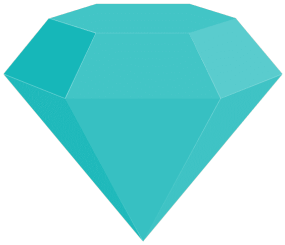
XXIII. reál- és humántudományi Erdélyi Tudományos Diákköri Konferencia (ETDK)
Kolozsvár, 2020. május 23–26.

Volumiz3D

Térfogatbecslés képek alapján



Szerzők:

Bándi Nándor
Babeş–Bolyai Tudományegyetem, Matematika és Informatika Kar, Informatika szak, III. év
Tunyogi Rudolf - Bálint
Babeş–Bolyai Tudományegyetem, Matematika és Informatika Kar, Informatika szak, III. év

Témavezetők:

dr. Sulyok Csaba, tanársegéd
Babeş–Bolyai Tudományegyetem, Matematika és Informatika Kar
Szabó Zoltán, szoftverfejlesztő
Codespring
Farkas Eszter, szoftverfejlesztő
Codespring

CODESPRING

Kivonat

A **Volumiz3D** projekt célja egy olyan szoftverrendszer kifejlesztése, amely lehetőséget ad tárgyak méretének és térfogatának megbecsülésére kizárólag képek alapján. Az első lépés a kamerák helyzetének megbecsülése a képeken található kulcspontok párosítása alapján. A második lépés egy sűrű pontfelhő felépítése, majd a felhőre egy Poisson felület illesztése, ugyanakkor a felhő konvex burkának a létrehozása. A harmadik lépés a konvex burok és a felület metszése, majd ennek a felületnek a továbbfinomítása. Az utolsó lépés a finomított metszet térfogatának a metrikus skálára hozása a referencia objektum segítségével.

A rendszer két részből áll: egy Android kliensalkalmazásból és egy központi szerverből. Az Android alkalmazás interaktív felületet nyújt a felhasználó számára, amely valósidejű visszajelzést ad az elkészült képek minőségéről valamint helyzetéről, így segítve a szükséges bemeneti adathalmaz létrehozását. Az elkészült képek a szerverhez lesznek továbbítva, ahol azok feldolgozásra kerülnek.

Tartalomjegyzék

Bevezető

A Volumiz3D projekt célja egy olyan Android alkalmazás fejlesztése, amely használatával tárgyak térfogatát lehet megbecsülni. A csapat kifejlesztett egy két főmodulból álló szoftverrendszert, amely a mobilalkalmazás által készített képeket egy magas teljesítménnyel rendelkező szerverrel dolgozza fel.

A dolgozat a következőképpen épül fel: az 1. fejezetben a számítógépes látás alapjai vannak bemutatva, a 2. fejezetben az korszerűbb, fejlettebb képfeldolgozási módszerekről esik szó, a 3. fejezet a képfeldolgozás eredményeként kapott modellből a térfogat létrehozásának menetét írja le, a 4. fejezetben a rendszer felépítéséről és működéséről van szó, illetve az 5. fejezetben az elért eredmények, következtetések és továbbfejlesztési lehetőségek vannak bemutatva.

A projekt a 2019-es nyári szakmai gyakorlat időtartama alatt indult útjára és ez idő alatt főleg a megfelelő képfeldolgozási módszerek keresésén volt a hangsúly. A csapat a mesterséges intelligencia és a számítógépes látás által nyújtott lehetőségeket is megvizsgálta, de végül a számítógépes látást alkalmazva jött létre a végső megoldás.

A szakmai gyakorlatot követően *a csoportos projekt* keretein belül folytatódott a projekt fejlesztése. Ebben az időszakban az Android alkalmazás fejlesztésén és a szerveroldali architektúra kialakításán volt a hangsúly. A csapat ez idő alatt kibővült néhány új taggal, név szerint: Antal Áron, Gálfi Csaba, Trinfa Botond és Tasnádi Zoltán.

A projekt létrejöttéért köszönet illeti a Codespring-et, illetve az általuk kijelölt mentorokat, akik név szerint a következők: Farkas Eszter, Szabó Zoltán, illetve a Mentorprogram irányítója dr. Sulyok Csaba.

1. A számítógépes látás alapjai

Ahhoz, hogy egy tárgynak a térfogatát meg lehessen becsülni, ábrázolni kell azt. Ez az ábrázolás ebben az esetben azt jelenti, hogy létre kell hozni az adott tárgynak a térbeli (háromdimenziós) modelljét, úgy, hogy annak reprezentatív térfogata legyen.

1.1. Térbeli rekonstrukció

A *térbeli rekonstrukció* vagy *3D-s rekonstrukció* az a folyamat, amely során egy adott tárgynak a számítógépes 3D-s reprezentációja jön létre [16]. Térbeli rekonstrukció létrehozására léteznek *aktív* és *passzív* módszerek is.

1.1.1. Aktív módszerek

Az aktív rekonstrukciós módszereknél két fő megközelítés létezik, az egyik esetben a felvevő eszköz kölcsönhatásba lép a vizsgált tárggyal a fényképezés ideje alatt, míg a másik esetben szenzorinformációk alapján von le következtetéseket.

Ebből a módszer kategóriából néhány példa:

- **Irányított fény** (Structured-light) : az irányított fény módszerrel egy projektor jól ismert mintázatú képet vetít az adott tárgyra, amely minta visszaverődése alapján ki lehet következtetni a tárgy alakját. [14]
- **SLAM** (Simultaneous localization and mapping): a SLAM módszer a kamerán kívül gyorsulásmérőt, magnetométert, giroszkópot is igénybe vesz, ezen szenzorok ötvözéséből (*sensor fusion*) nemcsak, hogy meghatározza egy készülék térbeli helyzetét, hanem rekonstruálja is az azt körülvevő teret. [27]
- **Infravörös** lézer: a módszer infravörös sugarak kibocsátásából, és azok visszaverődéséből kiszámítja a tér egy részének a mélységét. Ezen mélységek összesítéséből megkapható a tér azon részének a struktúrája. [15]

1.1.2. Passzív módszerek

A passzív módszerek az aktívakkal szemben nem használnak fel semmit, csupán a kész fényképeket dolgozzák fel. Ez azt jelenti, hogy nincs szükség semmiféle szenzorra ahhoz, hogy működhessen ez az eljárás, viszont ez azt is jelenti, hogy a kép feldolgozásából kell kinyerni minden információt amire szükség van. A passzív módszereknél is több kategóriáról beszélhetünk:

- **Árnyék alapú** [6]: az árnyék alapú alakmeghatározás a fények irányából, és az árnyékok vetüléséből következtet a tárgy 3D-s modelljének a tulajdonságaira. Ezzel főleg bizonyos pontok mélységét lehet meghatározni.

- **Textúra alapú**: a textúra alapú módszerek felhasználják a tárgyon található textúra dísztorzióját és ebből következtetnek a felület milyenségére.

- **Voxel alapú**: a voxel alapú módszerek a teret felosztják kis, egyenlő méretű kockákra, majd mindegyik kockáról eldöntésre kerül, hogy a tárgy része e, az alapján, hogy a kocka képe bármely képen a tárgy felületén helyezkedik e el. A módszerek hátránya, hogy a tárgy homorú részeit nem képesek rekonstruálni. [18]

- **Sztereó látás**: a sztereó látás két *(vagy több)* különböző nézőpont felhasználásával tudja meghatározni a képeken látható pontok mélységét. A mélység meghatározásához összehasonlítja hogy a nézőpontok közötti távolság hogyan befolyásolja a képeken található pontok távolságát.

A projekt célja egy általános Android alkalmazás, amely a hardverek nagyon széles skálájának támogatását feltételezi. Annak érdekében, hogy minél több eszközön működhessen az alkalmazás, a *passzív módszerek* lettek alkalmazva a Volumiz3D megvalósításánál.

1.2. Sztereó látás

Ha adott két fénykép egy tárgyról, akkor ezeknek az összevetéséből több információt is ki lehet nyerni. Ezek közül a sztereó látás szempontjából a legfontosabb a tárgyhoz tartozó pontok **relatív mélysége**.

Egy **pont mélysége** alatt egy adott pont kamerához viszonyított távolságát értjük.A tárgyhoz tartozó pontok mélysége csak egymáshoz viszonyítva határozható meg, ezért az így mehatározott mélység relatív. A továbbiakban a *mélység* a *relatív mélységet* jelöli.

