Kamera-LIDAR pár kalibrálása dobozokkal

Pusztai Zoltán[1], Hajder Levente[2]

[1] MTA SZTAKI, Eötvös Loránd Tudományegyetem
zoltan.pusztai@mta.sztaki.hu
[2] MTA SZTAKI
levente.hajder@mta.sztaki.hu

Kivonat Manapság egyre nagyobb az igény a minket körülvevő környezet felderítésére, szaknyelven szólva letapogatására (szkennelésére). Különösen igaz ez a robotok és önvezető autók világában, amelyeknek önmagukban kell döntéseket hozniuk a lehető legkevesebb emberi beavatkozással. Több eszköz is létezik, amelyekkel gépeinket felszerelve azok képesek lesznek a világ érzékelésére: kamerák, mikrofonok, radarok, stb. Az egyik ilyen gyakran használt eszköz a LiDAR (Light Detection And Ranging), amellyel ritka pontfelhőt készíthetünk a körülöttünk található világról. A LIDAR-os méréseket kiegészíthetik az RGB kamerák képeit feldolgozó algoritmusok, melyek például segíthetnek a kapott pontfelhő sűrítésében. Autonóm járművek és robotok esetében ezen eszközök fixálhatóak egymáshoz képest, ezért az egymáshoz képesti pozíciót és elforgatást kalibráció segítségével a működés előtt ki lehet számolni. Ebben a cikkben megmutatjuk, hogy ez a kalibráció megoldható dobozok segítségével. A javasolt eljárást részletesen bemutatjuk, a pontosságot elemezzük.

1.. Bevezetés

Manapság egyre nagyobb az igény a minket körülvevő környezet felderítésére, szaknyelven szólva letapogatására (szkennelésére). Különösen igaz ez a robotok és önvezető autók világában, amelyeknek önmagukban kell döntéseket hozniuk, a lehető legkevesebb emberi beavatkozással. Több eszköz is létezik, amelyekkel gépeinket felszerelve azok képesek lesznek a világ érzékelésére: kamerák, mikrofonok, radarok, stb. Az egyik ilyen gyakran használt eszköz a LiDAR (Light Detection And Ranging), amellyel ritka pontfelhőt készíthetünk a körülöttünk található világról. A LiDAR-os méréseket kiegészíthetik az RGB kamerák képeit feldolgozó algoritmusok, melyek például segíthetnek a kapott pontfelhő sűrítésében. Autonóm járművek és robotok esetében ezen eszközök fixálhatóak egymáshoz képest, ezért az egymáshoz képesti pozíciót és elforgatást kalibráció segítségével ki lehet számolni. Ebben a cikkben megmutatjuk, hogy ez a kalibráció hatékonyan megoldható dobozok segítségével. A javasolt eljárás pontosságát részletesen bemutatjuk.

Célunk, hogy egy félautomatikus eljárást hozzunk létre, amely valós helyzetekben is felhasználható. Egy GoPro Hero 4 Black kamerával és Velodyne LiDAR-ral teszteltük algoritmusunk. A kalibráció a következőek szerint ment

végbe: Először egy közönséges dobozt helyeztünk a kamera elé úgy, hogy az a LiDAR szemszögéből is jól látható legyen. Felvételt készítettünk a kamerával, és lementettük a pontfelhőt a LiDAR segítségével. Kézi beavatkozással elkülönítettük a doboz pontjait a pontfelhő többi részétől, a doboz méreteit lemértük. Az algoritmus ráillesztette a dobozt a ponthalmazra, majd a felhasználónak ki kellett választania a doboz sarkaihoz tartozó pixel koordinátákat a képen. A 3D-2D megfeleltetések alapján pedig ki lehetett számolni a LiDAR és a kamera közti transzformációt.

Néhány úgynevezett online módszer is elérhető a szakirodalomban, amelyekkel a kalibráció elvégezhető menet közben, azaz az eszköz használata során. Ezek közül néhány esetében nem is szükséges emberi közbelépés, viszont ezek nem olyan pontosak, mint az offline módszerek. Mi azt az esetet vizsgáljuk, amikor a kamera-LiDAR pár fix pozícióban helyezkedik el egymáshoz képest, ezért az egyszeri, offline kalibráció is használható. A pontosabb kalibráció elősegíti azon algoritmusok működését, amelyek felhasználják mind a két eszközt.

A 3D LiDAR eszközök legnagyobb előnye a kibocsájtott fény típusa, amely különbözik a látható fénytől. Ezért bármilyen megvilágítási környezetben használható és képes a világunkról 3D pontfelhőt szkennelni nagy távolságok esetén is. Viszont ezen eszközök még mindig elég drágák és a felbontásuk limitált – a Velodyne 64 LIDAR is csak 64 pontot képes horizontálisan letapogatni, lassú frissítési idővel. A kamerák olcsónak mondhatóak, így könnyen fel tudunk használni egyszerre többet is belőlük, melyekkel nagy felbontású, színes képeket nyerhetünk. Azonban használatuk nehézkes lehet különböző szélsőséges megvilágítási környezetekben, például az esti órákban. Ezen kívül árnyékok illetve takarás is nehezíti a képfeldolgozó algoritmusok dolgát. Egyértelmű, hogy a LiDAR eszközök hátrányát kompenzálhatjuk a kamerák előnyös tulajdonságaival és fordítva. Ezért a 3D LiDAR-t és a kamerákat gyakran használják egyszerre, főképp objektumok detektálásra, környezet rekonstruálásra és közlekedési feladatok megoldására. Ahhoz, hogy a két eszköz együtt tudjon dolgozni, szükséges a külső paraméterek meghatározása, ami a két eszköz relatív pozícióját és orientációját jelenti egymáshoz viszonyítva.

