagyatkoztak, addig a következő generációs megoldások már a környezet érzékelésére támaszkodva valósítottak meg olyan komplex funkciókat, mint a parkolássegéd vagy az adaptív távolságtartó tempomat (sebességtartó automata). A különböző távolságérzékelő szenzorok mellett ezek a megoldások elsősorban kamerákra támaszkodnak. A jövőbeli fejlődést elsősorban a járművek hálózatba való bevonásával, a jármű-jármű (V2V) és a jármű és egyéb rendszerek közti (V2X) kommunikáció fejlesztésével látják megvalósíthatónak.

Az autonóm közúti közlekedés lehetőségének jelentőségét, egyszersmind aktualitását jól mutatja Maurer et al. (2016) [2] a problémakör egyes technológiai aspektusai mellett annak jogi és társadalmi vonatkozásaival is foglalkozó kötete. Ennek egyik tanulmánya is fontos célként említi meg a járművek egymás felé történő információszolgáltatását, azzal a céllal, hogy egy, az aktuális feladat optimális végrehajtásához szükséges térkép álljon az autonóm járművek rendelkezésére.

Hasonló gondolat áll az [1] által említett DRIVE C2X projekt mögött, amely szintén az úthálózat részletes, dinamikus térképének létrehozását célozza. A jövőbeli fejlődésre való kitekintéskor idézi Nothdurft et al.-t (2011) [3], miszerint a pontos lokalizációban a jövőben nagy szerepet játszhatnak a részletgazdag háromdimenziós térképek. Ezeknek, már csak az adattömegek mérete miatt is, automatizált generálása és frissítése is fontos feladat. Emellett a fejlődést nem elsősorban az újabb szenzorok megjelenésében, hanem az adatgyűjtés, értelmezés és feldolgozás koncepcióinak megújulásában látják a szerzők.

Monokuláris 3D rekonstrukció

A CrowdMapping célja a minél nagyobb számú jármű bevonása, ebből a szempontból a monokuláris rekonstrukció kedvező lehet. A sztereó kamerarendszerhez, vagy más specifikus szenzorokhoz hasonlítva olcsóbb, és elterjedtebb vezetéstámogató funkciókat -például táblafelismerés - megvalósító járműveken is megtalálható legalább egy, jellemzően a szélvédőre rögzített kamera.

A monokuláris 3D rekonstrukciós eljárások több stratégiát követnek. Mivel a képalkotáskor történő leképezés dimenzióvesztéssel jár, így egyetlen kamera egyetlen felvételéből a kamerakalibráció ismeretében sincs lehetőség a környezet rekonstrukciójára. Korábban, az önálló laboratórium [4] keretében tekintettem át a monokuláris rekonstrukciós eljárásokat, amelyeknek az alábbi módszerei különíthetők el:

- Klasszikus, mozgásalapú megoldások
    - Structure From Motion (SfM, illetve valós időben jellemzően a vSLAM – visual Simultaneous Localization and Mapping – kifejezéssel illetik) eljárások: több, átfedésben lévő képre hagyatkozva, a képeken megtalálható képjellemzők kinyerésével, egymáshoz rendelésével, majd a kamerapozíciók és a környezet pontjainak elhelyezkedésének becslésével történik a rekonstrukció. Bár az SfM végrehajtására sokféle módszer létezik (Tomasi and Kanade 1992 [5], Song et al. 2016 [6], Nyimbili et al. 2016[7]), alapvetően része a feldolgozási láncnak a csoportigazítás művelete, amelynek segítségével az újabb képeken talált információ segítségével a környezet pontjainak korábban kiszámított pozíciója tovább pontosítható. A csoportigazítás műveletének számításigénye és a mozgó jármű készítette képek közti átfedés időbeli korlátozottsága miatt ezt egy mobil robot vagy gépjármű esetében csak adott számú képkockára korlátozódik. A kamerának tehát mozognia kell (több, különböző pozícióból kell felvételeknek készülniük), a környezetnek azonban eközben statikusnak kell lennie. Könnyen belátható, hogy utóbbi feltétel a közúti közlekedésben nem minden esetben adott. A metrikus rekonstrukcióhoz, valamint annak pontos elhelyezéséhez az úthálózat globális térképén egyes kamerapozíciók vagy képeken beazonosítható objektumok pontos elhelyezkedésének ismerete szükséges, amelyek közül jelen alkalmazásban előbbire kézenfekvőbb támaszkodni.
    - Interframe stereo: a klasszikus sztereó képalkotáshoz hasonlóan, ismert kamerapozíciók és kamerakalibráció esetén szintén rekonstruálhatjuk a

környezetet, tulajdonképpen egy csonka SfM eljárásként fogható fel, amely a háromszögelés művelete után nem kísérli meg a talált 3D-s pontok további finomítását, így valósidejű futtatásra adott esetben alkalmasabb lehet. A torzításmentes rekonstrukció feltétele ebben az esetben is, hogy környezet elemei nem mozdulnak el egymáshoz képest a két kép közt. Ilyen eljárásokkal dolgoznak például (Kuang et al. 2018)[8] vagy (Hasan et al. 2012)[9]. .

- Optical flow: Az optikai áramlás is felhasználható a saját mozgás becslésére, vagy a mozgás ismeretében a környezet struktúrájának meghatározására. Az optikai áramlás alkalmas nagyszámú képpont közti kapcsolat gyors kiszámítására, így segítve a valós idejű végrehajtást, beépülve valamely SLAM eljárásba. (Newcombe and Davison 2010 [10], Suhr et al [11], Mitiche and Sekkati 2006 [12],)

- Tanuló algoritmusok
  - Amikor csupán az adott helyzetben előálló konkrét bemenő adatok feldolgozására hagyatkozunk, mindig több képre van szükség a 3D rekonstrukció elvégzéséhez. Azonban a valamilyen a-priori információk alapján hangolt tanuló algoritmusok, amelyek a számítógépes látásban sok területen alkalmazott mély neurális hálók, adott esetben akár egyetlen bemeneti képre is képesek megbecsülni az adott, képen megjelenő térrész struktúráját, azaz az egyes pixelek által ábrázolt felületek távolságát Saxena et al. [13] (SIDE – Single Image Depth Estimation). Ez a mélységkép a kameraparaméterek ismeretében könnyen 3D pontfelhővé konvertálható. Ilyen eljárások fontos jellemzője, hogy olyan képjellemzőket is felhasználhat, mint a vonalélesség, a textúrák vagy a homályosság, amelyeket az SfM típusú algoritmusoknál, ahol inkább lokális jellemzőkkel dolgozunk, nem veszünk figyelembe. A vonatkozó tanulmányban fel is használják, hogy eme mélységkép eredményei nagyban függetlenek valamely sztereó eljárás eredményeitől, így az eredmények hatékonyan fuzionálhatók egy pontosabb globális mélységkép kinyerése érdekében [13]. Egy ilyen megoldás például jobban kezelheti a kamera mozgását kihasználó algoritmusok számára problémás nagyon távoli vagy homogén területeket. Külön előnye más monokuláris eljárásokkal szemben, hogy az egyetlen képre kapott válasz nem lesz terhelve a mozgó objektumok okozta strukturális zavarásoktól.

Általánosságban elmondható, hogy a SLAM vagy SfM feldolgozási módszerek elterjedtek, sok verziójuk létezik, különböző alkalmazási területekre (online végrehajtás, részletesebb offline rekonstrukció) optimalizálva. Legfőbb hátrányuk jelen alkalmazásban, hogy a közlekedés többi résztvevője mind mozog, így a rekonstrukció torzult lesz, illetve amennyiben a nagy hibával kiszámolt pontpárok elutasításra kerülnek (outlier rejection), akkor ezen képjellemzők megkeresésére, leírójuk definiálására és párosítására, valamint a csoportigazítására feleslegesen fordítunk erőforrást és időt a feldolgozás során.

Hogy a kiforrott és sokrétű SfM algoritmusok pontosságát felhasználhassuk, célszerű kijelölni a kép azon szegmenseit, amelyek a statikus környezetet ábrázolják, s az algoritmust csak ezen a régión futtatni. A konkrét alkalmazás esetében ráadásul a mozgó, dinamikus objektumok az út környezetének olyan változó elemei, amelyeket egy ilyen térképbe nem szeretnénk megjeleníteni. A mozgásalapú szegmentálás statikus kamera esetében egy egyszerű különbségkép számításával és küszöbözéssel megvalósítható, noha ennek eredménye például a háttér várható változásainak modellezésével finomítható. Szemenyei 2018 [14] Mozgó kamera esetében azonban a kép minden része változik - kivéve azon objektumok képét, amelyeknek térbeli konfigurációja nem változik meg a kamerához rögzített koordinátarendszerben. (Emellett lokálisan állandó lehet a kép, ha valamely homogén szegmensének elmozdulása az adott régióban változatlan képet eredményez, de épp ezeket a mozgásokat szeretnénk detektálni.)

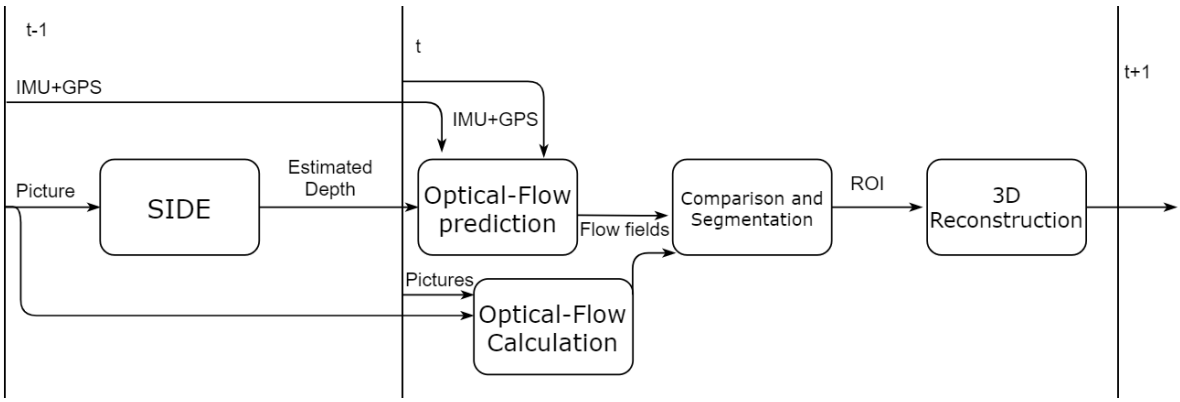
Az egymást követő képeken érzékelhető kis lokális elmozdulásokat az optical flow vagy optikai áramlás mező jellemzi. Amennyiben előzetes ismeretünk van a kamera által rögzített térrész struktúrájáról, valamint van információnk a jármű, azaz a kamera mozgásáról, akkor adhatunk egy becslést, hogy hogyan fog kinézni az optical flow mező. Ennek a predikciónak a kiszámítása után meghatározásra kerülhet az egymást követő képkockák közti „valós” optical flow. Természetesen a képek hibái, valamint a választott optical flow számító algoritmus jellegzetességeinek megfelelően ez is zajjal, illetve hibákkal terhelt lesz. Ezt a két mezőt összehasonlítva, és a számított, valamint az előrejelzett flow-mezők közt valamilyen távolságmátrixot definiálva, majd ezt küszöbözve meghatározható egy bináris mátrix, amely a kép mozgó, illetve nem mozgó objektumokat tartalmazó szegmenseit jelöli ki.

Ehhez hasonló elgondolást ír le (Klappstein et al. 2009) [15]: megoldásukban szintén egy optical flow predikciót vetnek össze a valódi flow mezővel, egyes alkalmazásokban bíztató eredményekkel, de az összehasonlítás alapjául vett sztereó megoldástól elmaradó teljesítménnyel – ugyanakkor kisebb számítási igénnyel. Megoldásuk ugyanakkor ritka flow

mezők összevetésén és a kép ezt követő szegmentálásán alapul. Míg a jól beazonosítható képjellemzőkre megbízhatóbb eredményeket szolgáltatnak mind az optical flow, mind a rekonstrukció során, kérdéses, hogy az ezt követő szegmentáció kevés pont alapján mennyire képes a releváns objektumok kiemelésére. Ezzel szemben jelen megoldás két sűrű mező összehasonlításán alapuló szegmentálást javasol. Ugyanakkor a valós környezet objektumainak figyelembevételével ezt érdemes lehet kiegészíteni egy másik képszegmentáló eljárással, a mozgó objektumok pontosabb kijelöléséért.

Az előzetes mélységképet potenciálisan a másik, kevésbé kiforrott, ugyanakkor dinamikusan fejlődő megközelítés, valamely tanuló algoritmus szolgáltatná. Hasonló körülmények közötti alkalmazásra is számos példa van. A fejlesztés során felhasznált KITTI adatbázishoz (lásd 4.) csatlakozó benchmarkok egyike a „Depth prediction", amely mélységképek generálására szolgáló algoritmusok kiértékelésére hivatott. Az itt felsorolt eljárások közül jó példa a SIDE-ra Ren et al. (2019) [16] munkája. A legjobb megoldások GPU-n 0,1 szekundumos futásidő mellett 2%-os négyzetes relatív hibával valósítják meg a feladatot, ez remélhetőleg kellő pontosságot biztosít a szegmentáció végrehajtásához.