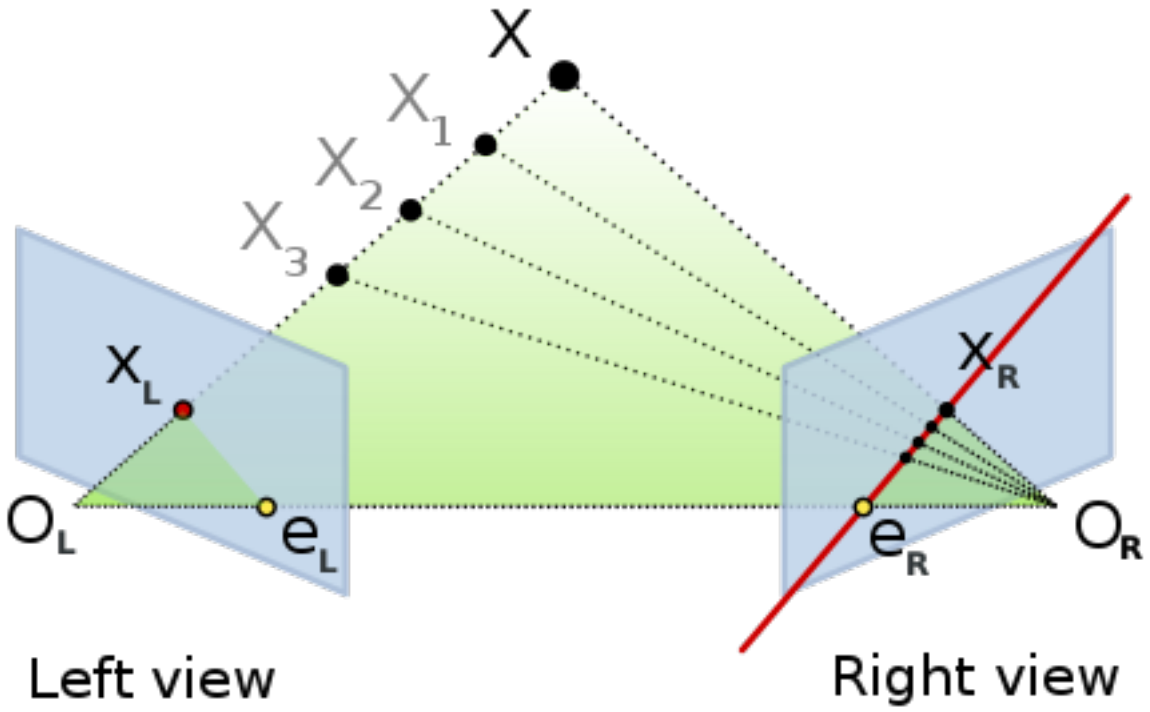
Egy pont mélysége meghatározható a két kép közötti elmozdulása alapján. A két képre általában jobb és bal képként hivatkoznak. Egy pont elmozdulásának meghatározásához szükség van arra, hogy ismerjük annak helyzetét mindkét képen, tehát a képeken szereplő pixelek között léteznie kell egy megfeleltetésnek. Az ugyanolyan elmozdulással rendelkező pontok ugyanahhoz a *mélységi szinthez* tartoznak, vagyis ugyanolyan távol vannak a kamerától. Ismerve több pont mélységi szintjét létre lehet hozni egy *mélységi térképet*.

A pixelek párosításához nagyon fontos az **epipoláris sík** [9] fogalma. Az epipoláris sík megszorításokat hoz be arra vonatkozólag, hogy egy a bal képen megjelenő pont, a jobb képen hol helyezkedhet el.

Tekintsük az X pontot a térben, amely jelen van mind a két képen. A bal képen ismert ennek a helyzete, jelöljük ezt X_L-el, valamint a képhez tartozó kamera középpontját O_L-el. Ezen két pont által meghatározott egyenes tartalmazza az X pontot. Az $O_L X_L$ egyenesnek a jobb képre eső vetületét **epipoláris egyenes**-nek nevezzük.

Ideális esetben egy a bal képen szereplő pontot a fentiek alapján be lehet azonosítani. Az azonosításhoz az *epipoláris egyenesen* lévő pontokat sorban meg kell vizsgálni, hogy azonosak-e a keresett ponttal. Ez azért ideális eset, mert itt pontosan ismertek a sztereó rendszer összes paraméterei (a két kamera *pontos* helyzete, fókusza, a lencse dísztorziós együtthatói és a képek méretarányai).

A sztereó módszer nagyon jól működik megfelelően kalibrált felszerelést használva, hiszen ismertek a fentebb említett paraméterek viszont a Volumiz3D projekt esetében a cél az, hogy a 3D-s rekonstrukciót Android készülék által készített képekkel végezzük el, tehát nem lehet ezek ismeretére számítani.



1. ábra. Az epipoláris egyenes és sík megszorításokat hoznak be a pontok megfeleltetésére vonatkozólag. [20]

1.3. Epipoláris mértan

Ebben az alfejezetben tárgyalásra kerül az 1.2 fejezetben felmerült ismeretlen paraméterek egy részét kiszámoló eljárás. Ezek után be lesz mutatva egy, az említett paraméterekre, és a megfeleltetett pontpárokra építkező folyamat a megfelelő térbeli pontok kiszámítására.

A kamerák ***relatív*** **helyzetének** a kiszámítása **epipoláris mértan** segítségével történik [16]. A két képen szereplő pontmegfeleltetésekből és a kamerák relatív helyzetéből a pontok mélysége kiszámítható **triangulálás** segítségével.

1.3.1. A kamera mátrix fogalma

Az a mátrix, amely minden észlelt térbeli pontnak meghatározza a képét, a kamera mátrix. A $\boldsymbol{P}$ mátrix, tartalmazza a kamera belső paramétereit, orientációját és térbeli helyzetét, alakja:

$$\boldsymbol{P} = \boldsymbol{K}\,[\boldsymbol{R}|\boldsymbol{t}]$$

Az $\boldsymbol{R}$ 3×3 forgatási mátrix a kamera orientációját, a $\boldsymbol{t}$ transzlációs vektor a kamera térbeli helyzetét adja. A $\boldsymbol{K}$ mátrix szintén 3×3 méretű, tartalmazza a kamera f fókuszát, a kép centrumának koordinátáit (p_x, p_y), illetve a kamera dísztorziós paramétereit [16]. Minden $\boldsymbol{X}$ térbeli ponthoz hozzárendelhető annak az $\boldsymbol{x}$ képe, az

$$\boldsymbol{x} = \boldsymbol{P}\boldsymbol{X}$$

összefüggés alapján [16]. A kamera (*projekciós*) $\boldsymbol{P}$ mátrix az úgynevezett fundamentális mátrix segítségével kapható meg.

1.3.2. A fundamentális mátrix fogalma

Az $\boldsymbol{F}$ fundamentális mátrix az epipoláris mértanban azt az összefüggést testesíti meg, amely összeköti egy térbeli pontnak a két képen szereplő képeit, a képeket készítő kamerák helyzetével. Az

$$\boldsymbol{x'}^{\top}\boldsymbol{F}\boldsymbol{x} = 0 \tag{1}$$

kifejezés igaz bármely $\boldsymbol{x}, \boldsymbol{x'}$ esetén, ahol $\boldsymbol{x}, \boldsymbol{x'}$ az $\boldsymbol{X}$ térbeli pont bal, illetve jobb oldali képe.

Az $\boldsymbol{F}$ kiszámítható több bal és jobb oldali pontmegfeleltetések ($\boldsymbol{x} \leftrightarrow \boldsymbol{x'}$) ismertével. Feltéve, hogy a pontok zajmentesek, az eredményt a fentebbi egyenlet révén kapjuk, $\boldsymbol{F}$ -et kifejezve. Valós helyzetekben a zajmentesség nem teljesül, ilyenkor az $\boldsymbol{F}$ egy megközelítését kaphatjuk meg. Az $\boldsymbol{F}$ megközelítésére több numerikus módszer létezik (*8 pont algoritmus*, *RANSAC*), részletes leírás a "Richard Hartley és Andrew Zisserman" [16] könyvben található.

Az $\boldsymbol{F}$ nem tartalmazza önmagában explicit módon a két kamera helyzetét, orientációját, ezek további kiszámolásra szorulnak [16].

Normalizált kameramátrix, esszenciális mátrix.