A kalibrációs eljárás lépéseit a következőek pontokban foglaljuk össze:

1. Egy dobozt kell a kamera és a LiDAR elé helyeznünk.
2. Felvételt kell készíteni a kamerával és le kell menteni a pontfelhőt a dobozról.
3. A doboz pontjait el kell különíteni a pontfelhő többi részétől.
4. A doboz-t detektálni kell és meghatározni annak sarokpontjait.
5. Ezen sarokpontokat ki kell választani a képen.
6. Végül a 3D-2D pontmegfeleltetésekből a külső paraméterek meghatározhatóak.

A cikk a következőek alapján épül fel. A 2. fejezetben egy rövid áttekintést adunk a témán belül megjelent korábbi cikkekről. A 3. fejezetben részletezzük a pontfelhő tisztítását és a doboz pontjainak elkülönítését. Az automatikus doboz illesztést a 4. szakaszban fejtjük ki, majd az 5. fejezetben lépésenként mutatjuk be az algoritmus működését egy valós példán. A 6. fejezetben összehasonlítjuk

algoritmusunkat egy konkurens módszerrel, végül a 7. szakaszban összefoglaljuk munkánkat.

2.. Irodalmi áttekintés

Több módszer is létezik a kamera-LiDAR pár kalibrációjára. A korábbiak még 2D LiDAR-t használtak, úgy mint [Zhang and Pless, 2004], majd az olcsóbb 3D LiDAR eszközök megjelenésének köszönhetően a kutatás átterelődött azok irányába. A módszerek több csoportra bonthatóak, sokan sakktáblákat használnak a kalibrációhoz, úgy mint [Geiger et al., 2012] és [Pandey et al., 2010], páran sík objektumokat, mint [Park et al., 2014] vagy [Veĺas et al., 2014] és páran egyáltalán nem használnak kalibrációs objektumok, mint például [Pandey et al., 2012]. A legtöbb esetben a LiDAR és a kamera már előre, külön-külön kalibrált, azaz a belső paramétereik ismertek. Léteznek olyan módszerek, melyekhez nem szükséges kalibrációs objektum vagy bármilyen előzetes tudás az eszközökről, viszont ezen módszerek pontossága bőven eltér az optimálistól.

A módszer, amelyet [Park et al., 2014] tárgyal közel áll a sajátunkhoz a felhasználás módját nézve. Homogén, fehér, sík háromszög vagy négyzet alakú táblát használnak a kalibrációhoz. Bár a módszerhez egynél több képet szükséges készíteni a tábláról, vagy több táblát kell használni egy kép felvétele során – a mi módszerünknél csak egy felvétel szükséges –. Ezen kívül a síkbeli tábla pontjait becslésekkel közelítik, nem pedig pontos mérésekkel, ami befolyásolhatja a kalibráció pontosságát.

A [Gong et al., 2013] módszer esetében több, mint 2 felvételt kell készíteni három egymásra merőleges síkról a kamerával és a LiDAR-ral egyaránt. Ez több feldolgozandó adatot jelent, a tesztjeik során 20 másodperc kellett a kalibrációhoz, melyet 9 megfigyelésből állítottak össze.

Egy nagyon érdekes módszert publikált [Veĺas et al., 2014], hiszen egy nem hétköznapi kalibrációs tárgyakt választottak ki. Fehér háttér elé egy sík lapot akasztottak, melyen 4 kör alakú lyukat vágtak ki. A pontfelhőben pedig az ez által keletkeződött négy lyukat automatikusan detektálták. [Levinson and Thrun, 2013] korábbi munkáin alapul a módszer és csak egy képet szükséges készíteni a kalibrációhoz.

[Geiger et al., 2012] egy új módszert mutatatott be a kamera-LiDAR pár kalibrálására, amelyhez elegendő egy kamera felvétel és egy LiDAR pontfelhő. A módszer teljesen automatikus, viszont több sakktáblát kell elhelyezni a kalibráció sikeréhez. Az eljárást részletesen be fogjuk mutatni a 6. fejezetben, ahol összehasonlítjuk a sajátunkkal.

A LiDAR az időt méri a lézer sugár kibocsájtása és visszaérkezése között, majd kiszámolja a távolságot a fény sebessége és az előbbi időkülönbség alapján. Ám a visszaverődés különböző lehet színenként. Ezért egy sakktábla pontjai például nem fognak egy síkon maradni a LiDAR pontfelhőjében, ez a tulajdonság a [Park et al., 2014]-ban részletezve megtalálható.

3.. Pontfelhő szeparálása

3.1.. Általános ötlet

A főbb gondolat az algoritmusunk mögött az, hogy ha a kamera belső paraméterei, a doboz sarkinak pontjai a LiDAR koordináta rendszerében és ezen sarokpontok helyzete a képen ismert, akkor a LiDAR és a kamera koordináta rendszere közti transzformáció meghatározható. A külső paramétereket a jól ismert PnP (Perspective-n-Point) probléma megoldása adja, ami egy fogatási mátrix és egy eltolás vektor. A lépések melyekkel a fenti feltételek teljesíthetőek a következő fejezetekben kerülnek részletezésre.