A leírt eljárásnak mind a valósidejű, mind a nem valósidejű megvalósítása releváns lehet. A valósidejű nem szolgál magyarázatra, hiszen a környezetérzékelés fontos feladata az autonóm járműveknek. Az eljárás lehetőséget ad egy, a CrowdMapping architektúrához kapcsolódó járműnek egy hosszabb útszakasz feldolgozására és az adatok folyamatos továbbítására. Emellett az olyan eredményekkel is szolgálhat, amelyet a jármű helyben használ fel: a statikus környezet felmérésére, valamint a mozgó objektumokat tartalmazó veszélyzónák kijelölésére és azok más, esetlegesen számításigényesebb eszközökkel való vizsgálatára ad lehetőséget. Az online vagy real-time feldolgozás ütemezését a 2. ábra - Valósidejű végrehajtásszemlélteti (t-1, t és t:+1 mintavételi időpillanatok):



2. ábra - Valósidejű végrehajtás

A CrowdMapping szempontjából ugyanakkor az sem probléma, ha az adott hardver nem képes valós időben végrehajtani a fenti lépéseket: ilyen eset előfordulhat például, ha az adott jármű csak kisebb számítási kapacitást igénylő vezetéstámogató funkciók megvalósítására képes. Ebben az esetben néhány másodpercnyi adat összegyűjtését, feldolgozását és továbbítását követően a jármű egy közeli, de nem közvetlenül kapcsolódó térrészen végzi el újra a fenti műveletet, így is értékes friss adatokkal frissítve a felhőben tárolt térképeket.

A dolgozat egyfajta proof-of-conceptként kíván szolgálni: a feldolgozás különböző elemeit, így egy-egy egyszerűbb, könnyen hozzáférhető optical flow számító algoritmust és SfM eljárást integrálva, valamint egy, szintén a projekt keretei közt elkészülő mélységbecslő háló kimenetét felhasználva vizsgálom meg a fentebb leírt mozgásalapú szegmentációs eljárás lehetőségeit, illetve annak a rekonstrukció kimenetére és futásidejére való hatását.

3) Mélységkép becslése képből, illetve képsorozatból konvolúciós neurális háló segítségével

A fentebb leírtak szerint a szegmentálás teszteléséhez felhasznált eljárás szintén a Crowdmapping keretei közt kerül kifejlesztésre. A megoldás egy konvolúciós neurális háló (CNN) segítségével valósítja meg az előzetes mélységkép becslését. A háló az egymás után feldolgozott képekből származó többletinformációt LSTM (Long-Short Term Memory) cellák alkalmazásával hasznosítja, így amennyiben egy adott szcenárióban egymás után több különböző pozícióból készített képet kap bemenetként, úgy a kimenet egyre kisebb hibával rendelkezik. A ~70 millió tanítható paraméterrel rendelkező modell „U-net” struktúrájú, ahol a konvolúciós rétegek homokóra szerűen, a képméretet mindkét dimenzióban 1/32 részére csökkentik, majd visszanövelik a kép eredeti méretére. Mind a hat „csökkentő” réteg előre van csatolva egy LSTM réteggel a vele megegyező méretű, növekvő ágban lévő réteghez, ez lehetővé teszi, hogy mind a nagy, egész képet érintő változások, mind a kisebb, lokális folyamatok is modellezve legyenek. (Tass [17])

A tanuló eljárás optimalizálásához a szegmentálás során is felhasznált KITTI Vision Benchmark Suite vonatkozó adatsorai kerültek felhasználásra, amely összesen 93 ezer ritka mélységképet jelent. (Uhrig et al. 2017 [18]). Az eredeti, rektifikált képek kevesebb mint fele méretre átméretezve, az adatrögzítés sajátosságaiból adódóan jellemzően érvényes adat nélküli felsőbb régiók elhagyásával lettek felhasználva. A be és kimenet felbontása így egyaránt

128x576 pixel, az átméretezés során a kép szélességéből szimmetrikusan nagyjából 7%-ot, magasságából, csak a kép felső részéről pixeleket elhagyva, nagyjából 30%-ot elhagyva. A bemenet az átméretezett, célszerűen egymás után rögzített RGB képek, a kimenet pedig minden egyes képre egy mélységkép, amelynek minden pixeléhez tartozik érték. A munka során a fejlesztés alatt álló modell egy korábbi, addigi legkedvezőbb tulajdonságokat mutató verziója (1) került felhasználásra, melynek jellemzőit, egy későbbi iterációéval (2) együtt, az 1. Táblázat tartalmazza.

1. Táblázat - A felhasznált mélységbecslő CNN tulajdonságai

Sorszám	sparse NMSE [-] (Normalised Mean Square Error)	sparse NMAE [-] (Normalised Mean Absolute Error)
1	0.0732	0.1683
2	0.0550	0.1295

4) KITTI Vision Benchmark Suite

Az előző feladatban felvázolt megoldáshoz, azaz a kép mozgó és statikus részei közt végrehajtott szegmentáláshoz és a környezet rekonstruálásához szükséges bemeneti jelek a következők:

- egy kamera képe, valamint
- egy GPS-IMU egység – vagy hasonló funkciójú szenzorok által szolgáltatott pozíció illetve elmozdulás adat

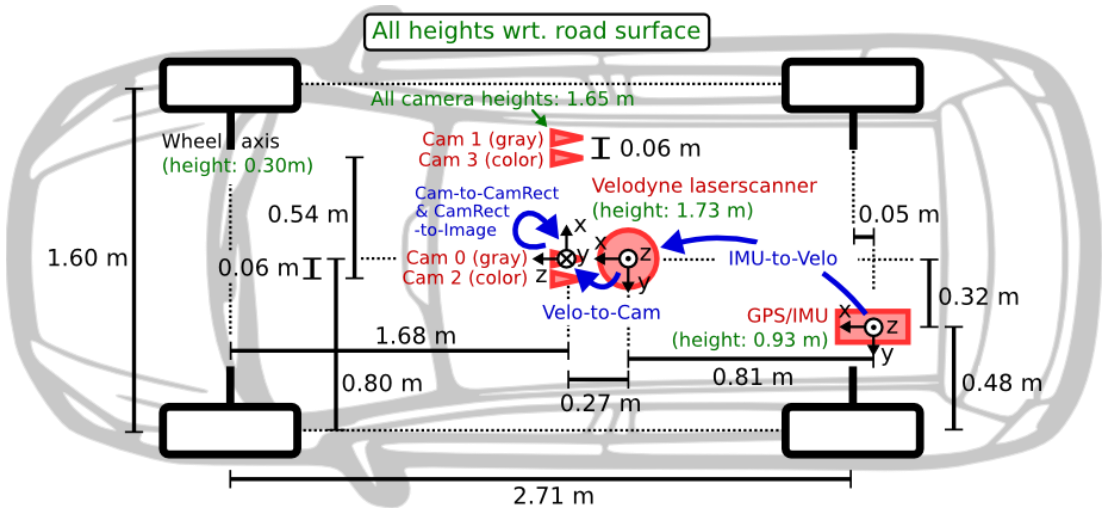
A kamera külső-belső paramétereinek ismerete is szükséges, hiszen ezek jellemzik a képalkotást, illetve a kamera konfigurációját a GPS-IMU egységhez rendelt koordinátarendszerben. Amennyiben a kapott eredmények - például távolságértékek vagy 3D koordináták – helyességét ellenőrizni kívánjuk, úgy ezt az információt szolgáltató szenzorok jeleinek ismeretében tehetjük meg közvetlen módon. Tipikusan ilyen a környezet objektumaira lézerfényt kibocsátó, majd a visszaverődő fényt mérő, 3D pontfelhőt szolgáltató LIDAR (Light Detection and Ranging).

Ilyen adatok megfelelő minőségben és mennyiségben történő előállítása és rögzítése költséges és időigényes feladat. A dolgozatban nem is általam rögzített mérések eredményeivel, hanem a különböző képfeldolgozási feladatokhoz - pl. odometria, objektumkövetés, mélységbecslés - az

előkészített, szinkronizált adatok mellett benchmarkokat is kínáló, számos vonatkozó kutatáshoz felhasznált „KITTI Vision Benchmark Suite” – ot használtam fel (Geiger et al. 2012 [19], Geiger et al. 2013)[20]. A http://www.cvlibs.net/datasets/kitti/ címen elérhető adatbázis az alábbi szinkronizált adatokat tartalmazza:

- 2 színes és 2 szürkeárnyalatos kamera képe
- minden képkockához tartozóan pozícióadat
- minden képkockához tartozóan 3D pontfelhő

A méréshez felhasznált szenzorrendszert a 3. ábra – A felhasznált adatok rögzítésére használt mérőkocsi szenzorainak elrendezéseszemlélteti. Az adatbázisban további feladatspecifikus adatsorok, például az egyes képkockákon található objektumokat jellemző információ, vagy az egyes képkockák közti valódi optikai áramlás (*ground truth*) (Menze and Geiger, 2015) [21] szintén elérhetőek.



3. ábra – A felhasznált adatok rögzítésére használt mérőkocsi szenzorainak elrendezése (Forrás [22])

Az adatok további feldolgozását a szükséges kalibrációs információk mellett MATLAB és C++ fejlesztőeszközök könnyítik meg. A munka során elsősorban MATLAB környezetben dolgoztam, és a nyers, a mélység, valamint az optikai áramlás adatokhoz tartozó fejlesztőkészleteket [22] (development kit) használtam fel. Ezek elsősorban az adatok további feldolgozásra alkalmas formátumban történő beolvasását, a szükséges konverziók megvalósítását és az adatok közti kapcsolat megteremtését segítik, mint például a LIDAR pontfelhők tetszőleges kamera képsíkjára történő vetítése.

5) Optical flow predikció és a mozgásalapú szegmentálás

Áttekintés

A korábban ismertetett feldolgozás lépései röviden:

- Adottak egymás utáni kameraképek, illetve a kamerák pozícióit jellemző adatok, valamint a kamerák külső és belső paraméterei
- Egy SIDE vagy hasonló módszerrel becslést kapunk a környezet struktúrájára egy mélységkép formájában. Ez jelen esetben egy, szintén a projekt keretei közt megvalósuló, konvolúciós neurális hálóra épülő megoldással valósul meg
- Felhasználva a becsült mélységinformációt és a kameraparamétereket, becslést adhatunk az egyes pixelekre leképzett objektumok háromdimenziós pozíciójára
- Feltételezve, hogy ez a háromdimenziós pozíció nem változik, azaz a környezet statikus, a kamera következő időpillanatban elfoglalt pozíciójának ismeretében meghatározhatjuk ugyanezen pontok pozícióját az új kamera-koordinátarendszerben
- A transzformált 3D pontfelhőt a képsíkra vetítjük, majd meghatározzuk az egyes képpontok elmozdulásvektorait az egymást követő képkockák közt
- A két egymást követő képkocka közt valamilyen alternatív, csupán a képeket felhasználó módszerrel is kiszámítjuk az optikai áramlás mezőt. Mivel a kép minél nagyobb részének lefedése a cél, ezért ennél a prototípusnál a Farneback-algoritmus került felhasználásra, amely meglehetősen sok képpontra ad valamilyen eredményt, azaz egy sűrű optical flow mezőt szolgátat kimenetként
- A két optical flow mező összehasonlítása a vektorok közti valamilyen távolságérték definiálásával történik. Az abszolút vagy relatív távolságot küszöbözzük, így kapva egy, a kép méretével megegyező bináris mátrixot
- A néhány pixel méretű, a környezetüktől eltérő értéket felvevő pixeleket a nyitás és zárás műveletével szűrhetjük: ekkora méretű objektumok amúgy is vagy nagyon elhanyagolhatóan kicsik, vagy nagyon távol vannak, mindenesetre érdemben nem befolyásolják a 3D rekonstrukciót
- A végrehajtás utolsó lépése a 3D rekonstrukció megvalósítása, egy SfM vagy SLAM algoritmussal, amely a számításhoz csak a képek statikus régióin talált képjellemzőket használhatják fel. Alapvetően a prototípusban itt is az adott fejlesztőkörnyezetben könnyen beépíthető alternatívára esett a választás

Fontos meghatározni, miképp lehetséges a javasolt megoldás értékelése. Amellett, hogy a legfontosabb kérdés az, hogy javítható-e a végső rekonstrukció valamely paramétere, amennyiben korlátozzuk a feldolgozott képszegmenseket a statikus részekre, különállóan érdekes lehet a mozgásalapú szegmentáció hatékonyságának megítélése.