A $\boldsymbol{P}$ azon formája, amelyben a $\boldsymbol{K}$ kalibrációs mátrix egyenlő az egységmátrixszal, a **normalizált kameramátrix**,

$$\boldsymbol{P} = [\boldsymbol{R}|\boldsymbol{t}]$$

alakú. Az ilyen típusú kameramátrixokhoz rendelt fundamentális mátrix, az **esszenciális** mátrix, kiszámítható az

$$\boldsymbol{E} = \boldsymbol{K'}^{\top}\boldsymbol{F}\boldsymbol{K}$$

összefüggéssel [16].

Adott $\boldsymbol{E}$ esszenciális mátrix esetében a bal illetve jobb oldali kameramátrixok kiszámíthatóak. A bal oldali normalizált kameramátrix minden esetben felírható a

$$\boldsymbol{P} = [\boldsymbol{I}|0]$$

alakban [16]. A jobb oldali kameramátrix normalizált alakja az $\boldsymbol{E}$ mátrix SVD [1] felbontásából kapható meg. Az $\boldsymbol{E}$ -nek több SVD felbontása is van. A felbontások közül csakis egy biztosítja a térbeli pontok azon tulajdonságát, hogy mindkét kamera előtt helyezkednek el. A $\boldsymbol{P'}$ jobb oldali kameramátrix ebből a felbontásból kapható meg. A felbontás precíz kifejtése a "Richard Hartley és Andrew Zisserman" [16] könyvben található.

subsubsection*Összefoglaló

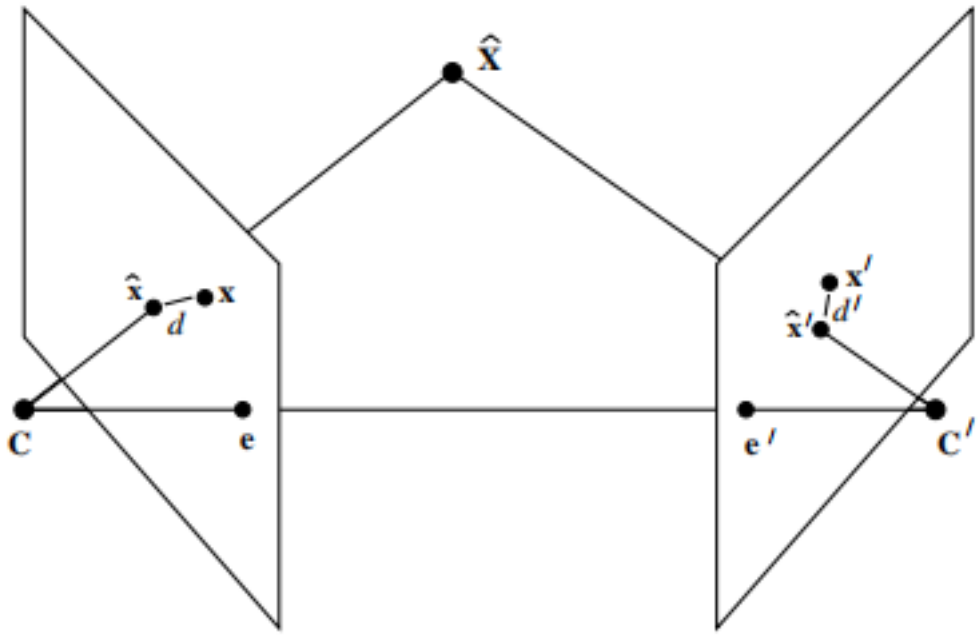
Az alfejezet eddigi részében a kamerák helyzetét kiszámító eljárás lett bemutatva. Ezen eljárás a képeken szereplő pontok párosítását veszi alapul, így becsüli meg a fundamentális és esszenciális mátrixot, majd a megfelelő kameramátrixokat. Az eljárás feltételezi a kamerák belső paramétereinek ismeretét ($\boldsymbol{K}$).

1.3.3. Triangulálás

Ebben a részben az az eljárás lesz bemutatva, amely a pontpárokhoz rendelt térbeli pontokat eredményezi. Az eljárás során ismertek a $\boldsymbol{x} \leftrightarrow \boldsymbol{x'}$ bal és jobb oldali képen található pontok közötti megfeleltetések, illetve a képeket készítő kamerák projekciós mátrixai ($\boldsymbol{P}$). A képeken szereplő pontok nem zajmentesek, ennek köszönhető az, hogy a kiszámolt térbeli pontok csupán megközelítések [16].

A bal oldali képeken található pontok $\boldsymbol{x}_i$-vel, illetve a jobb oldalon található pontok $\boldsymbol{x}'_i$-el vannak jelölve. Abból, hogy a pontok zajosak az következik, hogy ezek nem feltétlenül teljesítik

[1] Singular Value Decomposition



2. ábra. Triangulálással az $\hat{x}_i$, $\hat{x}_i'$ pontokból meghatározható az $\hat{X}_i$ térbeli pont. A kép a "Richard Hartley és Andrew Zisserman" [16] könyvből származik.

az (1)-es összefüggést. Ha ismertek az új $\hat{x}_i$, illetve $\hat{x}_i'$ becsült pontok, amelyek ezt teljesítik, és legközelebb vannak az eredeti megfelelőjükhöz, akkor kiszámíthatóak a hozzájuk rendelt térbeli pontok [16].

Az $\hat{X}_i$ pontok a kamerák középpontja, és a becsült pontokból fakadó egyenesek metszeteiből meghatározhatóak. Az (1)-es összefüggés garantálja azt, hogy ezen egyenesek, találkoznak egy térbeli pontban [16].

Az $\hat{x}_i$ illetve $\hat{x}_i'$ pontpárok becslése.

Az epipoláris sík (π) meghatározza a képeken található epipoláris egyenespárokat. Minden epipoláris egyeneshez hozzárendelhető egy egyedi, az egyenesen levő, az x_i-hez legközelebb eső pont. Ez az $\hat{x}_{i_\pi}$, illetve $\hat{x}'_{i_\pi}$ a jobb oldali esetben. Minden epipoláris egyenespáron levő pontpár teljesíti az (1)-es összefüggést. Az epipoláris sík paraméterezi ezeket a pontokat, tehát a triangulálás az

$$\underset{\pi}{\operatorname{argmin}} \left[d(\boldsymbol{x_i}, \hat{\boldsymbol{x}}_{i_\pi}) + d(\boldsymbol{x_i'}, \hat{\boldsymbol{x}}'_{i_\pi}) \right]$$

célfüggvényt minimalizáló epipoláris sík keresésére redukálódik, minden $x_i \leftrightarrow x_i'$ pontpár esetén [16]. A d euklideszi távolságot jelöl. A keresett π sík szabadságfoka egy, ugyanis a két kamera középpontja által meghatározott egyenest tartalmaznia kell. A minimalizáló síkot numerikus módszerekkel is ki lehet számolni, de direkt módon is. Ezek a módszerek a "Richard Hartley és Andrew Zisserman" [16] könyvben részletesen tárgyalva vannak.

Összefoglaló

Ebben az alfejezetben szó esett a kamerák helyzetének a meghatározásáról kizárólag pontpárosításokat és a kamera belső paramétereit ismerve (*fókusz,képközéppont*. . .). Továbbá a triangulálás során a pontpárosokhoz rendelt térbeli pontok is ki lettek számolva.

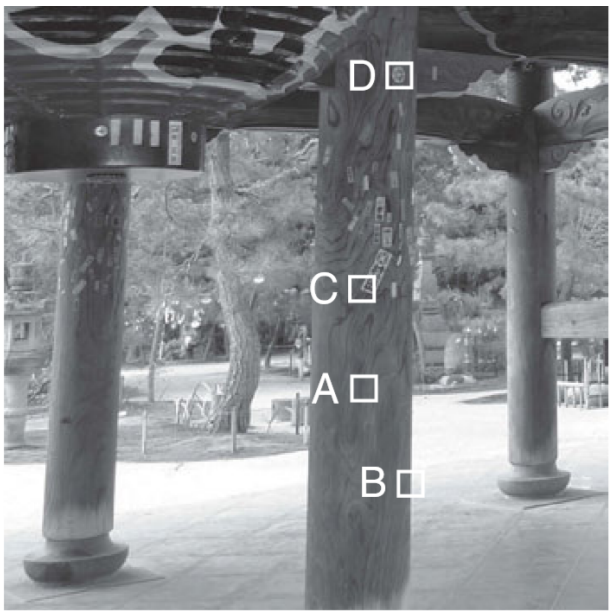
1.4. Kulcspontok

Korábban már szó esett a pixelek képek közötti megfeleltetéséről, viszont nem minden pont megfelelő választás erre, mivel általában nagyon sok hasonló pixel található egy-egy képen. Ebben a részben arról lesz szó, hogy hogyan keressünk olyan helyeket egy képen, amelyek megbízhatóan megfeleltethetőek. Ezeket a speciális helyeket hívjuk **kulcspont**oknak. Ezek után szó esik arról, hogy hogyan tudjuk őket leírni, megfeleltetni és akár több képen keresztül is felhasználni.