3.2.. A kalibráció felépítése

A sikeres kalibrációhoz legalább 4 különböző 3D-2D pontmegfeleltetésre lesz szükségünk. Ezért a kalibrációs dobozt úgy kell a kamera elé helyezni, hogy három oldalának láthatónak kell lennie a képen, illetve a pontfelhőn egyaránt. Bármilyen tetszőleges doboz használható a kalibrációhoz, de a doboz méreteit, azaz a doboz három látható oldalának a hosszát le kell mérnünk. Fontos megjegyezni, hogy a kalibráció során csak egyetlen képet és egy pontfelhőt használunk fel, viszont az eljárást ki lehet terjeszteni több kamera és több LiDAR eszköz egyidejű kalibrálásra. A részletek két LiDAR kalibrációjáról a 6.. fejezetben találhatóak.

A méréseink során egy GoPro Hero 4 Black kamerát és két különböző típusú LiDAR eszközt használtunk. Az egyik egy Velodyne VLP-16, ami 16 csatornás LiDAR eszköz, ezért meglehetősen ritka pontfelhőt képes generálni. A másik pedig a Velodyne HDL-64, ami 64 csatornával felszerelt, így sűrűbb pontfelhőt észlelésére képes.

A GoPro Hero 4 Black egy akciókamera, melyet főleg sportolás megörökítésére használnak. Az eszköz kalibrációja meglehetősen nehéz, mert nagy látószöggel rendelkezik, így nagyfokú torzítással kell számolni, ez a 1. ábrán is jól látható. Ipari projektek során nem használatos ez az eszköz, mi is csak a könnyű hordozhatósága miatt használtuk. A kis mérete ellenére a kamera képes 12MP fotókat és 4K videókat készíteni 30 FPS gyorsasággal. Az általunk bemutatott módszer természetesen bármilyen típusú LiDAR-kamera párosítással használható.

A LiDAR-kamera pár kalibrálása előtt a kamerát be kell kalibrálnunk. Ehhez a lyukkamera modellt használtuk, jelölje $P = C[R|t]$ a projekciós transzformációt, ahol C a kamera mátrix, R a forgatási mátrix és t az eltolás vektor. Ahogy azt már említettük, a torzítást is figyelembe kell vennünk, mi radiális torzítást használtunk (k_1, k_2) paraméterekkel. A C és (k_1, k_2) adja a kamera belső paramétereit, amiket a [Zhang, 2000] által bemutatott módszerrel határoztunk meg. Az R és a t pedig a külső paraméterek, amik a LiDAR és a kamera közötti koordináta rendszer váltást jelölik. A mi kalibrációs eljárásunk célja ezen külső paraméterek meghatározása.

A kalibrációs dobozt a kamera és LiDAR elé kell helyeznünk úgy, hogy annak három oldala jól látható legyen mind a két eszköz nézetéből, egy ilyen elhelyezés

látható az 1. ábrán. Majd a kamera képét és a LiDAR pontfelhőt kell felhasználni a továbbiakban. A kalibráláshoz elegendő csak egyetlen felvételt készíteni egyetlen kalibrációs objektumról. Más módszerek esetében több felvétel szükséges egy objektumról vagy több objektumról kell egy felvételt készítenünk, ami minden esetben több adat feldolgozását jelenti.

1. ábra. Példa a kalibrációs doboz megfelelő elhelyezkedésére.

3.3.. Pontfelhő klaszterezése

Az algoritmus működéséhez szükséges a kézi beavatkozás, hogy el tudjuk különíteni a doboz pontjait a pontfelhő többi részétől. A pontok jól klaszterezhetőek a Mean-Sift algoritmus által [Fukunaga and Hostetler, 2006]. A legtöbb mérésünk során a dobozt egy székre raktuk, mert a LiDAR eszközök horizontális látószöge nagyon kicsi, ezért a Mean-Sift a széket és a rajta fekvő kalibrációs dobozt egy klaszterbe sorolta. Eljárásunk képes az ilyen klasztereket is feldolgozni, a doboz egyedi tulajdonságainak keresésével. Az kézi beavatkozás során csak a megfelelő klasztert szükséges kiválasztani, amely tartalmazza a doboz pontjait.

3.4.. Doboz elkülönítése

Az alfejezet célja, hogy pontosan el tudjuk különíteni a doboz három látható oldalát a pontfelhő többi részétől. Mivel a pontfelhő többféle tárgy pontjait is

tartalmazhatja, először a dobozhoz tartozó pontokat kell felismernünk. Az előző lépésben a doboz környezete kézzel lett kiválasztva, most ebből a környezetből kell kiemelnünk azon pontokat, melyek a dobozhoz tartoznak. Habár ez a környezet már kevesebb outlier – a dobozhoz nem tartozó – pontot tartalmaz, erősebb szűrés szükséges a pontfelhőn.

A probléma megoldása két lépésből áll. Először szekvenciális RANSAC algoritmust használtuk a pontfelhő síkokra való felbontására – Euklideszi távolságot alkalmazva. Miután a pontfelhő síkjait meghatároztuk, a második lépésben kihasználjuk a doboz azon tulajdonságát, hogy a három látható oldala merőleges egymásra. Ezért azt a három síkot választjuk ki, amelyekre a következő hiba mértéke minimális:

$$E(n_1, n_2, n_3) = |n_1 n_2'| + |n_1 n_3'| + |n_2 n_3'|, \tag{1}$$

ahol n_k a k-adik sík normál vektora ($k \in \{1, 2, 3\}$). A pontfelhő ezen szegmense általában már nem tartalmaz sok síkot, így a síkok közötti kimerítő kereséshez nem szükséges nagy számítási idő.