Az optikai áramlás predikciója

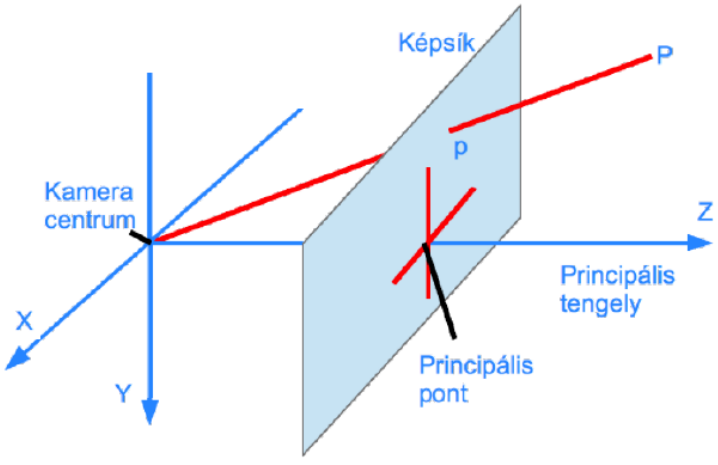
Az optikai áramlásra a fentiekben leírtak alapján egy

- ismert előzetes mélységkép, valamint
- ismert kamera elmozdulás (ego motion)

ismeretében szeretnénk becslést adni. A részfeladatok a mélységképből egy 3D pontfelhő generálása, a pontok kamerához viszonyított relatív pozíciójának transzformálása a két képkocka közt eltelt idő alatt történt elmozdulásnak megfelelően, a pontok visszavetítése a képsíkra, majd az egyes pontok esetében az optikai áramlás meghatározása.

3D pontfelhő generálása

A lézerszkenner által rögzített pontfelhő (a LIDAR forgásából adódó torzulásokat korrigálva), rendelkezésre áll az adatbázisban, ahogyan a rektifikált kamera-koordinátarendszerekbe történő homogén transzformációt és a rektifikált képsíkra történő vetítést leíró mátrixok is. A képsíkra történő vetítés ez alapján a pinhole-kameramodellnek megfelelően valósítható meg [20], így a kamera belső (intrinsic) paramétereit tartalmazó mátrixszal való szorzás után a mélységparaméterrel normalizálva kapjuk meg a pixelkoordinátákat.



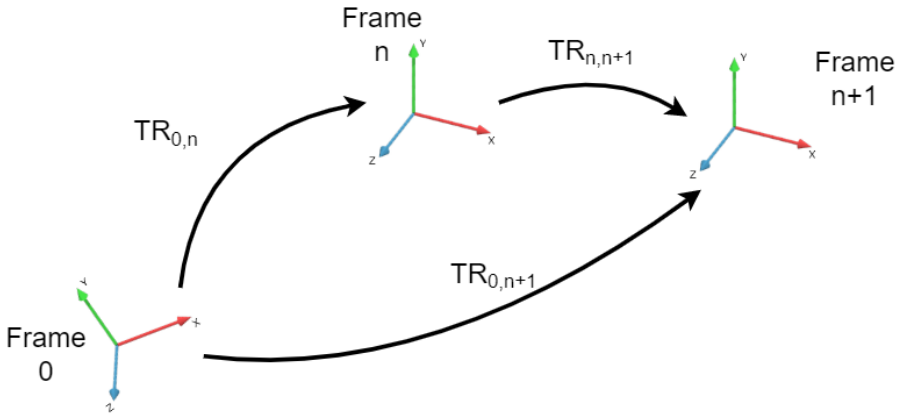
4. ábra - Pinhole kamera geometriai modellje, kamera-koordinátarendszer (Forrás: [14])

Amennyiben a vetítés során elvesztett 3. dimenzióra vonatkozó adat is rendelkezésre áll a pixelkoordináták és a kamerakalibráció mellett, úgy a művelet visszafordítható, az adott pixelre

leképezett 3 dimenziós pont rekonstruálható. Mivel azonban a pixelek nem ideálisak, azaz kiterjedéssel rendelkeznek, *ezért a visszavetítés sugarát körülvevő, a pixel alakjának megfelelő alapú, a szenzortól távolodva egyre nagyobbra nyíló gúla alakú térrész jelöli ki az adott pixelre leképezett pont lehetséges helyét*. Ez a generált és az adatbázisban található mélységképek esetén problémát jelent, hiszen a visszavetítés kiindulópontját egész pixelkoordinátákban ismerjük. A visszavetítés és így az optical flow vektor dimenzióinak meghatározása ugyanakkor szubpixeles pontossággal történik.

A pontfelhő transzformálása az új kamera-koordinátarendszerbe

A következő lépésben meg kell határozni, hogy az előző képkockához tartozó 3D pontfelhő elemei milyen koordinátákat vesznek fel a következő kép készítésének idejére már elmozdult kamera koordinátarendszerében. A jármű egyes időpontokban felvett konfigurációit a kiindulási GPS-IMU egység koordinátarendszerében ismerjük – azaz az első képhez tartozó rotációs mátrix egy egységmátrix, a transzlációs vektor pedig $\underline{0}$. Keresett az n-edik és n+1-edik konfigurációk közti kapcsolatot leíró homogén transzformáció.



5. ábra - A pozícióadatok szemléltetése

Az 5. ábra - A pozícióadatok szemléltetése alapján - ahol $TR_{0,n}$, $TR_{0,n+1}$ rendre a 0. és az n-edik illetve az n+1-edik képkockákhoz tartozó koordinátarendszerek közti ismert transzformációkat jelentik - a keresett $TR_{n,n+1}$ transzformációt leíró mátrix az alábbi módon számítható ki [Lantos 23]: (a nyíl irányával ellentétes irányban történő átmenetet a megfelelő mátrix inverze írja le)

$$TR_{n,n+1} = TR_{0,n}^{-1} TR_{0,n+1} \tag{1}$$

A pontokat először a kalibrációnak megfelelően egy homogén transzformációval a megfelelő kamera n. időpillanatbeli koordinátarendszeréből a GPS-IMU egység megfelelő koordinátarendszerébe transzformáljuk. Ezután a GPS-IMU egység n+1. időpillanatbeli

koordinátarendszerében írjuk fel a pontokat, amihez a saját mozgás kompenzálásának megfelelően $TR_{n,n+1}$ inverzével kell szorozni, majd egy újabb lépéssel a megfelelő kamera-koordinátarendszerbe jutunk. A teljes műveletsor:

$$P_{n+1} = TR_{Cam,IMU}^{-1} TR_{n,n+1}^{-1} TR_{Cam,IMU} P_n \qquad (2)$$

Ahol $TR_{Cam,IMU}$ a megfelelő kamera-koordinátarendszer és a pozícionáló egység koordinátarendszere közti kapcsolatot leíró homogén transzformáció, P_i pedig egy pont i-edik kamerakoordinátarendszerben adott pozíciója.

Visszavetítés a képsíkra és az optikai-áramlás kiszámítása

A megfelelő pontokhoz tartozó új pixelkoordinátákat a kalibrációs paraméterekkel definiált vetítési mátrix felhasználásával kapjuk meg. Az így kapott, rendre az i-edik képhez tartozóan ugyanazon háromdimenziós pont képét reprezentáló u_i és v_i pixelkoordináták segítségével az i-edik kép $[u_i, v_i]$ pontjához rendelt v_x és v_y optical flow vektorok rendre a

$$v_x = u_{i+1} - u_i \qquad v_y = v_{i+1} - v_i \qquad (3)$$

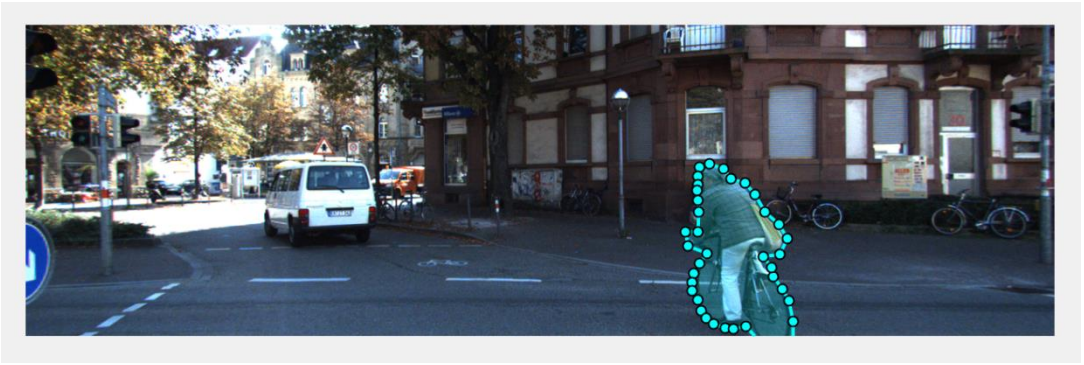
módon számíthatóak. A kiindulási mélységkép miatt u_i és v_i koordináták csak egész, míg u_{i+1} és v_{i+1} racionális számok lehetnek. Az optikai áramlás opticalFlow objektumban formájában reprezentálható MATLAB környezetben, ahol a v_x és v_y értékekkel feltöltött, a kép felbontásának megfelelő méretű mátrixokkal való inicializálás után a vektorok orientációja és nagysága is elérhető az objektum attribútumaként.

A mozgásalapú szegmentáció megvalósítása

Mielőtt a szegmentálás részleteibe bocsátkozom, fontos tisztázni, hogyan lehet egy ilyen szegmentálási eljárást kiértékelni. A szegmentálás minősítése feltételezi, hogy rendelkezésre áll olyan adat, ami a képeken látható objektumokat azok környezethez viszonyított sebessége alapján jellemzi (minden pixel a „mozog” vagy „nem mozog” osztályhoz van besorolva). Ilyen adatok a felhasznált adatbázisban nem voltak jelen, így azok előállítása a megfelelő kiértékeléshez szükségszerű volt.

Az adatgenerálásnál figyelembe kellett venni, hogy csupán egy képről az ember sem tudja megítélni minden esetben, hogy vajon egy, a képen látható objektum mozog-e a környezetéhet képest. Így a feladat elvégzéséhez a következő képkocka felhasználása is szükséges. Az erre a célra készített alkalmazás használatakor az operátor a mozgó részek kijelölése előtt tehát váltakozva látja a két képet, így meg tudja ítélni, a képen látható objektumok közül melyiknek van nullától különböző sebessége a statikus környezethez képest. Ezek után a megfelelő számú

sokszöggel körül határolja ezeket az objektumokat, amely körülhatárolt régió minden pixele a mozgó objektumot ábrázolóként lesz eltárolva. Az szegmentáló alkalmazás használatát a 6. ábra - A mozgó régiók kijelölése a viszonyítási alapnak felhasznált maszk előállításáhozszemlélteti.



6. ábra - A mozgó régiók kijelölése a viszonyítási alapnak felhasznált maszk előállításához

Noha a képeket nagy méretben van lehetőség megjeleníteni, nyilvánvaló, hogy ezzel az eljárással dolgozva lesznek helytelenül besorolt pixelek, ami a kiértékelési folyamatba hibát visz. Ugyanakkor ez a kompromisszumos megoldás megfelelőbbnek tűnt nagyobb mennyiségű adat gyors előállítására, a koncepció működőképességének ellenőrzéséhez.

A kiértékelés szempontjából kritikus még, hogy mely feldolgozási lépés pontossága, mely bemeneti adatok hibáinak, zajtartalmának mértéke az, ami jelentősen befolyásolja a szegmentálás minőségét. Az hatások elkülönítésére az

- optikai áramlást egymás utáni képekből becslő eljárást, illetve az
- optikai áramlás predikciójához felhasznált mélységkép-becslő megoldást

illetően van szükség. Minkét adatot illetően rendelkezésre állnak az adatbázisban viszonyítási alapnak vehető *ground truth* értékek, így a szegmentálás kiértékelése az alábbi összeállításokban történt meg:

1. *Ground truth*, azaz a rendelkezésre álló legpontosabb adatok mind az egyes pixelekhez tartozó távolság, mind az optical flow számítás esetében. Ezek egyrészt közvetlenül a szenzor adatai, amelynél egy ezeket az adatokat felhasználó tanuló algoritmus alapvetően nem adhat pontosabb eredményt (ugyanakkor figyelemreméltók lehetnek a szenzor által esetlegesen rosszul kezelt, pl. fényes vagy tükröződő felületeket ábrázoló képszegmensekre kapott eredmények), másrészt az offline, utólagosan előállított, kézzel validált optical flow mezők. Ez tulajdonképp a maszkolás – az alkalmazott szenzorok

szintjén – elérhető legjobb eredménye, a bemenő adatok így a legkevésbé terheltek különböző torzításokkal, zajokkal.

2. A becslések, azaz az integrálandó mélységbecslő és optical flow számító algoritmusok használata, így a maszkolás eredménye mindkét részfeladat hibáitól terhelt lesz. Ez – potenciálisan – a legrosszabb eredményeket szolgáltató összeállítás.
3. A távolságadatok pontosak, az optical flow kiszámítása a később felhasználandó eljárással történik. Ebben a konfigurációban az optikai áramlás meghatározására felhasznált eljárás tulajdonságainak a szegmentálás minőségére gyakorolt hatása szemléltethető.
4. A mélységértékek a becslő eljárásból származó értékek, az optical flow mezők a rendelkezésre álló *grond truth*. Ez az összeállítás megmutatja, hogy ha képesek vagyunk nagy pontossággal meghatározni az optikai áramlás mezőt, akkor hogyan befolyásolja az szegmentálást a becslő pontatlansága
5. A mélységértékeket a state-of-the-art színvonalú SIDE eljárásoknak megfelelő, vagy ahhoz közeli hibával terhelt pontos értékek, az optical flow mezők a rendelkezésre álló *ground truth*. Ezzel az esettel szemléltethető, hogy adott esetben a jelenleg alkalmazottak helyett pontosabb, már megvalósított mélységkép-becslő megoldásokkal milyen minőségben valósítható meg a szegmentálás. Ugyanakkor ezeknek az eredményeknek a használhatóságát megkérdőjelezi, hogy a kép egészére vonatkozóan várhatóan nem helyes a nulla középértékű, térben normális eloszlású hiba feltételezése, többek között például azért, mert egy CNN mélységbecslő eljárás várhatóan az egybefüggő szegmenseket illetően hasonló hibával rendelkezik, azaz a szomszédos pixelek esetében ezek a hibák erősen összefügghetnek.