Egy kulcspontot, meghatároz egy pixelekből álló négyzet a képen, amelynek a középpontja lesz a kulcspont helyzete. Az egyik fő gondolat az, hogy mitől lesz egy kulcspont megfelelő. A legfontosabb karakterisztikája egy kulcspontnak az, hogy ugyanaz a kulcspont bármilyen képen felismerhető legyen, függetlenül a nézőpontoktól, disztorziótól vagy akár a fényviszonyoktól is.

A jó és rossz kulcspontokat a 3. ábra szemlélteti. Ki van emelve rajta négy potenciális kulcspont.

- Az "A" az egyik legrosszabb jelölt az összes közül, mivel nincs benne semmi textúra. Ez a négyzet szinte bárhol lehetne az oszlopon és akkor is ugyanolyan lenne.
- A "B" az előbbinél kicsivel jobb, mivel a közepén a fény intenzitása megváltozik, emiatt megállapítható, hogy az oszlop szélén helyezkedik el, viszont ez még mindig nem elég pontos.
- A "C" már megfelelő választás lehet mivel látható rajta a mintázatnak a sarka, és ettől egyedi lesz, mivel nincs még egy ilyen hely a képen.



3. ábra. Négyzetekkel jelölt potenciális kulcspontok egy képen. [28]

- A "D" szintén megfelelő, mivel egy szabálytalan foltot jelöl, ami alapján könnyen megkülönböztethető a kép többi részétől.

1.4.1. Deszkriptorok

A kulcspontokat egy számokból álló vektorral írjuk le, ezek a **deszkriptor**ok. A deszkriptorjaik alapján tudjuk párosítani a kulcspontokat különböző nézőpontból készített képek között, tehát úgy kell ezeket felépíteni, hogy egy tárgynak egy adott részéhez tartozó pontnak ugyanaz, vagy nagyon hasonló deszkriptorja legyen bármilyen nézőpontból. A cél az, hogy teljesüljön a $D(f_1) \approx D(f_2)$ összefüggés, ahol D egy deszkriptort létrehozó algoritmus, és az f_1,f_2 megtalált kulcspontok különböző képeken, úgy, hogy $\exists T$, affin transzformációk összessége és $f_1 = T f_2$.

Egy egyszerű deszkriptor példa az lenne, hogy elhelyezzük egy vektorba az összes kulcspont körül lévő pixel intenzitását. Ilyen módon összehasonlíthatók a deszkriptorok a megfelelő elemek között négyzetes különbségek alapján. Ez a megközelítés működőképes, amikor a vizsgált képek között nem nagy az eltérés, például ha egy videónak két közel egymás utáni képéről van szó. Viszont ez a deszkriptor elbukik az olyan helyzetekben, mikor a képek el vannak forgatva, különbözőképpen vannak skálázva, különböznek a fényviszonyok, vagy más szögből vannak készítve, lásd a 4. ábrát.

4. ábra. (a) Két különböző perspektívából készített kép ugyanarról az épületről. Egy négyzet jelöli az általunk vizsgált helyet. (b) Ráközelítve ezekre a négyzetekre látható hogy ezek különbözőek. [28]

1.5. AKAZE

A hasonló képfeldolgozási feladatok megoldásához több ismert kulcspont felismerő és párosító algoritmust is kifejlesztettek. Ezek közül az elterjedtebbek a **SIFT** és a **SURF** amelyek ezeket a műveleteket nagyon gyorsan (akár valós időben) el tudják végezni. Ezt a sebeséget úgy érik el hogy Gauss-zajt visznek rá a képekre, viszont ennek a hátránya az hogy emiatt az apróbb részletek elveszhetnek, és ez ront a megismételhetőségen és a robusztusságon is. Ezeknek a problémáknak a kiküszöbölésére fejlesztették ki a **KAZE** algoritmusokat, amelyek ezeket több nemlineáris szűrő használatával kezelik. Ennek viszont az a hátulütője, hogy a sokkal több erőforrást igényel a futtatása. Ahogy a technológia fejlődött, és új képfeldolgozási módszerek lettek kifejlesztve, mint például a **FED** (Fast Explicit Diffusion) [11]. A KAZE által használt nemlineáris szűrőket kicserélve ezekre sikerült egy elég jó egyensúlyt létrehozni a sebesség és megbízhatóság szempontjából, igy jöttek létre az **AKAZE** (Accelerated-KAZE) kulcspontok. [24]

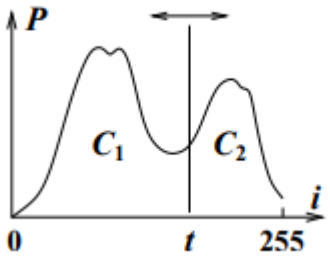
Kimutatták [1], hogy a jelenlegi általános algoritmusok közül az AKAZE a legrobusztusabb a forgatásra és a skálázásra való tekintettel. Ezen felül ha teljes képeket akarunk megfeleltetni, akkor ez a leggyorsabb a robusztus módszerek közül, fölülmúlva a fentebb említett módszereket (SIFT, SURF, KAZE). Ennek a figyelembevételével az alkalmazásban az AKAZE által nyújtott megfeleltetési algoritmussal és deszkriptorokat használtuk.

1.6. Szegmentálás

Az alkalmazásunk előfeltételei között szerepelt az, hogy a vizsgált tárgy egy homogén felületen helyezkedik el. Ennek érdekében a zaj és más esetleges hibák kiszűrése érdekében szegmentálás lesz végrehajtva a képeken a feldolgozás kezdetén.

Szegmentálás [25] alatt a kép pixeleinek különböző szegmensekre (csoportokra, halmazokra) bontása értendő. A szegmentálás eredménye, annyi pixelhalmaz, amennyire a képet szegmentáltuk, illetve a halmazokat elválasztó határvonalak. A Volumiz3D esetében a szegmentálás során két halmazra szeretnénk bontani a pixeleket, ahol az egyik a vizsgált tárgy, a másik pedig a háttér. Azt a szegmentálást, amely két csoportot hoz létre, bináris szegmentálásnak vagy binarizálásnak nevezik.

A legelterjedtebb módszerek felépítenek egy hisztogramot a képen megtalálható pixelek intenzitásából és az alapján próbálják csoportosítani. A projektben az **Otsu módszer** [7] lett alkalmazva. Az Otsu módszer elve megfigyehető az 5. ábrán. Miután létrehoztuk a képünk intenzitás hisztogramját, annyi a feladatunk, hogy úgy változtatjuk a t küszöbértéket, hogy a C_1 és C_2 szegmensek optimálisak legyenek, azaz a csoportban levő pontok szórása minimális legyen. A nagy előnye ennek a módszernek hogy ez az egyik leggyorsabb, és garantált hogy mindig ad egy eredményt.

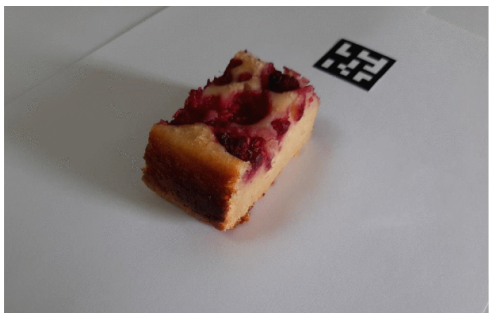


5. ábra. Az Otsu módszer által létrehozott hisztogram, a C1 és C2 a szegmensek, t a küszöbérték amit megakarunk határozni. [7]

Mivel tudjuk azt, hogy az általunk feldolgozott képen egy darab vizsgált tárgy lesz, ezért feltételezhetjük, hogy a számunkra érdekes terület egy zárt egységen belül helyezkedik el. A feltételezésnek a teljesítéséhez a szegmentálás eredményére "zárás"-t alkalmazunk. A zárás több morfológiai művelet összessége [8], és a hatása az, hogy kitölti a keskeny csatornákat és a kis lyukakat (példa a 6. ábrán).

6. ábra. Egy írott "j" karakterről létrehozott bináris kép zárása. (closing)[Forrás]

A 7. ábrán látható képet feldolgozzuk ennek a két módszernek az alkalmazásával. Az Otsu módszer által létrehozott szegmentálás eredményén zárást alkalmazunk addig, amíg egy összefüggő zárt szegmens jön létre. Ennek az eredménye megfigyelhető a 8. ábrán.