4.. Doboz illesztése a pontfelhőre

A doboz-illesztés két lépésből épül fel. Először megpróbáljuk a dobozhoz nem tartozó pontokat eltüntetni a pontfelhőből, majd a második lépésben egy iteratív algoritmussal ráillesztjük a dobozt a megmaradt pontokra.

4.1.. Outlier pontok eltüntetése

A méréseink során azt tapasztaltuk, hogy a kiválasztott síkokon még mindig található néhány outlier pont, ezek száma függ a doboz és LiDAR relatív orientációjától, utóbbi felbontásától és a doboz textúráltságától. A [Park et al., 2014] cikkben említettek alapján a doboz különböző színei szisztematikus visszaverődési kettősséget okozhatnak, azaz a síkon található különböző színek zajjal szennyezhetik annak pontjait. Ezért ebben a lépésben ismét egy RANSAC algoritmust használunk, de most feltételezzük a merőlegességet. Az előző lépés 3 ponthalmazt eredményezett nekünk, az egy-egy oldalhoz tartozó pontok halmazát. Jelölje L_1, L_2, L_3 rendre az első, a második és a harmadik síkhoz tartozó pontokat. A síkok sorrendje nem számít az algoritmus számára, csak a megkülönböztetés miatt jelöljük őket így.

Ezután három pontot kell kiválasztanunk az L_1 halmazból, hogy meghatározzuk az első síkot. A második síknak merőlegesnek kell lennie az elsőre, így már csak 2 pont szükséges az L_2 halmazból. Végül a harmadik síknak merőlegesnek kell lennie az előző kettőre, így elegendő 1 pontot választanunk az L_3 halmazból. Majd megvizsgáljuk az inlierek számát kis hibatűréssel. Inlierek azok a pontok, amelyek közel helyezkednek el valamelyik síkhoz. Pár iteráció után azt a három síkot választjuk, amelyek a legnagyobb inlier számmal rendelkeznek és az outlier pontokat pedig eldobjuk.

Az előző lépésben egy iteratív RANSAC algoritmust használtunk a doboz síkjainak elkülönítésére a pontfelhő többi részétől, most pedig ismét egy RANSAC alapú algoritmust használtunk fel. Ez a két lépés egymás után alkalmazva redundánsnak tűnhet, de nem az. A fő célunk az előző lépés során a doboz síkjainak a közelítő meghatározása volt, ezért nagyobb hibaküszöböt engedtünk meg, illetve a síkok merőlegességét sem követeltük meg. Ha kisebb hibaküszöbbel dolgoztunk volna, akkor a mérés zajossága miatt – főleg a nagyobb felbontású LiDAR esetében –, egyes a valóságban egy síkban lévő pontok több síkra estek volna szét, ez pedig megnehezítette volna a doboz síkjainak kiválasztását. A második RANSAC használatának célja pedig az olyan outlierek eltávolítása volt, amelyek a doboz egyik síkjához tartoznak, de túlságosan nagy zajjal terheltek, mint például az olyan esetekben, amit [Park et al., 2014] is említ. A mostanit lépés egy sokkal tisztább pontfelhőt eredményez.

4.2.. Iteratív doboz-illesztés

Az előző lépésben részletezett outlier eltávolítás után megkezdhetjük a doboz illesztést a pontfelhőre. Az algoritmusunk a legkisebb négyzetes illesztésen alapul és két lépésből épül fel, melyek egymás után ismétlődnek a konvergencia beállásáig. Először kiválasztunk két síkot a három közül, majd elforgatjuk őket a metszésvonaluk körül, majd eltoljuk a három síkot külön-külön a normál vektoruk irányába.

Síkok forgatása. Az algoritmus első lépésében kiválaszt kettőt a doboz síkjai közül, és a metszési egyenesük körül elforgatja őket. A fogatás célja, hogy a négyzetes távolságot csökkentsük a síkok és a hozzájuk tartozó pontok között.

Jelölje $Q_{1,j}, j \in \{1, 2, ...m_1\}$ és $Q_{2,j}, j \in \{1, 2, ...m_2\}$ az egyik, illetve a másik síkhoz tartozó pontokat. Legyen P_1 és P_2 egy-egy pont a síkokon (melyeknek nem feltétlenül kell benne lenniük a $Q_{i,j}, i \in \{1, 2\}$ halmazokban), illetve N_1 és N_2 legyenek a síkokhoz tartozó normálvektorok. A forgatást egyszerűbb úgy leírni, ha annak tengelye megegyezik a Z tengellyel. Így minden pontot és normál vektort áttranszformálunk egy új koordináta rendszerben.

A Z tengely körüli ϕ szöggel való forgatást a következő mátrix írja le:

$$R_Z(\phi) = \begin{bmatrix} \cos(\phi) & -\sin(\phi) & 0 & 0 \\ \sin(\phi) & \cos(\phi) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

A pontok és a normál vektorok áttranszformált megfelelőit az új koordináta rendszerben jelölje rendre $\widehat{Q_{i,j}}$, $\widehat{P_i}$ és $\widehat{N_i}$.