Az 1. összeállítás tulajdonképp a valódi proof-of-concept: ha ennek, az elérhető legjobb vagy azt megközelítő adatokkal dolgozó megoldásnak használható a kimenete, akkor a szegmentálási eljárás működőképes lehet, legfeljebb a megfelelő bemeneti adatok előállításához szükséges futásidő jelenthet problémát. Ez ugyanakkor mind a hardverek, mind az algoritmusok gyorsulásával javulhat a jövőben.

A 2. összeállítás, amely jelen esetben egy működőképes megoldáshoz kellene vezessen, nyilván a legnagyobb mértékkel terhelt hibákkal, zajjal, így lehetséges, hogy a kimenete nem lesz megfelelően használható. Ahhoz, hogy meg lehessen állapítani, hogy a hiba csökkentéséhez elsősorban melyik lépés javítása járul hozzá, a 3-as illetve 4-es lépéseknél az egyik elem az elérhető (leg)jobb adatokat szolgáltató forrásból származik.

A szegmentálás részletei, paraméterei

A szegmentálás vagy maszkolás fentebb leírt elgondolását követve, az alábbi részleteket kell pontosabban kidolgozni:

- Hogyan definiáljuk két optical flow vektor távolságát, hasonlóságát?
- Milyen küszöbértékekkel működnek a lehető legkisebb hibával a fenti összeállítások?
- Milyen struktúráló elemmel érdemes végrehajtani a nyitás és a zárás műveletét a legjobb eredmény eléréséért?

Vektorok távolsága

Az optikai áramlás mezőket MATLAB környezetben célszerűen egy optical flow objektumban tárolhatjuk. Ennek az objektumnak négy azonos méretű tömbje az adott pixelre jellemző x és y irányú sebességkomponenseket (V_x és V_y, a 4. ábra - Pinhole kamera geometriai modellje, kamera-koordinátarendszer (Forrás: [14])án szemléltetett kamera-koordinátarendszer irányainak megfelelően), és az ugyanezen vektort jellemző orientációt és hosszt (Orienation, Magnitude) tartalmaz, ezek közül bármelyik kettő leírja a vektort, azonban a könnyebb értelmezésért célszerű a függőleges és vízszintes komponensek, vagy az orientáció és a nagyság kettősét vizsgálni.

Az előzetes elképzelések alapján az optical flow predikció, amely egy mélységbecslésen alapul, várhatóan nem ad pontos eredményt, de a jármű saját mozgásából becsült sebességvektor jellegre hasonló lehet, mint a valódi: például egyenesvonalú, a kamera principális tengelyével megegyező irányú mozgás esetén a kép bal alsó sarkában található pixelek jellemzően a kép bal és alsó széle felé tartanak, noha a vektor pontos iránya és nagysága a reprezentált objektum pontjának helyzetétől függ. Azonban amennyiben a mozgás iránya ettől jelentősen eltérő, akkor az nagy valószínűséggel valamilyen nem statikus objektum miatt van. Sajnos a közlekedésben tipikusak az olyan helyzetek, amikor egyes objektumok ugyan nem statikusak, de saját, statikus környezetükhöz képest értelmezett mozgásirányuk miatt nagyjából a várható irányban mozdulnak el a képen, ilyen például egy kétsávos út szembeforgalma. Az összehasonlításhoz is mindenképp be kell vonni egy, a vektorok méretét jellemző összetevőt, és emiatt az igen gyakori jelenség miatt várhatóan a ennek a paraméternek a jelentősége sem lesz elhanyagolható.

A részletes tesztelések jobb képet adnak az egyes paraméterek hasznosságáról s optimális értékéről. A két paramétert, jellegükből adódóan, különféleképp érdemes kezelni:

- A „Magnitude” paraméter esetében alapvetően a relatív abszolút eltérést érdemes vizsgálni, hiszen egy nagyobb sebességvektornál adott hosszúságú eltérés egy

hasonlóbb mozgást ír le, mint ha a predikció eleve nagyon kicsi elmozdulást jelzett. Ugyanakkor fontos körülmény, hogy noha a képsíkra vetített pontok pozíciója szubpixeles pontossággal ismert, azok valamely pixelhez rendelése (egyszerű kerekítéssel) további hibát visz be a rendszerbe, így várhatóan a megengedett legkisebb abszolút eltérés nem lehet túl kicsi. Ez alacsonyabb sebességnél, amikor az optikai áramlás vektorok is kicsik, problémát jelenthet.

- A nagyság relatív abszolút eltérésének kiszámítása:

$$\Delta Magn_{rel} = \left| \left(Magn_{pred} - Magn_{act} \right) \Big/ Magn_{pred} \right| \quad (4)$$

Ahol $Magn_{pred}$ az adott pixelhez rendelt optical flow vektor hosszának prediktált, $Magn_{act}$ pedig ugyanezen pixelhez tartozó számított értéke. Amennyiben a predikció értéke 0, annak két oka lehet:

  - A jármű nem mozog: ekkor kiforrottabb és egyszerűbb módszerekkel megvalósítható a mozgásalapú szegmentálás, illetve az SfM végrehajtása sem lehetséges
  - Az adott pont esetében becsült optical flow vektor valóban 0: ekkor, amennyiben a számláló értéke nem zérus, a relatív különbséghez egy olyan nagy értéket rendelünk, amely mindenképp a definiált küszöbérték felett lesz

- Az orientáció eltérésének ($\Delta Orient$) számításakor nincs értelme relatív eltérést számítani, egyszerűen a két orientációérték 0 és π közé eső abszolút különbségét kell meghatározni, az alábbi módon:

$$\Delta Orient = min\left(\left| Orient_{pred} - Orient_{act} \right|, \left(2\pi - \left| Orient_{pred} - Orient_{act} \right| \right) \right) \quad (5)$$

Ahol a korábbiakhoz hasonlóan $Orient_{pred}$ az adott pixelhez rendelt optical flow vektor orientációjának prediktált, $Orient_{act}$ pedig ugyanezen pixelhez tartozó számított értéke.

- Természetesen amennyiben valamelyik megoldásnak nincs érvényes kimenete az adott pixelre, úgy nem 0-val érdemes helyettesíteni az orientációt, hanem egyszerűen ismeretlen régóként megjelölni. Tipikusan ilyenek például az eget jelölő pixelek, szerencsére ezek sokszor nem hordoznak releváns információkat.

Amennyiben a két paraméter esetében definiálunk egy maximális megengedett eltérést $\{\Delta Magn_{rel,max}, \Delta Orient_{max}\}$, a küszöbértékek felhasználására az alábbi lehetőségek nyílnak:

a) A két küszöbérték egymástól független ellenőrzése. Ez a legmegengedőbb változat, amelynek a hasonlóság megítélési képessége megkérdőjelezhető, hiszen nem veszi figyelembe, hogy az egyes dimenziókban mennyivel van a küszöbérték alatt az eltérés.

b) A kiszámított eltérések és a vonatkozó küszöbértékek hányadosát képezzük, így a két paramétert egyforma skálára transzformáljuk.

$$\Delta Magn_{scaled} = {\Delta Magn_{rel}}/{\Delta Magn_{rel,max}} \quad (6)$$

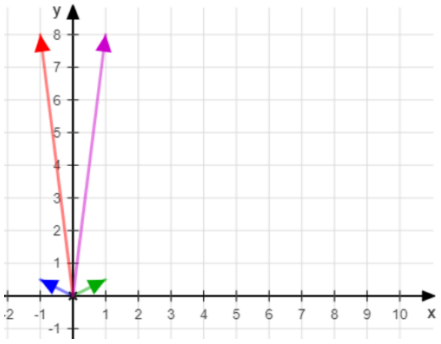
$$\Delta Orient_{scaled} = {\Delta Orient}/{\Delta Orient_{max}} \quad (7)$$

Mindkét mérőszám 0 és 1 közti értéket vesz fel, amennyiben egyik eltérés sem haladta meg az előírt küszöböt. A két eltérést így már könnyedén összevonhatjuk egy Minkowski-távolsággal:

$$d = \left(\Delta Magn_{scaled}^{p} + \Delta Orient_{scaled}^{p}\right)^{1/p} \quad (8)$$

Célszerűen p = 2 választással az euklideszi, vagy p = 1 választással a még gyorsabb számítást lehetővé tévő Manhattan távolságot alkalmazhatjuk. Amennyiben az így kapott távolságérték 1-nél nagyobb, akkor megállapítjuk a két vektor különbözőségét. Ez a független értékelésnél szigorúbb, hiszen nem csak akkor utasítjuk el a hasonlóságot, ha bármelyik érték eléri a meghatározott küszöböt, hanem akkor is, ha a mindkét eltérés a megengedett maximumokhoz képest kellően nagy.

Az orientáció-méret paraméterpáros választásának indoklását a 7. ábra - A paraméterek alkalmasságának összehasonlításaszemlélteti:



7. ábra - A paraméterek alkalmasságának összehasonlítása, az x és y tengelyek mértékegysége pixel

A fenti távolság-megközelítéseket a V_x, illetve V_y sebességösszetevőkre hasonlóan alkalmazva tekintsük az ábra 4 vektorát, páronként a kék-zöld illetve a piros-lila vektorokat. Mindkét esetben az y irányú sebességösszetevő azonos, így nem játszik szerepet a különbözőség meghatározásában. Az X irányú sebességösszetevő a párosok egyik tagjában rendre -1 illetve

1, így a relatív eltérés 200%. Ezek alapján a két vektor különbözősége a két esetben azonos. Ugyanakkor a két hosszabb vektor egy sokkal inkább hasonló mozgást jellemez, mint a két rövidebb- hiszen mindkét esetben azonos nagyságú vektorokról beszélünk, azonban az x összetevő, amely a különbséget okozza, jóval kisebb nagyságú az y-hoz képest az első esetben. A két vektor hasonlóbb voltát az orientációbeli eltérés egyértelműen megmutatja. A relatív hibáknak az adott komponens nagysága szerinti súlyozása megoldást jelenthetne a problémára, azonban ez a plusz számítás egyszerűen elkerülhető az orientáció - nagyság paraméterpáros alkalmazásával.

Strukturáló elem, a szegmentálás javítása

A keletkezett maszkot a korábbiakban leírtak alapján a gyorsaság érdekében első megoldásként egy bináris morfológiai nyitás-zárás művelettel javítjuk. Így mind a dinamikus részekbe ékelődött apró statikusnak érzékelt, mind az alapvetően statikusnak feltételezett szegmensekben található dinamikus szigetek, pontok a domináns szegmensbe olvaszthatók. Mivel a képeken detektálandó mozgó objektumok alakjáról nincs semmilyen előzetesen feltételezés, így célszerű lehet a minden irányban megközelítőleg azonos méretű „*disk*” (kört közelítő alakú) strukturáló elem használata. Ennek mind a nyitás, mind a zárás során alkalmazott mérete az egyéb paraméterekhez hasonlóan az optimalizálás során meghatározandó.

A valódi mélységképek ritka mátrixokként állnak rendelkezésre, így az összehasonlítás eredménye is egy ritka mátrix lesz. Ebből egy olyan maszkot, amely alapján a kép minden pixeléről eldönthető, hogy statikus, avagy dinamikus területhez tartozik-e, szintén egy strukturáló elem és egy zárás művelet segítségével képezhető. A fenti megfontolások alapján itt is a „disk” alakú strukturáló elem alkalmazása a célszerű, azonban itt a méretet a kiindulási mátrixok sűrűsége, azaz az értékkel rendelkező elemek távolsága is erősen befolyásolja. Ha egy adott szegmensben minden érvényes adat valamely adott kategóriába tartozik, akkor egyéb információ híján ezt az értéket rendeljük a többi szomszédos pixelhez is.

A gépi látásban gyakori feltételezés – miszerint az objektumok rendszerint egybefüggők, nem, vagy csak kis mértékben deformálódnak – a jelen alkalmazásban is fennáll. A szemantikus szegmentálás javítása így egy általános szegmentáló eljárással ötvözve a mozgó és statikus objektumok körvonalainak pontosabb lekövetését, ezáltal pedig akár megkönnyített további feldolgozást tesz lehetővé – pl. osztályozás, majd ezt követően a rekonstrukció valamilyen magasabb absztrakciós szintű reprezentációba.