7. ábra. Sütemény aminek a térfogatát szeretnénk megállapítani.

8. ábra. A baloldali kép szegmentált verziója. A fekete rész a háttér, a fehér rész amit tovább feldolgozunk.

1.7. Pontfelhők és azok egyesítése

Ebben a fejezetben a triangulált pontokból keletkező pontfelhők egyesítésére vonatkozó eljárás lesz bemutatva.

1.7.1. Motiváció

Két kép alapján lehetetlen egy tárgyat rekonstruálni, hiszen ez nem látszik teljességében, ezért felmerül a több nézőpontból készített képek szükségessége. Ahhoz, hogy egy tárgy minden oldala rekonstruálva legyen, legalább egyszer körbe kell fényképezni azt. Minden képpárból egy-egy pontfelhő kiszámolható, az előző alfejezetek alapján. Az így kapott pontfelhők egymásra illesztése után a tárgy minden oldala modellezve van.

1.7.2. Pontfelhők regisztrálása

A képpárokból kapott pontfelhők nem egy egységes koordináta-rendszerben vannak elhelyezve, hiszen a képpárok egy skála erejéig határozhatják meg a fundamentális mátrixukat [16]. Következésképpen ezek a felhők nem illeszkednek egymáshoz, el vannak egymástól forgatva. Regisztráció alatt a pontfelhők azon hasonlósági transzformációinak $(\boldsymbol{S_i})$ [2] kiszámítása értendő, amelyek a felhőket a lehető legpontosabban[3] egybeforgatják. Az eltérő koordináta-rendszereknek köszönhetően nem megfelelő az a megközelítés, amely a pontfelhőket a kiszámolt kameramátrixok forgatási és eltolási részének láncolt összefűzésével regisztrálná.

9. ábra. Pontpárosítások

10. ábra. Triangulált pontfelhő

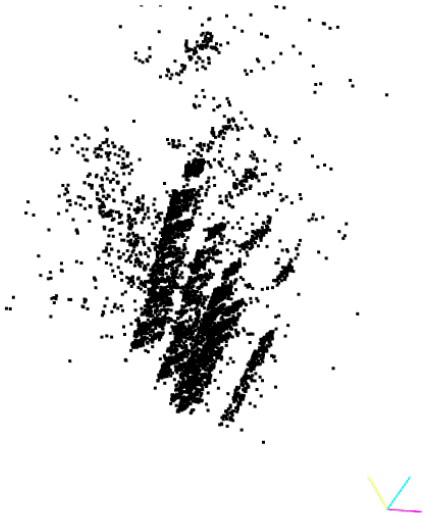
[2]Egy $\boldsymbol{S} = s[\boldsymbol{R}|\boldsymbol{t}]$ hasonlósági transzformáció forgatásból, eltolásból és skálázásból áll

[3]Pontosság alatt a pontfelhőkben szereplő pontpárok térbeli közelsége értendő.

1.7.3. Hasonlósági transzformációk becslése

Az $\boldsymbol{S_i}$ transzformációk kiszámítására a pontfelhők olyan közös attribútumai használhatóak fel, amelyek ezekre invariánsak. A pontfelhőkben levő pontok, amelyek közös képbeli kulcspontok alapján triangulálódnak, azonosnak tekinthetőek. Azonos $\boldsymbol{X_i}$ pontok $\boldsymbol{S}$-től függetlenül, ideális esetben ugyanoda, valós esetben egymáshoz közel kell kerüljenek a regisztráció folyamán. Egy másik, a hasonlósági transzformációra invariáns tulajdonság, a pontok közötti távolság aránya. Feltéve, hogy két pontfelhő között ismertek $\boldsymbol{X_i} \longleftrightarrow \boldsymbol{X_i'}$ azonos pontok, a két felhő közötti $\boldsymbol{S}$ transzformáció s skálázási tényezője meghatározható a pontok közötti távolságok $s = \dfrac{d(\boldsymbol{X_i}, \boldsymbol{X_{i+1}})}{d(\boldsymbol{X_i'}, \boldsymbol{X_{i+1}'})}$ hányadosaként. A zajnak köszönhetően nem elegendő egyetlen arányt figyelembe venni, több pont jobb megközelítést adhat, következésképpen jobb az $s = \dfrac{1}{n}\sum_{i}^{n} \dfrac{d(\boldsymbol{X_i}, \boldsymbol{X_{i+1}})}{d(\boldsymbol{X_i'}, \boldsymbol{X_{i+1}'})}$ várható érték. Az s_i skálatényezők kiküszöbölése után a regisztráció felírható olyan $[\boldsymbol{R}|\boldsymbol{t}]$ transzformációk kereséseként, amelyek az $\boldsymbol{X_i} \longleftrightarrow \boldsymbol{X_i'}$ azonos pontok közötti távolságot minimalizálják.

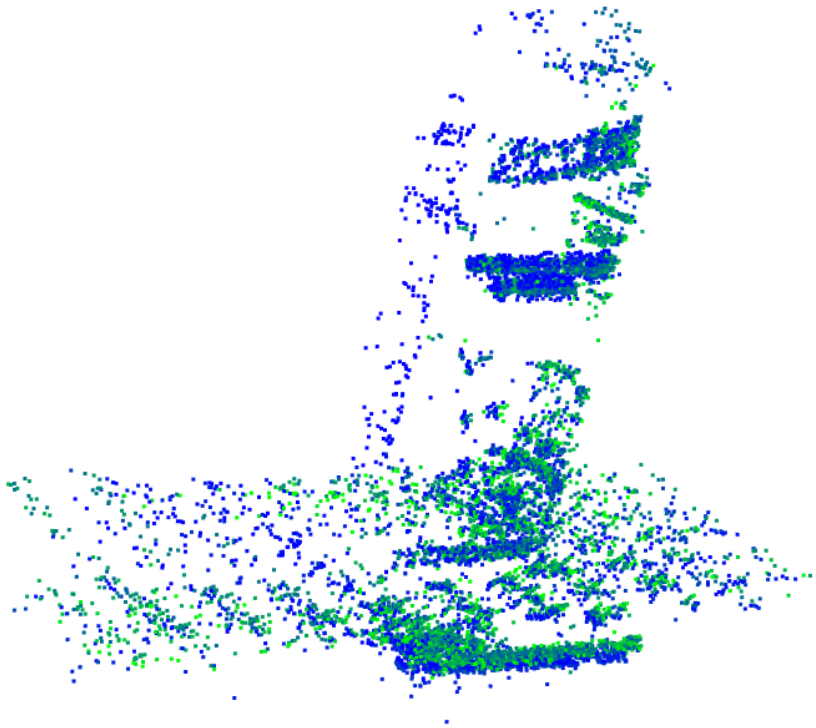
A **Random Sample Consensus (RANSAC)** módszer egy túlhatározott rendszer paramétereit képes megbecsülni. A rendszer előnye, hogy jól működik akkor is, ha a rendszerben jelentős mennyiségű zaj van jelen, más módszerekkel ellentétben nem a megfigyelések egyszerre való figyelembevételével végez paraméterbecslést. A **RANSAC** egy iteráció alapú módszer, minden iterációban véletlenszerűen kiválasztásra kerül m darab, a modell paramétereinek megbecsléséhez szükséges minimális számú megfigyelés, majd ezekből megkapható a modell egy $\boldsymbol{T_i}$ becslése. Ezek után a becsült modell minősége[4] van kiszámolva.



11. ábra. Pontfelhők skála és koordináta-rendszerbeli különbsége.

[4]Minőség alatt az olyan megfigyelések száma értendő, amelyek illeszkednek a modellhez, egy *t* küszöbhatáron belül.