A következőképpen írható le a síkok és a hozzájuk tartozó pontok közötti távolságok összege, célunk ennek a minimalizálása lesz:

$$Err = \sum_{i=1}^{2} \sum_{j=1}^{m_i} ((\widehat{Q_{i,j}} - \widehat{P_i})^T \widehat{N_i})^2$$

A fenti távolság felírható a forgatási paraméter ϕ függvényében is, ami a síkok Z tengely körüli forgatását jelöli:

$$Err(\phi) = \sum_{i=1}^{2} \sum_{j=1}^{m_i} ((\widehat{Q_{i,j}} - R_Z(\phi)\widehat{P_i})^T (R_Z(\phi)\widehat{N_i}))^2$$

Ahhoz, hogy megkeressük a fenti függvény milyen ϕ értéknél veszi a fel a minimumát, a függvény deriváltját kell vennünk.

A derivált függvény gyökeinek megtalálásához a [Brent, 1973]-ban bemutatott algoritmust használtuk. Ahol a $Err(\phi)$ függvény deriváltjának értéke megközelíti a nullát, ott az eredeti függvény lokális minimumot vagy lokális maximumot vesz fel. A nullához közeli ϕ érték a minimális forgatási szöget fogja adni, mivel a síkokhoz elég közel esnek a pontjaink. Így 0 mint kezdőérték az alkalmazott módszer esetében kielégítő.

Miután megtaláltuk azt a szöget, amivel a két síkot elforgatva a legkisebb négyzetes hibát kaptuk, visszatranszformáljuk őket az eredeti koordináta rendszerükbe. Ezt a lépést megismételjük a maradék két síkpárra is. Két sík ilyen módon való elforgatása nem változtatja meg a harmadikkal való merőlegességüket. Miután mind a három síkpár elforgatásra került, az algoritmus a síkokat a normálvektoruk irányába fogja mozgatni.

Síkok mozgatása. Miután a síkokat elforgattuk úgy, hogy azok a legkisebb négyzetes hibát közelítsék, a síkok mozgatásával folytatódik az algoritmus. A síkokat kizárólagosan csak a saját normálvektoruk irányába mozgatjuk, hogy megőrizzük merőlegességüket. A legjobb helyzet a síkok számára az, ahol legkisebb a négyzetes távolság a sík és pontjai között. Ahogy a forgatási lépés során, itt is bevezettünk egy hibát. Legyen

$$Err(\alpha) = \sum_{j=1}^{m_i} ((Q_{i,j} - (\alpha \cdot N_i + P_i))^T (N_i))^2$$

a hiba az i. síkra felírva. Keresünk az α értéket, amely az eltolás nagyságát fogja jelölni a normálvektor irányába.

Az α értéket ott veszi fel a függvényünk, ahol a derivált értéke nulla. Hasonlóan forgatási lépéshez, itt is a [Brent, 1973]-ban leírt módszert használjuk 0 kezdeti értékkel. Mivel a síkjaink már közel helyezkednek el a kívánatos pozíciójukhoz, így várhatóan csak kisebb eltolásra van szükségük. Ezt a lépést megismételjük minden egyes sík esetén.

A végső algoritmus A fentebb részletezett forgatás és eltolás lépések a konvergenciáig ismétlődnek. Az tapasztaltuk, hogy abból a kezdőpontból, amit az utolsó RANSAC algoritmus adott, kevesebb, mint 30 iteráció elegendő a megfelelő konvergenciához. Kipróbáltuk, hogy az origóból mint sarokpontból és a tengelyek által meghatározott síkokból indítva is képes volt konvergálni az algoritmus.

Miután a síkokat meghatároztuk, a doboz sarkai könnyen kiszámíthatóak, ismerve a doboz méreteit. Ezután a sarkokhoz tartozó pixeleket kell kiválasztani a képen. Ezt a lépést az emberi beavatkozásra bízzuk. A kiválasztott sarokpontok a [Harris and Stephens, 1988]-ban publikált ún. Harris detektorral lettek finomítva.

Ha a 3D-2D pont megfeleltetéseket ismerjük, akkor a külső paraméterek meghatározhatóak az ismert PnP probléma megoldása által. Az implementációnkban a [Lepetit et al., 2009]-ben bemutatott EPnP algoritmust alkalmaztuk.

5.. Implementációs megfontolások

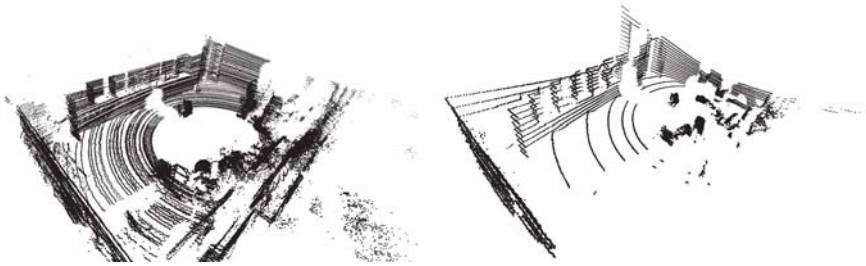
Amikor felvételt készítettünk a kamerával és a LiDAR-ral a kalibrációhoz, akkor a dobozt az eszközök elé kellett helyeznünk úgy, hogy annak három oldala jól látható legyen mind a két eszköz szemszögéből. A doboz pontfelhőjén látható pontjainak a száma függ a doboz és a LiDAR közötti relatív orientációtól, ezért előfordulhat, hogy az egyik oldal alig látszódik, kevesebb pontot tud a LiDAR regisztrálni rajta. Ezért a 3.4. fejezetben nem is foglalkoztunk a síkok pontjainak a számával, csak az egymásra való merőlegességükkel.