A mozgásalapú szegmentálás kiértékelése

A két optikai áramlás mező összevetése valamely távolságkritériumok alapján egy bináris mátrixot eredményez. Ez alapvetően minden pixelen valamilyen érvényes adattal rendelkezik, mert vagy sűrű bementi mátrixok álltak rendelkezésre, vagy ki lett terjesztve az osztályozás a többi pixelre is. Ugyanakkor a valósi optical flow értékeket tartalmazó adatbázis bizonyos képek esetében jelentős egybefüggő régiókkal rendelkezik, amelyre nem áll rendelkezésre valódi adat. Ennek oka, hogy az adatok előállítása az alábbi módon történt [21].:

- A háttér mozgása a jármű saját mozgásával kompenzálva, a lézerszkenner rögzítette pontok alapján lett meghatározva. A mozgás becslése során mind a GPS-IMU egység adatait, mind a LIDAR pontfelhők adatait használták, képkockánként a szkenner 7 egymás utáni mérését felhasználva. A mozgó objektumokhoz tartozó pontok az adatbázisban rendelkezésre álló, különböző objektumokhoz rendelt határoló-testek (bounding box) segítségével lettek eltávolítva.
- A mozgó objektumok elmozdulását egy 16 különféle jármű háromdimenziós CAD modelljének az egymás utáni képkockákhoz tartozó 3D pontfelhőre való illesztésével reprodukálták, amely folyamat során a nem merev, deformálódó objektumokat tartalmazó képszegmenseket manuálisan maszkolták ki, így ilyen objektumokra nem áll rendelkezésre adat.

Emiatt amikor a generált flow adatok is felhasználásra kerülnek, akkor a kiértékelés csak azokat a pixeleket illetően történik meg, amelyekre van érvényes adat. Az egyéb esetekben, ahol nincsenek nagy, adat nélküli szegmensei a bemeneti mátrixoknak, a kiértékelés a teljes képen történik.

A kapott bináris mátrixot ezután a manuálisan készített maszkkal összevetve az alábbi módon történik a szegmentáció értékelése: minden pixelt 4 kategóriába (TP - true positive, FN – false negative, TN – true negative, FP – false positive, ahol egy TP besorolású pixel egy mozgó objektumot jelöl, és így is lett osztályozva) sorolhatunk, amennyiben rendelkezésre áll érvényes adat. A cél a helytelenül osztályozott (FP és FN) pixelek arányának (FP_{rate} , FN_{rate}) csökkentése.

$$FP_{rate} = {FP}/{(FP + TN)} \qquad FN_{rate} = {FN}/{(FN + TP)} \tag{9}$$

Első optimalizálási célként rendre a pozitív, illetve negatív területek arányával súlyozva vehetjük figyelembe a téves pixeleket, így minimalizálva a tévesen osztályozott egységek teljes számát. Ennek a megoldásnak a hátulütője, hogy a tesztelésre előkészített képeken kijelölt mozgó szegmensek aránya összességében nagyon kicsi – átlagosan kevesebb, mint 5%. Így egyszerűen minden pixelt a statikus környezethez sorolva is igen kedvező ez az arányszám, azonban a feladatot nem oldjuk meg. Épp ezért a választott mérőszám a fals pozitív és fals negatív arányok középértéke:

$$Q = {(FP_{rate} + FN_{rate})}/{2} \tag{10}$$

Ahol Q (quality) az osztályozás minőségét, egyfajta átlagos hibáját jellemzi. A hasonló átlagos minőségű paraméterkombinációk közül ugyanakkor célszerű lehet azt választani, ami összességében kevesebb helytelenül maszkolt pixelt eredményez – azaz jelen esetben jellemzően az $FN_{rate} > FP_{rate}$ egyenlőtlenség áll fenn e paraméterkombinációk közt.

Az optimális paraméterkombináció megkeresésének célja elsősorban az adott megoldás működőképességének felmérése, valamint az egyes feldolgozási lépések hibáinak a végső kimenetre való hatásának vizsgálata. A különböző konfigurációk rendre ugyanazon a képsorozaton kerültek tesztelésre. Mivel a működőképesség megítélése volt a cél, és a valódi működési paraméterek beállításához változatosabb és nagyobb adatmennyiségre lenne szükség, így a pontos minimum megtalálása nem fontos. A cél egy kisebb paraméterrégió kijelölése volt, ahol a legjobb megoldást megközelítő a szegmentálás minősége. Az egyszerű brute-force megoldás, amelynek során a paraméterteret mintavételezve keressük az optimális kombinációt, a célnak megfelelő eredményt hozhat.

A paramétertér mintavételezése az alábbi megfontolások szerint történt:

- Az orientációeltérés legnagyobb értéke π, így előzetesen π/2 vagy kisebb értékre számítottunk az optimum kapcsán (inkább hasonlít, mint nem)
- A méret esetében nincs ilyen természetes korlát, azonban mivel a predikció esetében kiindulási koordináták egész pixelben vannak megadva, így kedvezőtlen esetben ez akár fél pixel közeli kerekítési hibát vihet az optikai áramlásba. Mivel itt is a relatív eltérést került küszöbözésre, a fél pixel nem definiált konkrét alsó limitet, de a várakozások szerint az optimum nem a nagyon kicsi megengedett relatív eltérések esetén lesz elérhető.

- Először rendre a paramétertér minél nagyobb részét tekintjük, kellő számú mintavétellel, hogy a főbb trendek a pontokat egy egyszerű 3 dimenziós koordinátarendszerben ábrázolva áttekinthetők legyenek. A várakozásoknak megfelelően a túl szigorú limitek esetén a fals pozitív, a túl megengedők esetében pedig a fals negatív pixelek aránya lesz igen magas, jelentősen rontva a minőséget jellemző értéket. Egy potenciális optimális közeli pontból a paramétertér tetszőleges irányába indulva kedvezőtlenebb kimenetet kapunk.
- A globális trendek felmérése után az optimum közelében, egy vagy két további iterációban magasabb mintavételi sűrűséggel is kiértékelésre kerül az adott szegmentálási megközelítés a pontosabb, biztosabb eredmény megtalálásáért.

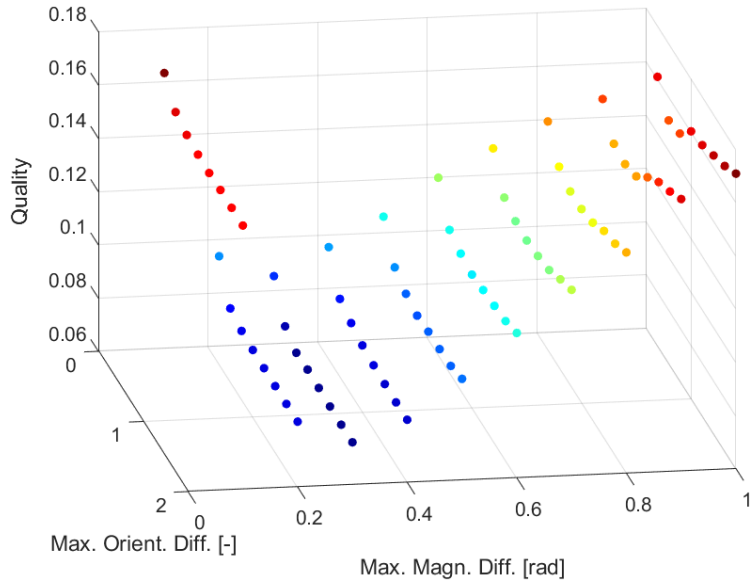
Minden egyes összeállítás a képek egy ugyanazon csoportján lett tesztelve, amelyek változatos vezetési szituációkat jelenítenek meg – úgy mint pl. városi közlekedés, egy és kétsávos utakon való haladás, villamos és kerékpáros mellett/mögött való haladás, osztott pályás úton való haladás, stb. A képek szélein, a kép aktuális méretétől függően egy bizonyos szélességű határ ki lett hagyva a kiértékeléskor, mert a jellemző, előre haladó mozgás során a kép szélei a következő képkockán már nem szerepelnek, így mind az optical flow mezőt számító, mind az SfM eljárások számára tulajdonképp használhatatlanok.

A mozgásalapú szegmentáció hatékonysága, eredmények

1) A pontos értékekkel végzett szegmentálás

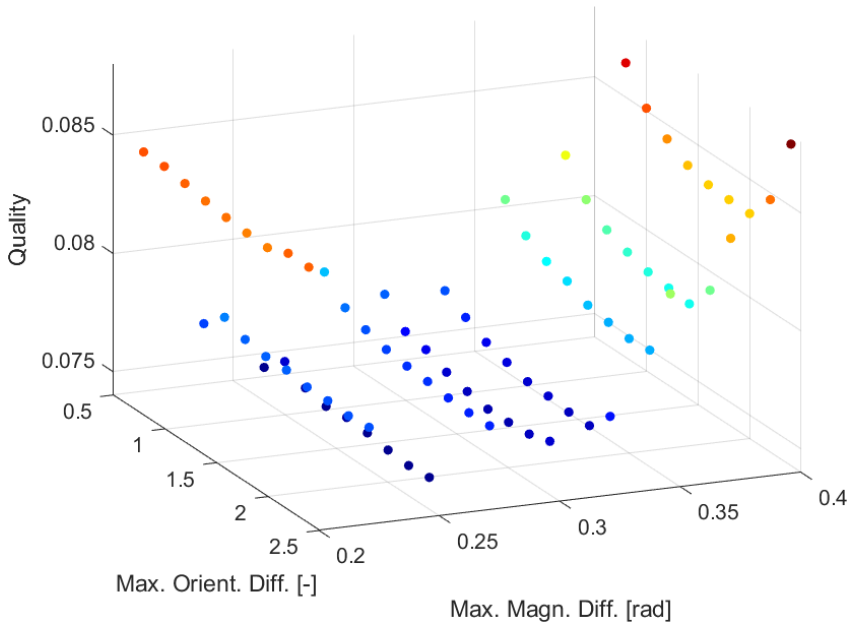
Ebben az összeállításban az adatbázisban elérhető, a szenzorok és a megfelelő adatgenerálások pontosságának megfelelő adatok kerültek felhasználásra, amelyek több, az adatbázishoz kapcsolódó benchmark viszonyítási alapját jelentik. Mivel mind a az optikai áramlást, mind a mélységadatokat leíró mátrixok sok üres elemmel rendelkeznek, ezért a kiértékelést csak azokra a pixelekre végeztem el, amelyekre vonatkozóan mindkét adatforrás tartalmaz érvényes értékeket.

Az optimumkeresés teljes folyamatát csak ebben az esetben veszem végig, a többinél csak az érdekesebb grafikonok kerülnek bemutatásra.

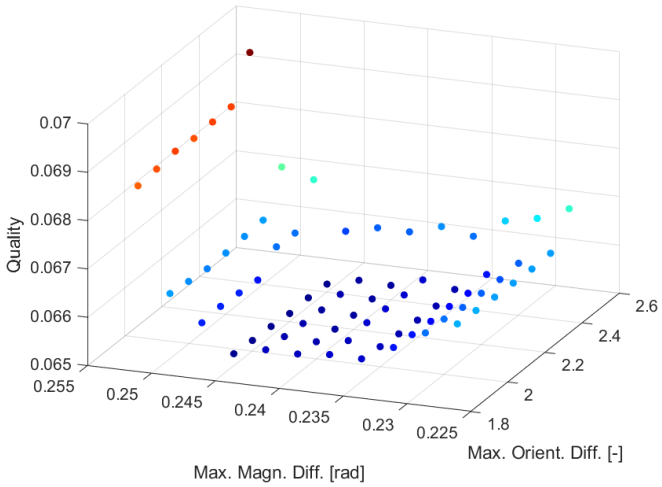


8. ábra - Optimumkeresés, ground truth adatok, 1. iteráció

Az első optimumkeresés rögtön váratlan eredményeket hozott: a vektor mérete sokkal jobb összehasonlítási alapnak mutatkozik, mint az orientáció, amely dimenzió mentén – a nagyon kis értékektől eltekintve – nem mutat jelentős változást a jósági tényező. A 8. ábra - Optimumkeresés, ground truth adatokán kapott kimenet alapján a méretbeli relatív eltérést 0.2 és 0.4 közt sűrűbben kell mintavételezni, míg az orientációeltérést szélesebb intervallumon érdemes vizsgálni.



9. ábra - Optimumkeresés, ground ttruth adatok, 2. iteráció



10. ábra - Optimumkeresés, ground truth adatok, 3. iteráció

A 2. és 3. iterációban (9. és 10. ábrák) az optimum helyét a engedélyezett relatív nagyságbeli eltérés tengelyén 0.225 és 0.25 közé, majd a [0,2435; 0,245] intervallumra szűkítik, míg az orientáció esetében a 2-nél nagyobb értékekere is ki kell terjeszeni a keresést, ahol az [1,8; 2,4] intervallumon mindenhol a 6,6%.-os átlagos hibaarány alatt marad a maszkolás. A művelet tehát a vártnál jóval megengedőbb az orientációeltéréssel szemben, és sokkal érzékenyebb a méret-paraméter változtatására. Ezt a hibaarányt 8 pixel sugarú „disk" strukturáló elem alkalmazásával volt lehetőség elérni.