N darab iteráció után eredményként a legminőségibb modell kerül kiválasztásra [16]. Jelen esetben, a módszer egy $\boldsymbol{T} = [\boldsymbol{R}|\boldsymbol{t}]$ euklideszi transzformáció paramétereit kell megbecsülje. A $\boldsymbol{T}$ transzformáció szabadságfoka 6, hiszen mind az $\boldsymbol{R}$ úgy a $\boldsymbol{t}$ 3 szabadságfokú [5]. $\boldsymbol{X_1}, \boldsymbol{X_2}, \boldsymbol{X_3}$ pontok szabadságfoka összesen 9, viszont ez az érték 6-ra csökken ha a pontok relatív helyzete meg kell egyezzen azok $\boldsymbol{X_1'}, \boldsymbol{X_2'}, \boldsymbol{X_3'}$ megfelelőivel. A pontok közötti távolsági megszorítások mind egy-egy szabadságfokvesztéssel járnak [6]. Ezekből kifolyólag 3 pontpárból, amelyek általános helyzetben vannak (nem *kollineárisak*) ideális esetben ki lehet számolni a $\boldsymbol{T}$ transzformációt. Valós esetben a pontpárok relatív helyzete nem azonos, ebből kifolyólag a $\boldsymbol{T}$ transzformáció csak megközelíthető. A megközelítésre a legkisebb négyzetek módszere használható [3]. Minden megbecsült $\boldsymbol{T_i}$ transzformáció esetében ki kell számolni annak a minőségét, ez a hasonlósági transzformációk minden pontpár esetén való elvégzésével, és az elfogadható közelségbe került pontpárok megszámlálásával történik. Egy pontpár akkor van elfogadható közelségben, ha a $d(\boldsymbol{X_i}, \boldsymbol{X_i'})$ távolság egy adott t küszöbhatár alatt van [16]. N iteráció után a becslések közül a legminőségibb lesz elfogadva eredményként.



12. ábra. Pontfelhők illesztése RANSAC módszerrel.

[5] Az $\boldsymbol{R}$ szabadságfoka intuitívan 9 lenne, viszont nem minden mátrix rendelkezik a forgatási mátrix tulajdonságaival [32].

[6] $\boldsymbol{X_1}$ bárhol elhelyezkedhet a térben, tehát 3 szabadságfokú, $\boldsymbol{X_2}$ pont $\boldsymbol{X_1}$-től $d(\boldsymbol{X_1'}, \boldsymbol{X_2'})$ távolságra lehet, ebből következik, hogy szabadságfoka 2, hiszen az $\boldsymbol{X_1}$ körüli, $d(\boldsymbol{X_1'}, \boldsymbol{X_2'})$ sugarú gömbön helyezkedhet el, $\boldsymbol{X_3}$ szabadságfoka 1, mert $\boldsymbol{X_1}$-től $d(\boldsymbol{X_1'}, \boldsymbol{X_3'})$, $\boldsymbol{X_2}$-től $d(\boldsymbol{X_2'}, \boldsymbol{X_3'})$ távolságra lehet.

Az iterációszám meghatározása

N-et szükséges úgy meghatározni, hogy egy adott p valószínűséggel[7] az iterációk közben legalább egy olyan kiválasztásra kerüljön sor, amelyben minden párosítás hibamentes [16]. Egy párosítás hibamentes, ha a párosított pontok tényleg azonosak. Az

$$N = \frac{\log(1-p)}{\log(1-(1-e)^m)}$$

összefüggéssel[8] meghatározható ez az iterációszám [16].

Összefoglaló

Ebben az alfejezetben a pontfelhők egy skálára való hozása, illetve egymáshoz való illesztése volt tárgyalva. A felhők közötti s skálakülönbség azonos pontok távolságának arányával, az egybeillesztést elvégző $\boldsymbol{S}$ hasonlósági transzformációk Random Sample Consensus (RANSAC) módszerrel lettek kiszámolva.

[7] Általában $p = 0.99$

[8] Az e annak a valószínüsége, hogy egy adott párosítás hibás.

2. Korszerű módszerek

Ebben a fejezetben tárgyalva lesznek a korszerűbb rekonstrukciós technikák újításai, és ezeknek a szükségessége.

2.1. Bundle adjustment

Az előző fejezetben szereplő módszerek ideális, és kicsi adathalmazok esetén megfelelően működnek, viszont a sok képből fakadó hibák kumulatív természete[9] egy jobb, hibákra robusztusabb megközelítést követel meg. A fejezet további része a "Noah Snavely, Steven M. Seitz, Richard Szeliski" [29] cikkét veszi alapul. A szakirodalomban a **bundle adjustment** egy széles körben elterjedt módszer kép alapú térbeli rekonstrukció kivitelezésére. A rekonstrukciót fel lehet írni egy olyan optimalizálási feladatként, amelyben a távolság a térbeli pontok képe és visszavetítése között a minimalizálandó célfüggvény. A célfüggvényt a kamerák mátrixai, és a térbeli pontok koordinátái paraméterezik. Egy $\boldsymbol{P}$ projekciós mátrix általánosan 11 szabadságfokú, illetve 7 ha teljesülnek azok a megkötések melyek szerint egy képet alkotó pixelek négyzet alakúak, illetve a visszavetítési központ megegyezik a kép középpontjával. A kamerák túlnyomó részében teljesülnek ezek a feltételek. Jelen esetben egy kamerát a térbeli helyzete (3 paraméter), az orientációja (3 paraméter), illetve a fókusztávolsága (1 paraméter) határoz meg. Minden pontot 3 paraméter jellemez, az *x, y* és *z* koordinátája.

Az eddig ismert $\boldsymbol{x} \leftrightarrow \boldsymbol{x'}$ kulcspontpárok általánosítása a **pontpálya**. Egy pontpálya egymást követő képeken szereplő ($\boldsymbol{x^i} \leftrightarrow \boldsymbol{x^{i+1}} \leftrightarrow \cdots \leftrightarrow \boldsymbol{x^{i+k}}$) kulcspontpárokból áll. A továbbiakban a kamera paramétereiből álló vektor $\boldsymbol{\Theta}$-val, egy térbeli pont *p*-vel, a j-edik pontpálya i-edik kamerán látható kulcspontja q_{ij}-vel lesz jelölve. Abban az esetben, ha n képből pontpárosítással m pontpálya meghatározható, a minimalizálandó célfügvény

$$f\left(\Theta, p\right) = \sum_{i}^{n} \sum_{j}^{m} w_{ij} \left\| q_{ij} - P(\Theta_i, p_j) \right\|^2 \tag{2}$$

alakú. A w_{ij} az a függvény, amelyik az 1-es értéket veszi fel abban az esetben, ha az i-edik kamerán látható a j-edik pont, különben 0. $P(\Theta_i, p_j)$ az a függvény, amelyik a j-edik pontot az i-edik kamera képére vetíti vissza, majd az így kapott homogén koordinátát átalakítja képbeli koordinátává. Azért szükséges az átalakítás, hogy a fókusz kiiktatásával a visszavetítés összehasonlítható legyen a megfelelő q_{ij} kulcsponttal.

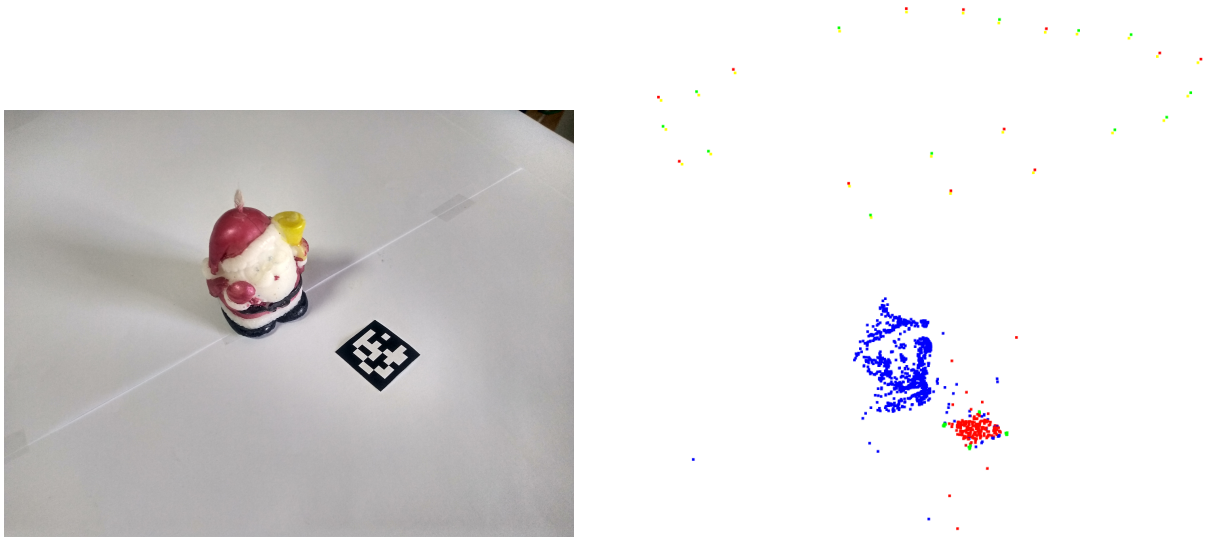
A rekonstrukció f-ben egy nemlineáris minimalizálási problémaként van felírva, az ilyen típusú problémák Levenberg-Marquadt módszerrel minimalizálhatóak. Bővebb leírás a

[9] Az $\boldsymbol{S}$ hasonlósági transzformációk egymás után való kiszámítása a hibák propagálásával jár, mert minden transzformáció kizárólag az azelőttihez igazodik. Egy hibás transzformáció hatással van az összes utánalevőre.