Ebben a fejezetben egy példán keresztül levezetjük a kalibráció lépéseit, és példát mutatunk annak előnyeire és robusztusságára.

2. ábra. A kalibráció kiterjeszthető több doboz alkalmazására is.

A 2. ábrán látható, hogy három dobozt használtunk fel a kalibrációhoz. A 3. és 4. fejezetekben bemutatott eljárás csak egy dobozt használ fel, de könnyen

kiterjeszthető több doboz használatára is. A dobozok számának növelésével nagyobb pontosságot tudunk elérni.



3. ábra. Két pontfelhő ugyanarról a szobáról. A bal oldalit a Velodyne 64, a jobb oldalit a Velodyne 16-os LiDAR-ral készítettük.

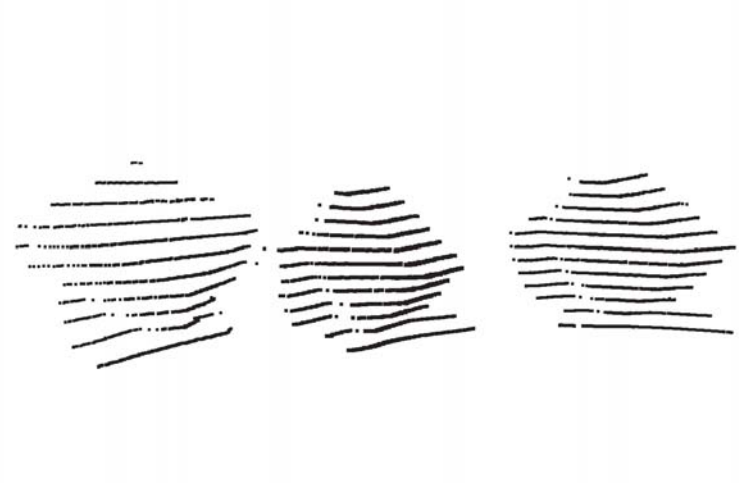
Ahogy a 3. ábrán látható, a pontfelhő felbontása a LiDAR eszköz típusától függ. Minél több csatornát használ a LiDAR, annál több pontot képes mintavételezni a környezetéből. A [Park et al., 2014] cikkben említett módszer megpróbálja megkeresni a LiDAR elé kihelyezett tábla szélét az oda eső pontok segítségével. Egy kisebb felbontású LiDAR esetében a pontok közötti hézag nagyobb, így a tábla széleit nehezebb megbecsülni. Ezzel szemben a mi algoritmusunk síkokat keres, majd azok metszéspontjait veszi figyelembe. Eljárásunkat nagy felbontású (64 csatornás) és kis felbontású (16 csatornás) LiDAR eszközökön is teszteltük.

A következő lépés a doboz pontjainak elkülönítésre a pontfelhő többi részétől. Itt kézi beavatkozásra van szükség, a felhasználónak ki kell választania azt a klasztert, ami a dobozt tartalmazza. Ezeket a klaszterekt a Mean-Sift algoritmus adja. Az 4. ábrán a dobozok megmaradt pontjai láthatóak, Velodyne 64 által felvett pontfelhőből. A következőekben ezt a pontfelhőt használjuk fel a példák során, hiszen ez jól látható nyomtatásban is.

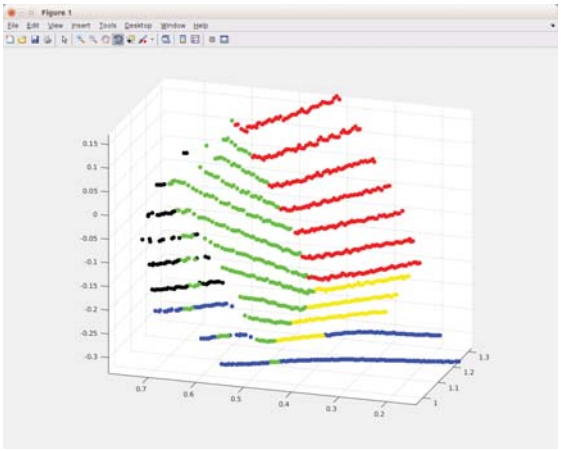
Ahhoz, hogy eltüntessük a maradék outlier pontokat, amelyek nem részei a doboznak, síkokat detektálunk a 3. fejezetben leírtak szerint. Az 5. ábrán láthatóak a detektált síkok, melyek közül azokat választjuk ki, amelyek leginkább merőlegesek egymásra.

Miután a doboz három síkját megtaláltuk, a többi outlier pont kivágása következik. Ezek a pontok a nagy mérési hiba miatt befolyásolnák a legkisebb négyzetes doboz illesztő algoritmust, ezért ki kell szűrnünk őket. Példa az ilyen pontokra a 6. ábrán látható. Itt a 4. fejezetben leírtakat használjuk a doboz illesztéséhez, 10 mm-es küszöbértéket használva az inlier-ek számításához.