Az elérhető legjobb bemeneti adatokat felhasználva így a szegmentálás képes arra, hogy átlagosan 6,6%-os hibával osztályozza a kép minden egyes pixelét. Noha ez a hiba viszonylag nagynak tűnhet, az elért fals pozitív és fals negatív ráták mellett ez azt jelenti, hogy

- A kép CrowdMapping szempontjából lényegtelen részeinek kb. 94%-át ki tudjuk maszkolni, ennyivel csökkentve mind az online, mind az offline feldolgozandó adatmennyiséget
- Eközben a kép releváns részeinek mindössze kb. 7%-át veszítjük el a helytelen szegmentálás miatt

Ezek az arányszámok nagy beérkező adatmennyiség esetén azt jelentik, hogy a maszkolás során megtartott releváns régiók várhatóan előbb-utóbb átfedésben lesznek egymással, hiszen a megvilágítástól és az időjárási körülményektől eltekintve egy statikus környezetről van szó. A változó, dinamikus környezetet reprezentáló adatoknak csak egy kis része kerül feldolgozásra, amelyek azonban egy változó környezetet ábrázolnak, így az újabb adatok beérkezésével nem kapnak megerősítést. A kevesebb nem releváns adat a további feldolgozás során megkönnyíti a térképdarabok egymáshoz illesztését, a térkép frissítését.

Alább egy, a kijelölt optimum-régióból származó paraméterpárossal végrehajtott szegmentálás látható, párba állítva a viszonyítási alapnak vett, kézzel szegmentált maszkkal.

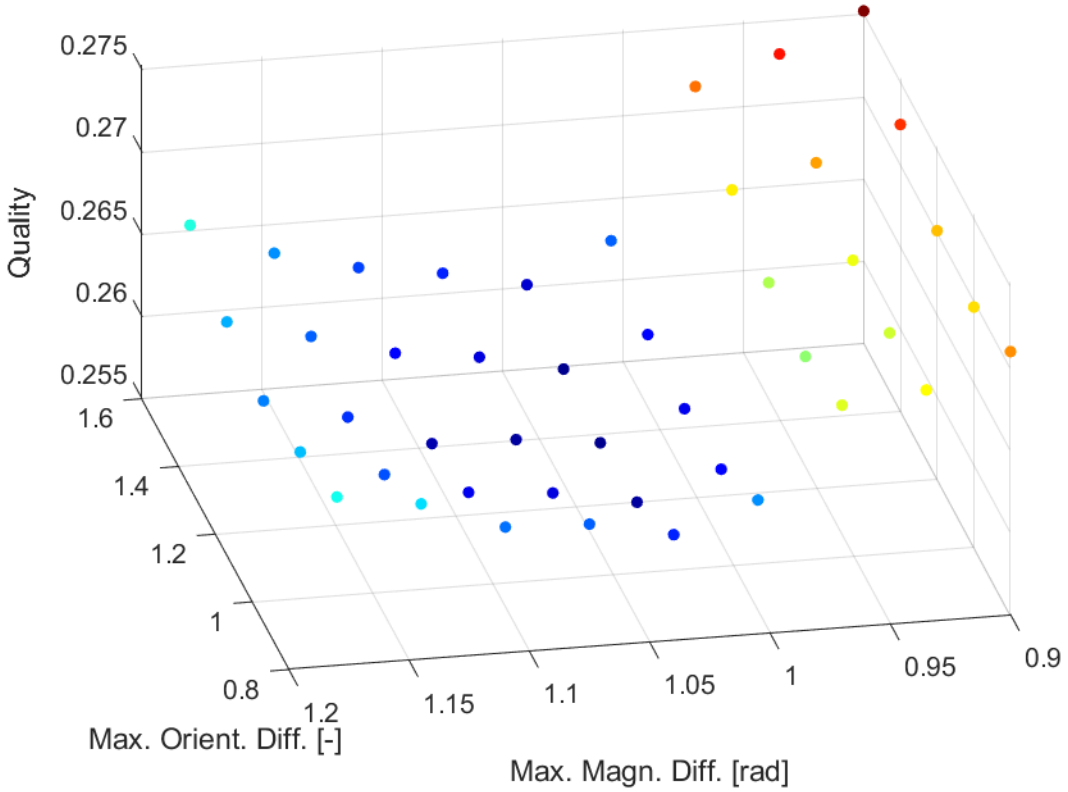
11. ábra - Mozgásalapú szegmentálás, felül a kézzel készített, alul a számított maszk, sárgával jelölve a mozgóként besorolt szegmenseket

A 11. ábrán látható, hogy a kép – kiértékeléskor az egymás utáni képek közti átfedés hiánya miatt figyelmen kívül hagyott – szélein kívül gyakorlatilag minden mozgó besorolást kapott pixel valamilyen valóban mozgó objektumot ábrázoló, vagy ahhoz közel álló képszegmens része, valamint minden nagyobb mozgó objektumot sikerült detektálni. A strukturálás hatásaként ugyan eltűnhetnek egyes kisebb, hibásan besorolt pixelcsoportok, viszont a nagyon kis, valóban a környezetüktől eltérően mozgó szegmensek is beleolvadhatnak a környezetbe. Szintén a bináris morfológiai nyitás-zárás műveletek eredményeként alakulhatnak ki a nagyobb, mozgó szegmensek közt helytelenül osztályozott „összeköttetések", mint a az ábrán a kép felső részéről benyúló nyúlványok. Ugyanakkor a strukturáló elem méretének kis mértékű változtatása nem hozott érdemi változást, az is inkább kedvezőtlen volt. Nagyobb mértékű méretnövelés már nehezen lett volna indokolható a képek jellemző struktúrája miatt (mekkora objektumok fordulnak elő), kisebb méretek esetén pedig a ritkán mintavételezett képen feldolgozott pixelek régiókká való összeolvasztása nem volt lehetséges.

2) Szegmentálás becsült és számított értékekkel

Ez a szegmentálás célzott megvalósítása, ahol valóban csak valós időben, a kamera és a lokalizációs szenzorok bemeneteit használva törekszünk a mozgó objektumokat ábrázoló képrészek kimaszkolására. Jelen összeállításban nagyon jelentős minőségromlást jelent ez a pontos, mért adatokkal való szegmentáláshoz képest, valamivel 26% alatti hibaaránnyal. Ekkor ugyan még mindig jelentős mennyiségű irreleváns adattól szabadulhatunk meg, ugyanakkor a kép jelentős használható területét dobjuk el. Ráadásul a tesztképek nagyjából 5%-os dinamikus

képszegmens aránya mellett a 26%-os hiba azt jelenti, hogy az kimaszkolt képrészek túlnyomó része a statikus részhez tartozik. A kimenetet a 12. ábra mutatja.



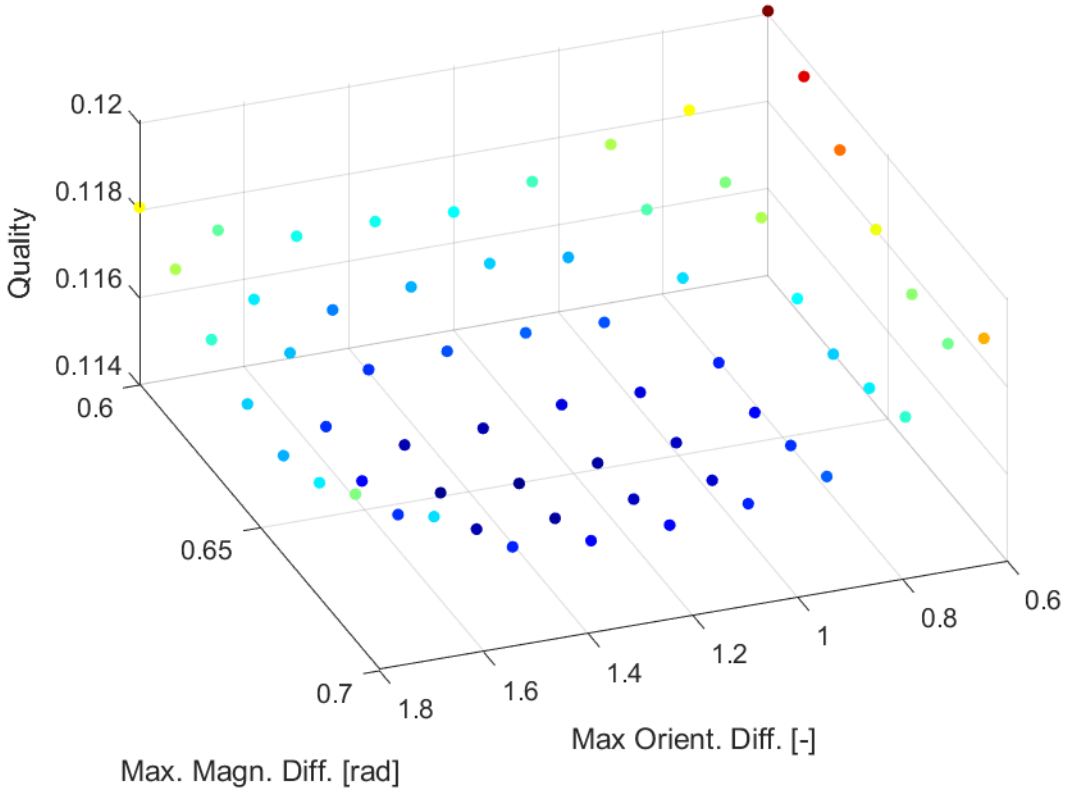
12. ábra - Optimumkeresés, becsült mélység és optikai áramlás adatok mellett

Az optimumot megközelítő kimenet biztosításához a megengedett nagyságeltérést a [1,005; 1,11], az orientációeltérést pedig a [1; 1,4] intervallumról választva kaphatjuk. A mért adatokkal történő szegmentálás esetével ellentétben, mind itt, mind a továbbiakban 4 pixel sugarú „disk" strukturáló elemmel lehetett a legkedvezőbb eredményeket kapni, ami vélhetően a fele méretre történő átskálázás következménye.

3) Szegmentálás pontos optical flow és becsült mélységadatok felhasználásával

Az adott mélységbecslési eljárás melletti legjobb eredményeket akkor kaphatjuk, amennyiben az optikai áramlás számítását a lehető legkisebb hibával tudjuk megvalósítani. Ennek a forgatókönyvnek a szimulálására az optikai áramlás számítása az adatbázisban elérhető, pontos értékeket tartalmazó optikai áramlás mezőkkel lett kiváltva. Ezeket a mélységbecslő eljárásnak megfelelően végrehajtott átméretezés – és az optical flow vektorok esetében átskálázás – után lehet összevetni a predikcióval. Az optimális közeli régiót az első, viszonyítási alapként szolgáló összeállításhoz képest nagyobb megengedett nagyságeltérés és kisebb megengedett orientációeltérés jellemzi. A minőség jelentősen romlik ugyan, de a 11,5%-os hiba még mindig azt jelenti, hogy kevés releváns adatot (11-12%) veszítünk el, miközben a hibás és felesleges

adatok jelentős részét (88-89%) a további feldolgozás előtt kiszűrjük. Az optimumkeresés utolsó iterációjának kimenetét a 14. ábra mutatja.

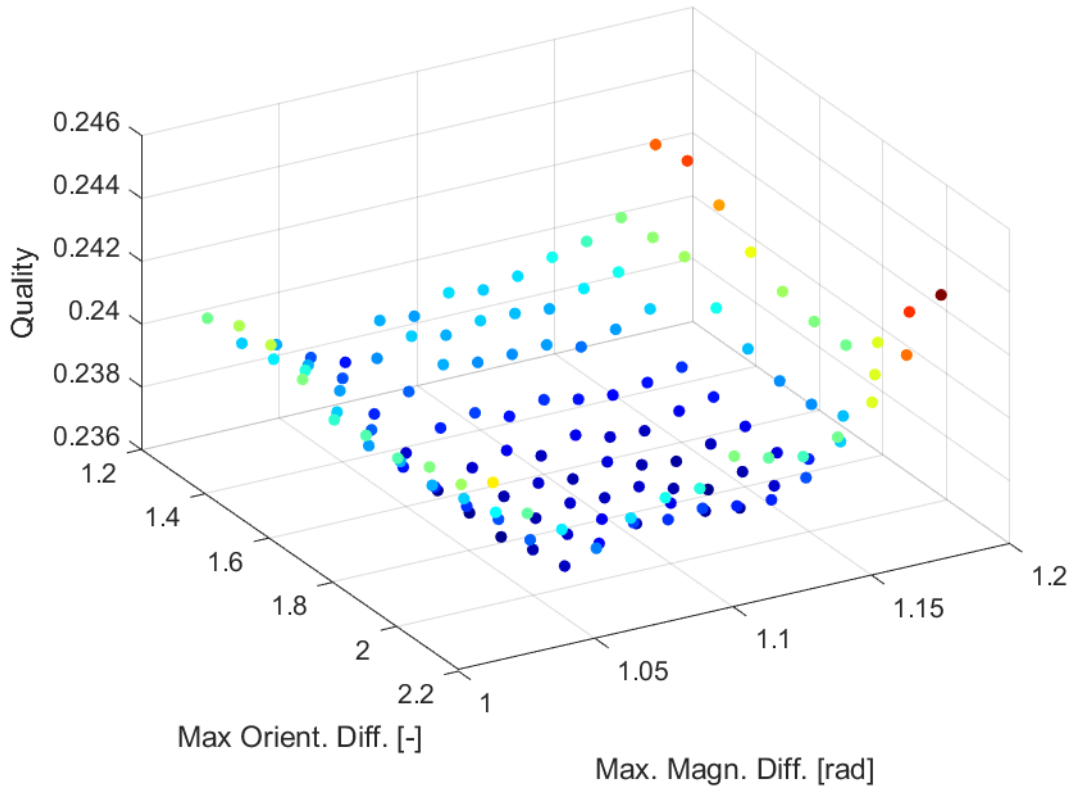


13. ábra - Optimumkeresés, becsült mélység és ground truth optikai áramlás adatok mellett

Az kimenet alapján az elért legjobb maszkolást a megengedett orientációeltérés-nagyságeltérés paraméterek [1,05; 1,5] -[0,645; 0,675] intervallumról történő kiválasztása esetén kaphatjuk.