"Nocedal, S.Wright" [22] könyvben található.

Bundler

A Bundler a "Noah Snavely, Steven M. Seitz, Richard Szeliski" [29] által implementált szoftvercsomag a bundle adjustment kivitelezésére. A Levenberg-Marquadt módszer [22] lokális minimumokat talál meg, ezért szükséges a kezdeti paraméterek egy jó becslése. A kezdeti becslés a Bundler esetében az optimalizálási feladat részleges megoldásával, majd a megoldás folyamatos kibővítésével történik. A legjobb potenciális kezdeti becslést a legtöbb kulcspontpárosítást tartalmazó kamerapár rekonstrukciója adja. Ezt követően egy újabb kamera lesz az optimalizálási feladathoz hozzáadva. Az a kamera kerül kiválasztásra, amelyik a legtöbb rekonstruált pontpályát észleli. Az új kamera kezdeti paramétereire való becslés a már tárgyalt **RANSAC** módszerrel történik [16]. Ezek után a kibővült célfüggvény újra minimalizálva lesz. A pontpályák kibővülnek a jelenlegi kamera pontpárosításaival. Az optimalizálás az összes kamera hozzáadásával és részfeladatok minimalizálásával végződik. A minimalizálás egy a Google által fejlesztett optimalizálási könyvtár, a ceres-solver [2] segítségével történik, amelyik szintén a Levenberg-Marquadt [22] módszeren alapul.



13. ábra. Bundle adjustment-el rekonstruált pontfelhő. A felhő fölötti pontok a kamerák helyzeteit láttatják.

2.2. Patch-based Multi-view Stereo

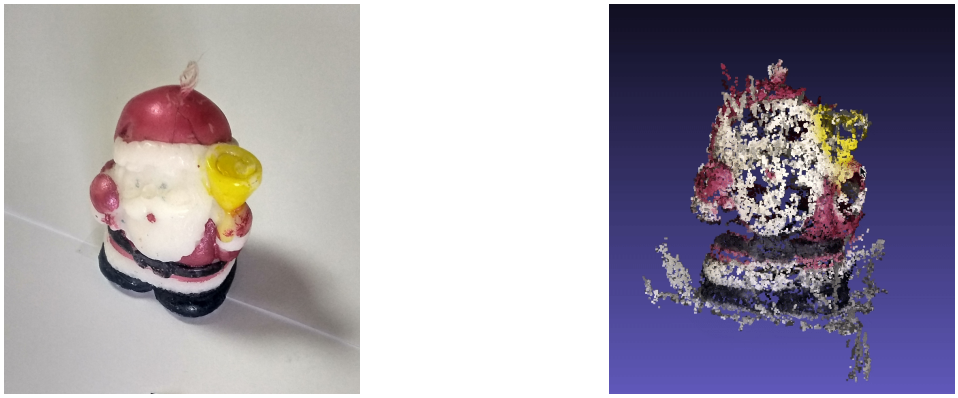
Ebben az alfejezetben az a módszer lesz bemutatva, amelyik rekonstruált kamerák alapján egy sűrűbb, irányított, jobb minőségű pontfelhőt eredményez. A Patch-based Multi-view Stereo módszert "Yasutaka Furukawa és Jean Ponce" [12] fejlesztették, a fejezet további része az ő eredményeik bemutatása.

A **PMVS** feltételezi a kamerák paramétereinek ismeretét, ezek az ezelőtti alfejezetben tárgyalt Bundle-adjustment módszerrel meghatározhatóak. A módszer nem használja az ezelőtti pontfelhő pontjait, újakat számol ki. A kamerák relatív helyzeteinek ismeretében a kulcspontpárosítás egyszerűbbé válik, hiszen a pontok a kiszámolt epipoláris vonalakon keresendőek. Következésképpen olyan kulcspontok is használhatóak, amelyek nem teljesen egyediek. A pontpárosításokra az eddig használt **AKAZE**, **SIFT** típusú kulcspontok helyett **Harris** sarkok vannak használva. A módszer által generált pontfelhő sűrűsége annak köszönhető, hogy sokkal több **Harris** sarok típusú kulcspont található a képeken, mint **AKAZE** vagy **SIFT**, illetve a felhő a kezdeti pontok szomszédaival is bővül.

A módszer sajátossága, hogy a tárgy felületét apró, négyzet alakú **foltokkal** közelíti meg, majd a foltot alkotó térbeli pontokat ellátja a folt irányvektorával. Minden pont irányvektora meghatározza a tárgy felületét abban a pontban érintő síkot. A **PMVS** a következő három lépésből áll.

- *Párosítás*: A **Harris** sarkok párosításából a kezdeti foltok meghatározása.
- *Kiterjesztés*: A kezdeti foltok a környező pixelekkel való bővülése.
- *Szűrés*: A helytelen párosítások eltávolítása.

A módszer folyamán a kiterjesztés, szűrés a kezdeti párosítás után többször ismételve van. A módszer részletesebb bemutatása a "Yasutaka Furukawa és Jean Ponce" [12] cikkben található.



14. ábra. PMVS által létrehozott sűrű pontfelhő

2.3. Poisson felületek

Ebben az alfejezetben egy a tárgyra illeszkedő felület létrehozásáról lesz szó. Egy pontfelhő ismeretében, egy arra illeszkedő felületet több módszerrel is meg lehet határozni (*Delaunay trianguláció [10], Marching cubes [21]*). Az említett módszerek irányítás nélküli pontfelhők esetében használtak, a pontok irányítását ismerve létezik egy jobb megközelítés. "Michael Kazhdan, Matthew Bolitho és Hugues Hoppe" [17] munkájában egy irányított pontfelhő felületét megközelítő eljárás van bemutatva, az ő általuk kapott felület Poisson típusú, hiszen egy térbeli Poisson feladat megoldásaként kapható meg. Ez a fejezet az ő munkájukat bemutatva ismerteti egy felület létrehozását.

A feladat reprezentációja

A módszer a felületet implicit alakban reprezentálja egy χ indikátorfüggvény segítségével. A $\chi(p)$ függvény értéke 1, ha a p pont a modell belsejében található, illetve 0, ha azon kívül. Következésképpen a felületet az indikátorfüggvény 0 és 1-es értékek közötti határvonala képezi. Minden $s \in S$ felhőbeli pont a felület egy mintavétele, tehát kötelező módon a keresett χ függvény határvonalának tartalmaznia kell ezeket a pontokat.

Poisson feladat

A megfelelő indikátorfüggvények halmazát tovább szűkíti az a megszorítás, miszerint a pontok irányai egyben a keresett felület normálvektorai is, illetve az, hogy ezen normálvektorok változása megegyezik a mintavételbeli irányok változásával. A normálvektorok egyenlőségére vonatkozó megszorítás a

$$\nabla\chi(p) = \vec{V}(p) \tag{3}$$

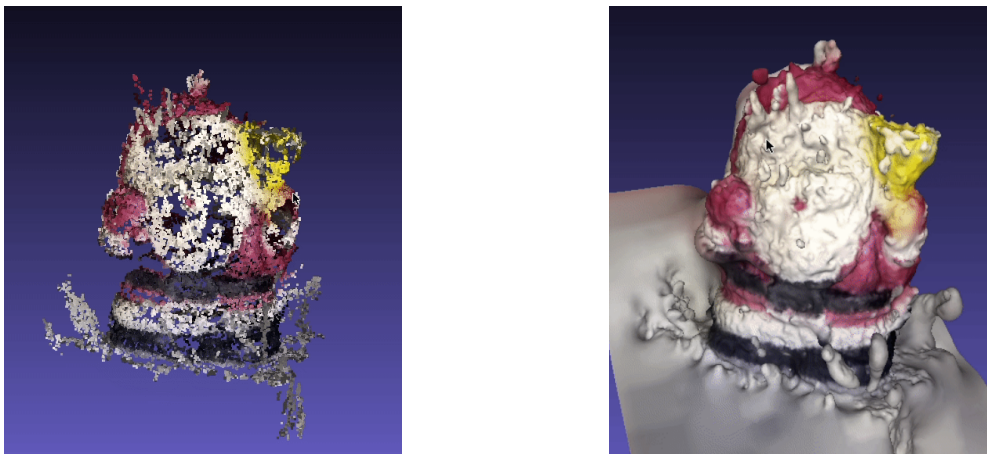
alakban a χ gradiense [10] és a pontfelhő irányai által alkotott vektortér egyenlőségeként formalizálható. Továbbá a

$$\Delta\chi \equiv \nabla\nabla\chi = \nabla\vec{V} \tag{4}$$

egyenlettel [11] megfogalmazható az irányok változására vonatkozó megszorítás. A térbeli Poisson feladat tehát azon χ indikátorfüggvény meghatározása, amelyik teljesíti a (3)-as, és a (4)-es összefüggéseket minden $s \in S$ felhőbeli pontban. A "Michael Kazhdan, Matthew Bolitho és Hugues Hoppe" [17] által írt cikkben megtalálható a feladat octree segítségével való diszkretizálása és megoldása.