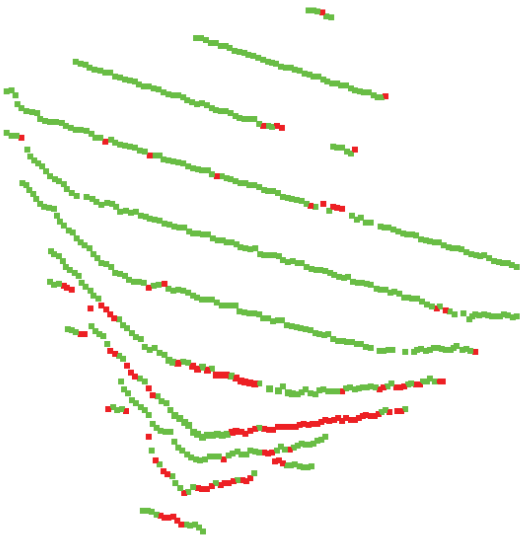
Végül a négyzetes doboz illesztést alkalmazzuk a megmaradt pontokat. A részleteket a 4. fejezetben olvashatjuk. Az algoritmus a konvergencia beállásáig finomítja a síkokat, majd kiszámolja a doboz sarokpontjait a fizikai méretek segítségével. A végső eredményt a 7. ábrán láthatjuk.



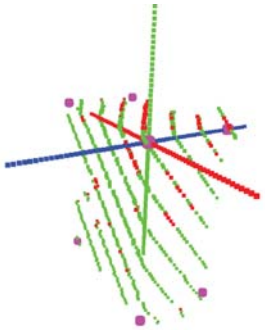
4. ábra. A dobozok kivágott pontfelhői a Velodyne 64-es LiDAR pontfelhőjéből. A vágásnak nem kell pontosnak lennie, algoritmusunk ezeken is megtalálja a doboz pontjait.



5. ábra. A pontfelhőben megtalált síkok. Az algoritmus a piros, zöld és sárga síkot választotta a doboz síkjainak.



6. ábra. Pirossal színeztük az outlier pontokat, melyeket a merőleges sík keresése után detektáltunk.



7. ábra. A végső doboz a pontokra illesztve. A rózsaszín pontok a doboz sarkait jelölik.

6.. Összehasonlítás

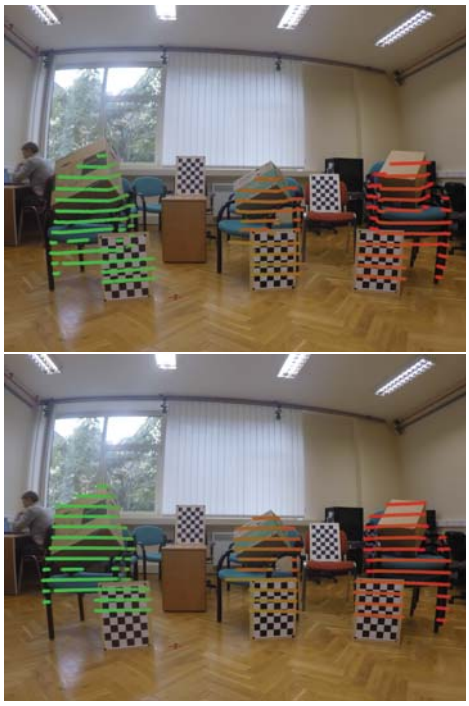
Összehasonlítottuk algoritmusunk a [Geiger et al., 2012] -ban bemutatott eljárással. A két kalibrációs eljárás hasonló abban, hogy csak egy képet és egy LiDAR felvételt használ a kalibrációhoz. De az ő esetükben több sakktáblát kell a kamera elé helyezni, a mi esetünkben pedig dobozokat dolgozunk fel. A [Geiger et al., 2012] megemlítették, hogy több képet készítve növelhető a kalibrációjuk pontossága. Az összehasonlításunk a következőképpen ment végbe:

Először a dobozokat helyeztük a kamera és a LiDAR elé, majd felvételt készítettünk a kamerával. Ezután elhelyeztünk néhány sakktáblát úgy, hogy azok jól láthatóak legyenek a pontfelhőn és a kamerán egyaránt. Majd két képet készítettünk, az egyiket pontosan az előzővel megegyező helyről, illetve egy másikat egy másik szemszögből, majd ismét lementettük a pontfelhőt. A dobozok és a sakktáblák nem lettek elmozdítva a két kép között. A Velodyne-16 LiDAR-t használtuk a teszt során, szerettük volna a nagyobb LiDAR-t is használni, de az túlságosan zajos pontfelhőt generált a sakktáblákról, így a konkurens módszer nem volt képes azokat felismerni.

8. ábra. Ezt a három pontfelhőt választottuk ki a Mean-Sift algoritmus általt adott klaszterekből a teszt során. A színes gömbök a pontfelhőkön megtalált dobozok sarkait jelölik.

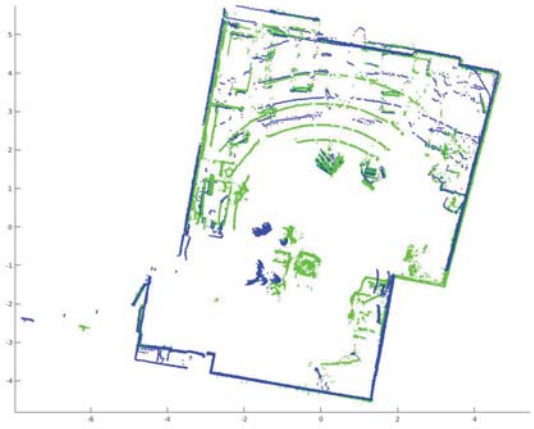
Az ilyen kalibrációs méréseket nehéz validálni, hiszen a valós külső paraméterek nem meghatározhatóak. Ezért úgy döntöttünk, hogy a székek pontfelhőjét fogjuk visszavetíteni a képekre, és ott vizsgáljuk a különbségeket. A visszavetítés eredménye a 9. ábrán látható.