4) Szegmentálás pontos mélység és számított optical flow adatok felhasználásával

Hasonlóan a 3-as ponthoz, itt az optikai áramlás meghatározására felhasznált eljárás szegmentálása gyakorolt hatását izoláljuk a becsült helyett a mért mélységinformáció felhasználásával. Az elérhető minőség így nagyon jelentősen leromlik, nagyrészt tehát ez a lépés felelős a végső szegmentáció jóságának leromlásáért a 2-es összeállításban. A legjobb kimenet a megengedett orientációeltérés-nagyságeltérés [1,7; 2] - [1,05; 1,1375] intervallumokról választott küszöbértékekkel érhető el, azonban a hiba átlagosan itt is 23,8%, ami csak kis mértékben jobb, mint a becsült mélységértékekkel dolgozó variáns.



14. ábra - Optimumkeresés, mért mélység és becsült optikai áramlás adatok mellett

Összefoglalás

Az egyes bemeneti adatokhoz tartozóan megkeresett alkalmas paramétertartományokat, illetve a szegmentálások minőségét a 2. Táblázat foglalja össze. Fontos megjegyezni, hogy a paramétertartományok nem az adott legkisebb hibától való adott abszolút vagy relatív eltérésen belül eső paraméterkombinációkat jelölik ki, csupán a mintavételezésnek megfelelően egy régiót, amelyen belül található a minták minimuma, valamint amelynek határai felé haladva tetszőleges paraméter mentén nagyjából hasonló mértékben változik a hiba. Így az elérhető legkisebb hiba nagyságán kívül a paraméterintervallumok paramétertérben való elhelyezkedése, illetve az egyes intervallumok dimenzióinak egymáshoz viszonyított arányai lehetnek érdemesek az összehasonlításra.

2. Táblázat - A szegmentálás jellemzői különböző bemeneti adatokkal

Mélységadat	Ground truth	CNN	Ground truth	CNN
Optikai áramlás	Ground truth	Ground truth	Farneback	Farneback
Disk radius [pixel]	8	4	4	4
Max. megengedett orientációeltérés [rad]	[1,8; 2,4]	[1,05; 1,5]	[1,7; 2]	[1; 1,4]
Max. megengedett relatív nagyságbeli eltérés [pixel]	[0,235; 0,245]	[0,645; 0,675]	[1,05; 1,1375]	[1,005; 1,11]
Minőség	6,6%	11,5%	23,8%	25,9%

A várakozásoknak megfelelően a pontosabb adatok bevonásával precízebbé vált a szegmentálás. A két kérdéses bemenet közül az optikai áramlás sokkal nagyobb hatást jelentett a szegmentálás minőségének szempontjából, ami vélhetőleg a Farneback-eljárás adott paraméterek melletti hibáinak is köszönhető.

A megfelelő paraméterintervallumok elhelyezkedésénét illetően a megengedett relatív legnagyobb méretbeli eltérés a rosszabb adatok bevonásával mindig nőtt, a legjobban a valódiról a számított optikai áramlásadatokra való áttéréskor. A megengedett legnagyobb orientációeltérés ezzel szemben inkább a kisebb értékek felé tolódott el, elsősorban a pontatlanabb, becsült mélységadatokra való áttéréskor.

Az intervallumok dimenzióinak arányai drasztikusan nem változnak, de a legszélsőségesebb viszonyt a mért, pontos adatok esetén figyelhetjük meg, ekkor a legnagyobb a hiba érzékenységének különbsége a két paraméter esetén. Mindkét eltolódás azt jelzi, hogy rosszabb, zajosabb bemeneti adatok esetén az orientáció jelentősége megnő a hasonlóság megállapításában – relatíve szigorúbb feltételt jelent a kiindulási esethez képest.

5) A maszkolás hatása a háromdimenziós rekonstrukcióra

A mozgásalapú szegmentálás a különböző vezetéstámogató funkciók megvalósításában - például más közlekedők észlelése, ütközéselkerülés – önmagában is hasznos, azonban a CrowdMapping projekten belül elsősorban a háromdimenziós rekonstrukcióra kifejtett hatásának vizsgálata volt a kitűzött feladat. E hatások többféleképp jelentkezhetnek:

- A feleslegesen feldolgozott és továbbított adatmennyiség csökkentése: mind lokálisan, mind a felhőben végzett műveletek során feleslegesen számítanánk ki a mozgó objektumokhoz tartozó pontokat, hiszen ezek a környezet olyan elemeit jellemzik, amelyeket nem szeretnénk eltárolni egy térképen.
- A feldolgozási idő rövidülése: azonos sűrűségű rekonstrukció mellett, csak a releváns régiókra koncentrálva kevesebb pontot kell megkeresi, leírni, párosítani és rekonstruálni, amely feldolgozási idő nyereség kompenzálhatja a korábbi, szegmentáláshoz szükséges műveleteket
- A rekonstrukció átlagos pontosságának, megbízhatóságának javulása: az olyan pontok egy jelentős részének kiesésével, amelyek mozgó objektumhoz tartoznak, és így a különböző időpillanatokban készített képeken különböző pozíciót foglaltak el a

referencia-koordinátarendszerben, várhatóan csökken a nagy hibával rekonstruált pontok aránya, nő az eljárás átlagos pontossága.

A következőkben röviden kitérek az SfM eljárás lépéseire, sajátosságaira, illetve bemutatom a szegmentálás rekonstrukcióra gyakorolt hatásának vizsgálatát.

A rekonstrukciós eljárás rövid leírása

Az eljárás egy klasszikus Strucutre from Motion végrehajtást követő rekonstrukció, némi egyszerűsítésekkel, nagyon hasonlóan a szemléletes [25] példához.

- Egymás utáni képeket kap bemenetként, midig 3 képkocka kerül feldolgozásra, mivel ugyan a több kép nagyobb pontosságot eredményezhet, azonban jelentősen növeli a feldolgozási időt, amit a valósidejű esetben az új képek beérkezése is korlátoz
- A képeken jellemzőket keresünk. Ezek lehetnek kevés, de nagy biztonsággal beazonosítható elem, amelyet jellemzően a kamerapozíciók becslésére használnak, azonban itt ennek a lépésnek a végrehajtására nincs szükség, hiszen a kamerapozíciók ismertek. A sűrű rekonstrukciónál nagyszámú képjellemző kinyerése és lokalizációja történik meg, rendre az egymás utáni képeken, például a minimum eigenvalue leíró segítségével.
- A pontokat beazonosítjuk leíróik segítségével az egymás utáni képeken, figyelembevéve azok elhelyezkedését is. Ezzel a képkockákon átívelő pályákat kapunk, amelyek megteremtik a kapcsolatot az összerendelt képjellemzők közt.
- A kapott képjellemző-pályák, ismert kamerapózok és kamerakalibráció ezután lehetővé teszi, hogy háromszögeléssel meghatározzuk a képjellemzőkre leképezett háromdimenziós pontok helyzetét a referencia-koordinátarendszerben. Ez jelen esetben mindig az első kamera helyzete, hiszen ennek a képére készült el a maszk.
- A kapott 3D-s pontok pozícióját és a nem fix kameraparamétereket a csoportigazítás során kísérelhetjük meg iteratív módon addig módosítani, amíg el nem érjük a megkövetelt tulajdonságokat. A felhasznált MATLAB függvénykönyvtár megfelelő függvénye a Levenberg-Marqardt algoritmust alkalmazza az adatok további finomítására.

A kimenetet a háromdimenziós pontfelhő jelenti, amelynek pontosságát a pontokat a képsíkra vetítve, majd azt az eredeti, képkockához tartozó mélységképpel összevetve ellenőrizhetjük. Érdemes megjegyezni, hogy a rekonstrukciós eljárás a – rendkívül gyakran előforduló – egyenes vonalú haladás folyamán a kép egy bizonyos részén, amely a jármű haladási irányába

eső térrészt ábrázolja, az egymás utáni képkockákat közti kis változások miatt meglehetősen pontatlan eredményt szolgáltat. Ugyanakkor a közelebbi, illetve a mozgás irányától markánsan eltérő irányokba eső részletek esetén a rekonstrukció feltételei kedvezőbbek.

A szegmentáció hatásának vizsgálata

A mozgásalapú maszkolás hatásának vizsgálata a korábbi lépésben felhasznált képeken végrehajtva történik. Először a teljes képen – a széleket elhagyva a mozgás miatt eltűnő képrészek csökkentése érdekében – keresünk jellemzőket, és így hajtjuk végre a rekonstrukciót. A kapott rekonstrukció minőségét jellemzi a visszavetítési hibák értéke, valamint a kapott pontfelhőt a megfelelő képsíkra vetítve összevethetjük azt az elérhető *ground truth* mélységképekkel. Ezt követően ugyanezen képekre kiszámoljuk az eljárásnak megfelelően beállított paraméterekkel a mozgásalapú maszkot, és csak a statikus régióra eső pontokra hajtjuk végre a rekonstrukció további lépéseit. Kérdés, hogyan változik a csoportigazítás előtti visszavetítési hiba, a mélységképek közti hiba, a rekonstrukció további lépéseihez szükséges futási idő, valamint a feldolgozott, továbbítandó adatmennyiség. A szegmentálás után visszavetített pontokat vizuálisan vizsgálva is könnyen ellenőrizhető, vajon valóban a megfelelő szegmenseket illetően korlátoztuk-e a rekonstrukciót.

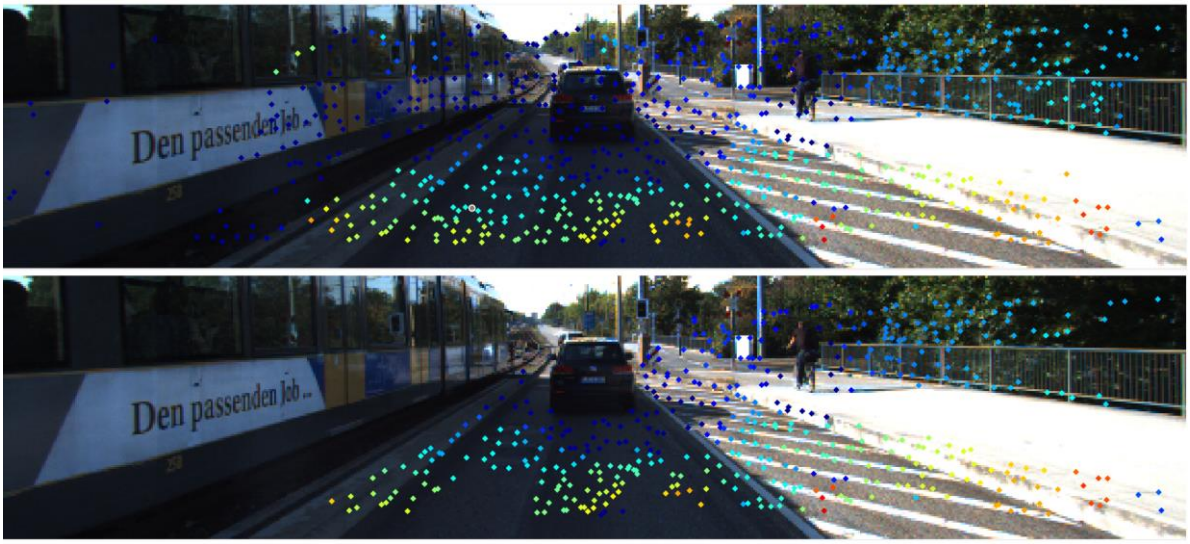
A legkisebb hibaaránnyal – a várakozásoknak megfelelően – a pontos, mért adatokkal dolgozó eljárással volt megvalósítható a szegmentálás. Így a potenciális hatások felméréséhez így ezt a módszert alkalmazva, a megkeresett optimális paraméterintervallumokból vett értékekkel határozzuk meg a kép statikus-dinamikus részeit elkülönítő maszkot. A fentiekben leírtak szerint ezután az SfM eljárást ugyanazon képeken a statikus részekre korlátozva, illetve anélkül is lefuttatjuk, majd megvizsgáljuk a fentebb említett paraméterek változását.

A tesztelést a korábban felhasznált képek egy részén lett végrehajtva, a képeket úgy válogatva, hogy azok közt ne legyen sok nagyon hasonló képszekvencia. Az eredményeket a 16. ábra és a 3. táblázat szemlélteti.

A képen látható, hogy a mozgó, és így a CrowdMapping szempontjából érdektelen, a rekonstrukciót torzító autóhoz, villamoshoz és kerékpároshoz tartozó képjellemzők elvetésével sok pont rekonstrukcióját, követését megspóroljuk, valamint ezek a nem releváns adatok nem kerülnek továbbításra a távoli részbe sem. Az összesen követet háromdimenziós pont a maszkolás hatására átlagosan 35%-kal változott, a visszaesés 20 és 57% közti értéket vett fel. Ez jelentősen nagyobb arány, mint a képeken jellemzően kimaszkolt kb. 15 százaléknyi térrész, ami több tényezőnek is betudható.