A fő különbséget a székek lábainál és a sakktáblák körül vehetjük észre. Amíg a konkurens eljárás néhány sakktábla pontot a földhöz tartozó pixelekre vetít vissza, addig a miénk a megfelelő helyen tartja azokat. Megfigyelhetjük, hogy a dobozok pontjai is különböző helyre vetítődtek vissza.



9. ábra. Ezeket a képeket készítettük a teszt során. A felsőre a konkurens módszerrel vetítettük a pontokat, az alsóra pedig a saját módszerünkkel.

Az eljárásunk kiterjeszthető LiDAR-LiDAR pár kalibrálásra is. A két pontfelhő egyesítésére léteznek algoritmusok ám ezek pontossága nem megfelelő. Ebben a tesztben egy dobozt használtunk fel a kalibrációhoz, és célunk a két LiDAR koordináta rendszere közötti transzformáció meghatározása volt. A dobozt úgy helyeztük el, hogy három odala jól látható legyen mind a két pontfelhőn. Majd kivágtuk a dobozt környezetét a pontfelhőkből, ahogy azt a 3. fejezetben leírtuk. Majd az algoritmust lefuttatva mind a két pontfelhőre külön-külön megtaláltuk és párosítottuk a doboz sarokpontjait, végül a transzformációt ezekből a pontpárokból számoltuk ki SVD-alapú pontregisztrációs algoritmus segítségével. A két pontfelhő egyesítése a 10. képen látható madártávlatból.



10. ábra. Két LiDAR kalibrációja. Az egyik pontfelhő pontjait zölddel, a másik pontjait pedig kékkel rajzoltuk ki.

7.. Összegzés

Cikkünkben egy LiDAR-kamera páros kalibrálására használható módszert mutattunk be, amely egy közönséges dobozt használ fel. A módszer képes a doboz pontjainak kiválasztására mind ritka, mind pedig sűrű pontfelhő esetében. Tesztjeinkhez 16 és 64 csatornás LiDAR mérőeszközt használtunk. Bemutattuk, hogy hogyan lehet a pontfelhőket megtisztítani az outlier pontoktól és részleteztük az iteratív doboz kereső algoritmust. Összehasonlítottuk módszerünk az egyetlen hasonló konkurrens módszerrel, illetve bemutattuk, hogy a módszer használható LiDAR-LiDAR pár kalibrálására is.

A jövőben szeretnénk módszerünket továbbfejleszteni, az emberi beavatkozást nullára csökkenteni. A módszer pontosságát további tesztekkel tervezzük még meggyőzőbben igazolni.

Hivatkozások

[Brent, 1973] Brent, R. P. (1973). *Algorithms for minimization without derivatives.* Prentice-Hall series in automatic computation. Englewood Cliffs, N.J. Prentice-Hall.

[Fukunaga and Hostetler, 2006] Fukunaga, K. and Hostetler, L. (2006). The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Trans. Inf. Theor.*, 21(1):32–40.

[Geiger et al., 2012] Geiger, A., Moosmann, F., Car, O., and Schuster, B. (2012). Automatic camera and range sensor calibration using a single shot. In *IEEE International Conference on Robotics and Automation, ICRA 2012, 14-18 May, 2012, St. Paul, Minnesota, USA*, pages 3936–3943.

[Gong et al., 2013] Gong, X., Lin, Y., and Liu, J. (2013). 3d lidar-camera extrinsic calibration using an arbitrary trihedron. *Sensors*, 13(2):1902.

[Harris and Stephens, 1988] Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In *In Proc. of Fourth Alvey Vision Conference*, pages 147–151.

[Lepetit et al., 2009] Lepetit, V., Moreno-Noguer, F., and Fua, P. (2009). Epnp: An accurate o(n) solution to the pnp problem. *Int. J. Comput. Vision*, 81(2):155–166.

[Levinson and Thrun, 2013] Levinson, J. and Thrun, S. (2013). Automatic online calibration of cameras and lasers. In *Robotics: Science and Systems*.

[Pandey et al., 2010] Pandey, G., McBride, J., Savarese, S., and Eustice, R. (2010). Extrinsic calibration of a 3d laser scanner and an omnidirectional camera. In *7th IFAC Symposium on Intelligent Autonomous Vehicles*, volume 7, Leece, Italy.

[Pandey et al., 2012] Pandey, G., McBride, J. R., Savarese, S., and Eustice, R. M. (2012). Automatic targetless extrinsic calibration of a 3d lidar and camera by maximizing mutual information. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, pages 2053–2059, Toronto, Canada.

[Park et al., 2014] Park, Y., Yun, S., Won, C. S., Cho, K., Um, K., and Sim, S. (2014). Calibration between color camera and 3d lidar instruments with a polygonal planar board. *Sensors*, 14(3):5333–5353.

[Veĺas et al., 2014] Veĺas, M., Španěl, M., Materna, Z., and Herout, A. (2014). Calibration of rgb camera with velodyne lidar. In *WSCG 2014 Communication Papers Proceedings*, volume 2014, pages 135–144. Union Agency.

[Zhang and Pless, 2004] Zhang, Q. and Pless, R. (2004). Extrinsic calibration of a camera and laser range finder (improves camera calibration). In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems, Sendai, Japan, September 28 - October 2, 2004*, pages 2301–2306.

[Zhang, 2000] Zhang, Z. (2000). A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(11):1330–1334.