Egyrészt a mozgó objektumok jellemzően nem tartoznak azokhoz a nagy, homogén felületekhez, amelyekből nehéz képjellemzőket extrahálni, mint amilyen a 16. ábrán is megfigyelhető járda. Másrészt az objektumhatárokon jellemzően sok követhető képjellemző található, de mint a korábbi példából is kitűnt, a mozgó objektumok környezete is sokszor bekerül ebbe a dinamikusként besorolt szegmensbe. Vagyis a kimaszkolt szegmenseken belül potenciálisan nagyobb sűrűséggel fordulhatnak elő a megtalált képjellemzők. Valamint, mivel a képjellemzőket csak a kép széleitől valamivel beljebb keressük a haladás miatt, ezért a kép belső részein a tesztelt képek esetében lehet, hogy magasabb volt a mozgó objektumok elfoglalta térrészek aránya.

Az okoktól függetlenül ugyanakkor a továbbított adatmennyiség nagyjából egyharmadával való csökkenése elsősorban a korlátozott számítási kapacitással bíró lokális részben bír nagy jelentőséggel. Ezzel párhuzamosan, és elsősorban ennek betudhatóan a további feldolgozási idő is jelentős, nagyjából 20%-os visszaesést mutatott, ami a nem valósidejű környezetben történő tesztelés miatt ugyan további vizsgálatokkal alátámasztandó, de egyértelműen kirajzolódó, kedvező tendencia. Ennek jelentősége azért is nagy, mert a bináris maszk felskálázása egy egyszerű, kis számítási költségű művelet. Így a további munka során felmérhető, mennyivel kisebb felbontáson teljesít még jól a szegmentálás, potenciálisan annyira felgyorsítva azt, hogy az extra műveletekhez szükséges számítási idő a későbbi nyereségekkel kompenzálva elhanyagolható teljes futtatási időnövekményt jelentsen.



15. ábra - A képsíkra visszavetített, rekonstruált pontfelhő, mélység szerint színkódolva

3. Táblázat - A maszkolás hatása a rekonstrukcióra

Megengedett orientációeltérés [rad]	Megengedett relatív hosszeltérés [-]	3D pontok számának változása	Futási idő változása	Visszavetítési hibák változása	A rekonstrukció RMSE-jának változása
2,2	0,25	-35%	-21%	-7,6%	-0,05%

A további vizsgált paraméterekben kismértékű – a pixelben mért visszavetítési hibák a meghagyott képjellemzők esetében 7,6%-kal kisebbek voltak – vagy jelentéktelen javulás volt megfigyelhető, mint a rekonstrukciós hibák esetében. Előbbi kis mértékben csökkentheti a csoportigazításhoz szükséges időt, de várhatóan nem jelentősen. Utóbbi esetében a rekonstrukció általánosan kedvezőtlen mivolta - a kiugró értékek kiszűrése után is jellemzően 20% feletti volt a mélységképek közti eltérés – is okozhatja, hogy gyakorlatilag nem volt érdemi változás, ahogyan a teljes képhez hasonlítva rendkívül kevés összehasonlításra alkalmas pixel is torzíthatta az eredményeket – ugyanis az összehasonlításhoz arra volt szükség, hogy mindkét ritka mélységkép azonos koordinátájú pontjai rendelkezzenek érvényes hozzárendelt mélységinformációval. A további vizsgálódások során ezért mindenképp érdekes lehet más, az jellemzően előforduló helyzeteket potenciálisan nagyobb pontossággal kezelő monokuláris SfM vagy vSLAM eljárásokkal történő tesztelés, majd megvalósítás.

Konklúzió, a munka folytatása

A dolgozat során megvalósításra került az optical flow mező predikciója egy mélységbecslő konvolúciós neurális háló kimenetének felhasználásával, valamint az ezen becslés segítségével a mozgás alapú szegmentálásra tett javaslat is kidolgozásra került. A pontos adatokkal dolgozó szegmentálás – tökéletlen viszonyítási alap mellett is – akár 6,6%-os átlagos hibával is képes volt osztályozni a teszt képsorozat pixeleit, és ez az eredmény a CNN mélységbecslő bevonásával sem romlott drasztikus mértékben, ekkor is 12%-os hibaarány alatt maradva. A mozgás alapú szegmentálásnak kifejezetten az autonóm járművek esetén számos alkalmazása lehet, így a működőképesség igazolása után további, nagyobb és még változatosabb adathalmazon való tesztelés és finomítás lehet indokolt. Fontos felmérni, hogy a jelenleg alkalmazottnál jobb, de szintén megfelelően kis számításigényű optical flow számító algoritmus bevonásával, a mélységbecslő újabb iterációinak, vagy egy jobb becslőnek az alkalmazásával hogyan változik a szegmentálás jósága. Emellett érdekes lehet megvizsgálni, hogy egyéb szegmentálási eljárások bevonásával valósítható meg az objektumok pontosabb elhatárolása, és mindez mekkora plusz erőforrásigényt jelent.

A rekonstrukció során elsősorban a feldolgozott adatmennyiség csökken, azonban itt fontos megjegyezni, hogy ezek az elvesztett adatok jelentős arányban tartalmaznak nem releváns, a további feldolgozás során valamiképp egyébként is kiszűrendő információt. A megtartott pontok a vártnak megfelelően általánosan pontosabban lettek rekonstruálva, így akár a további csoportigazítást kiváltva, vagy legalábbis azt jobb bemeneti adatokkal segítve. A teljes megoldás működőképességének felméréséhez további, a jelenleginél kedvezőbb tulajdonságokat egykamerás rekonstrukciós eljárások integrálása és tesztelése is célszerű lehet, a kimeneti potenciális javulásának jobb felmérésére.

A fentiek mellett a valósidejű környezetben történő implementáció és az így elérhető futási idő felmérése is releváns, hiszen, noha a távoli rész szempontjából kedvező, ha a nagy beérkező adatmennyiség kevesebb hibás vagy felesleges adatpontot tartalmaz, a lokális rész számítási kapacitása sokszor igen korlátozott. További kérdés, hogy elsőszámú környezetérzékelési eljárásként egy ilyen monokuláris eljárást alkalmazva hogyan lehetséges a detektált, mozgó objektumok jobb kezelése, pozíciójuk, méreteik, mozgásuk pontos felmérése.

Köszönetnyilvánítás

Szeretném megköszönni a dolgozat készítése során nyújtott támogatását a konzulensemnek, Szántó Mátyásnak, illetve a dolgozat elkészítéséhez nélkülözhetetlen hozzájárulását Tass Benedeknek, aki a CrowdMapping projekt keretein belül a felhasznált mélységkép-becslőt készítette. A dolgozat az EFOP-3.6.1-16-2016-00014, „Diszruptív technológiák kutatás-fejlesztése az e-mobility területén és integrálásuk a mérnökképzésbe” pályázat támogatásával készült.

Irodalomjegyzék

1. Bengler, Klaus; Dietmayer, Klaus; Farber, Berthold; Maurer, Markus; Stiller, Christoph; Winner, Hermann (2014): Three Decades of Driver Assistance Systems: Review and Future Perspectives. In *IEEE Intell. Transport. Syst. Mag.* 6 (4), pp. 6–22. DOI: 10.1109/MITS.2014.2336271.

2. Maurer, Markus; Gerdes, J. Christian; Lenz, Barbara; Winner, Hermann (2016): Autonomous Driving. Berlin, Heidelberg: Springer Berlin Heidelberg.

3. Nothdurft, Tobias; Hecker, Peter; Ohl, Sebastian; Saust, Falko; Maurer, Markus; Reschka, Andreas; Bohmer, Jurgen Rudiger (2011. 10. 2011. 10. 07): Stadtpilot: First fully autonomous test drives in urban traffic. In : 2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC). 2011 14th International IEEE Conference on Intelligent Transportation Systems - (ITSC 2011). Washington, DC, USA, 2011. 10. 05. - 2011. 10. 07: IEEE, pp. 919–924.

4. Tamás Mészégető 2018,2019: Önálló Laboratórium 1-2.

5. Tomasi, Carlo; Kanade, Takeo (1992): Shape and motion from image streams under orthography: a factorization method. In *Int J Comput Vision* 9 (2), pp. 137–154. DOI: 10.1007/BF00129684.

6. Song, Shiyu; Chandraker, Manmohan; Guest, Clark C. (2016): High Accuracy Monocular SFM and Scale Correction for Autonomous Driving. In *IEEE transactions on pattern analysis and machine intelligence* 38 (4), pp. 730–743. DOI: 10.1109/TPAMI.2015.2469274.

7. Nyimbili, Penjani & Demirel, H. & Seker, Dursun & Erden, Turan (2016): Structure from Motion (SfM) - Approaches and Applications. International Scientific Conference On Applied Sciences

8. Kuang, Hailan; Zhang, Kaiwei; Li, Ruifang; Liu, Xinhua (2018. 02. 2018. 02. 11): Monocular SLAM Algorithm Based on Improved Depth Map Estimation and Keyframe Selection. In : 2018 10th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA). 2018 10th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA). Changsha, 2018. 02. 10. - 2018. 02. 11: IEEE, pp. 350–353.

9. Hasan, Mohamed; Abdellatif, Mohamed (2012): Monocular Depth from Motion Using a New Closed-Form Solution. In David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell et al. (Eds.): Intelligent Robotics and Applications, vol. 7508. Berlin, Heidelberg: Springer Berlin Heidelberg (Lecture Notes in Computer Science), pp. 473–483.

10. Newcombe, Richard A.; Davison, Andrew J. (2010. 06. 13. - 2010. 06. 18): Live dense reconstruction with a single moving camera. In : 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). San Francisco, CA, USA, 2010. 06. 13. - 2010. 06. 18: IEEE, pp. 1498–1505.

11. Suhr Jae, Bae Kwanghyuk, Kim Jaihie, Jung Ho. (2007). Free Parking Space Detection Using Optical Flow-based Euclidean 3D Reconstruction.. Proceedings of the IAPR Conference on Machine Vision Application, p. 563-566.

12. Mitiche, Amar; Sekkati, Hicham (2006): Optical flow 3D segmentation and interpretation: a variational method with active curve evolution and level sets. In *IEEE transactions on pattern analysis and machine intelligence* 28 (11), pp. 1818–1829. DOI: 10.1109/TPAMI.2006.232.

13. Saxena, J. Schulte, and A. Y. Ng (2007): Depth estimation using monocular and stereo cues. International Joint Conference on Artificial Intelligence
14. Márton Szemenyei (2019): Számítógépes Látórendszerek jegyzet, Budapesti Műszaki és Gazdaságtudományi Egyetem, Villamosmérnöki és Informatikai Kar, Irányítástechnika és Informatika Tanszék
15. Klappstein, Jens; Vaudrey, Tobi; Rabe, Clemens; Wedel, Andreas; Klette, Reinhard (2009): Moving Object Segmentation Using Optical Flow and Depth Information. In Toshikazu Wada, Fay Huang, Stephen Lin (Eds.): Advances in Image and Video Technology, vol. 5414. Berlin, Heidelberg: Springer Berlin Heidelberg (Lecture Notes in Computer Science), pp. 611–623
16. Haoyu Ren, Mostafa El-khamy, Jungwon Lee (2019): Deep Robust Single Image Depth Estimation Neural Network Using Scene Understanding
17. Benedek Tass (2019): Mozgás alapú sztereóképek generálása gépi tanulás használatával
18. Uhrig, Jonas; Schneider, Nick; Schneider, Lukas; Franke, Uwe; Brox, Thomas; Geiger, Andreas (2017. 10. 2017. 10. 12): Sparsity Invariant CNNs. In : 2017 International Conference on 3D Vision (3DV). 2017 International Conference on 3D Vision (3DV). Qingdao, 2017. 10. 10. - 2017. 10. 12: IEEE, pp. 11–20
19. Geiger, A.; Lenz, P.; Urtasun, R. (2012. 06. 16. - 2012. 06. 21): Are we ready for autonomous driving? The KITTI vision benchmark suite. In : 2012 IEEE Conference on Computer Vision and Pattern Recognition. 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Providence, RI, 2012. 06. 16. - 2012. 06. 21: IEEE, pp. 3354–3361.
20. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. (2013): Vision meets robotics: The KITTI dataset. In *The International Journal of Robotics Research* 32 (11), pp. 1231–1237. DOI: 10.1177/0278364913491297.
21. Menze, Moritz; Geiger, Andreas (2015. 06. 2015. 06. 12): Object scene flow for autonomous vehicles. In : 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Boston, MA, USA, 2015. 06. 07. - 2015. 06. 12: IEEE, pp. 3061–3070.
22. A fejlesztőkészletek a http://www.cvlibs.net/datasets/kitti/index.php webcímen elérhetőek
23. Béla Lantos (1991), Robotok irányítása, Akadémiai Kiadó, Budapest
24. A fehasznált kódrészletek a https://www.mathworks.com/help/vision/examples/structure-from-motion-from-multiple-views.html webcímen elérhetőek
25. Mátyás, Szántó (2019): Kutatási összefoglaló, Komplex vizsga

Ábrák jegyzéke