[10] A gradiens (∇) operátor egy függvény parciális deriváltjaiból álló vektort adja egy adott pontban. Felfogható a legnagyobb elmozdulás irányaként is.

[11] A Laplace-operátor (Δ) egy függvény másodrendű parciális deriváltakból álló vektort adja meg egy adott pontban. Felfogható a gradiensek változásaként.



15. ábra. Pontfelhőhöz illesztett Poisson felület.

Összefoglaló

Ebben a fejezetben olyan korszerű módszerek lettek bemutatva, amelyek jó minőségű pontfelhőket és felületeket eredményeznek, robusztus módon. A **Bundle-adjustment** a kamerák paramétereit, és egy pontfelhőt határoz meg egy optimalizálási feladat folyamatos minimalizálásával. Ezt követően a **PMVS** az ismert kamerák alapján egy új, sűrűbb, irányított pontfelhőt eredményez, amihez egy **Poisson felület** illeszthető a felhő pontjai irányításának figyelembevételével.

3. Térfogatbecslés

Ebben a fejezetben a tárgy zárt modelljének a létrehozásáról, relatív térfogatának kiszámításáról, illetve ennek a térfogatnak a metrikus skálára való hozásáról lesz szó.

3.1. Zárt felület térfogatának a kiszámítása

Egy általános A, B, C, D csúcspontú tetraéder térfogata a $V_{A,B,C,D} = \frac{1}{6}|\overrightarrow{AB} \cdot (\overrightarrow{AC} \times \overrightarrow{AD})|$ összefüggéssel kiszámítható. Egy zárt felület térfogata az azt alkotó n darab felületi háromszög, és az origó által alkotott tetraéderek előjeles

$$V_M = \frac{1}{6} \sum_{i=1}^{n} \overrightarrow{OM_1^i} \cdot (\overrightarrow{OM_2^i} \times \overrightarrow{OM_3^i})$$

térfogatainak összegével egyenlő [33]. A kiszámított térfogat helyességéhez szükséges az M^i háromszögek azonos irányítása. A háromszögek normálvektorai egyidejűleg a modell belseje, vagy külseje fele kell mutassanak. Ha a normálvektorok befele mutatnak, a V_M negatív, viszont abszolút értékben megegyezik a valós térfogattal.

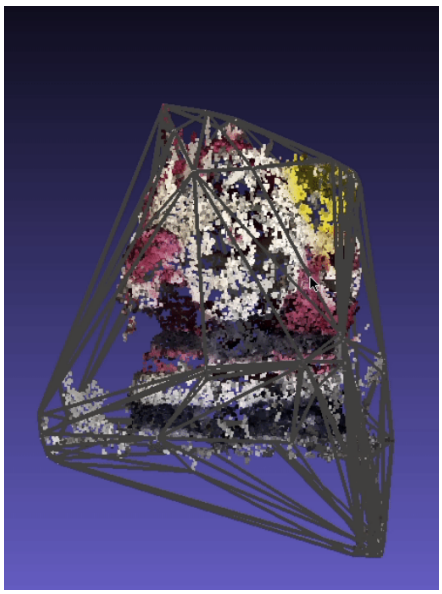
3.2. Zárt felületek létrehozása

Egy tárgy térfogata akkor értelmezhető, ha annak a felülete zárt. Az eddig létrehozott Poisson felületek nem feltétlenül zártak, következésképpen ezek önmagukban nem elegendőek a térfogatbecsléshez. Ebben az alfejezetben szó lesz a pontfelhő konvex burkáról, ami egy kezdeti megoldást jelenthet, ezt követően be lesz mutatva a felület és a burok továbbfinomított metszete, ami egy pontosabb térfogatot eredményez.

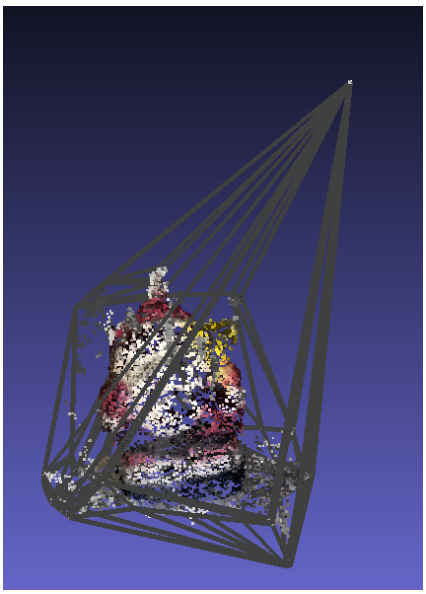
Konvex burok

Az *S* pontfelhő konvex burka megfelel annak a minimális méretű konvex halmaznak,[12] amelyik tartalmazza az *S* összes pontját [4]. Feltételezve, hogy egy pontfelhő minden pontja a becsülendő tárgy része és a tárgy teljes mértékben rekonstruálva van, a konvex burok térfogata felülről korlátolja a tárgy térfogatát. Ha a tárgy konvex, akkor a térfogata megegyezik a konvex burokéval, ha pedig konkáv, akkor a térfogata csakis kisebb lehet. Egy felhő konvex burkának a kiszámítására több módszer is létezik (*Gift Wrapping, Graham Scan, Divide et impera*).

[12] Az $\mathbb{R}^3$ minden olyan $\boldsymbol{P}$ részhalmaza, amelyre $\forall p, q \in \boldsymbol{P}$-re teljesül, hogy $\forall r \in \overline{pq}$ szakaszon levő pont szintén a $\boldsymbol{P}$-ben van, konvex halmaz.



16. ábra. A pontfelhő által meghatározott konvex burok.



17. ábra. A konvex burok zajra való érzékenysége.

Quickhull

Háromdimenziós esetben az egyik elterjedt ilyen algoritmus a quickhull [4]. Az algoritmus iteratív, hiszen a kezdeti négy pont véletlenszerű kiválasztása után folyamatosan kibővíti a burkot létrehozó halmazt a burokhoz legtávolabb eső pontokkal. A kezdeti négy pont egy tetraédert határoz meg, amelyik mindig konvex. A tetraéder mindegyik oldalát alkotó sík meghatározza a burkon kívül eső pontok egy részhalmazát, hiszen ezek a pontok a sík felett helyezkednek [13] el. Egy iteráció során minden oldal által meghatározott külső pontokból álló részhalmazból azok a pontok válnak a burok részévé, amelyek a legtávolabbra helyezkednek el a meghatározó oldaltól. A folyamat akkor ér véget, amikor a halmaz által alkotott konvex burok tartalmazza a felhő minden pontját. A folyamat véges iteráció után biztosan véget ér, hiszen az oldalak együtt meghatároznak minden burkon kívül eső pontot és ennek a véges halmaznak a számossága minden lépésben csökken.

A konvex burok hátrányai

A konvex burkok értelemszerűen konkáv tárgyak esetében rossz becslést adnak. Továbbá nagyon érzékenyek a zajra, ugyanis elég egyetlen olyan pont jelenléte a pontfelhőben amelyik a tárgyon kívül helyezkedik el ahhoz, hogy a konvex burok, és annak térfogata jelentős mértékben megnőjön (17-es ábra).

[13] Ismert a konvex burok minden oldalának az irányítása.

3.3. A konvex burok továbbfinomítása

Ebben az alfejezetben egy a tárgyat jobban megközelítő zárt felületet eredményező eljárás lesz bemutatva.

A konvex burok és a Poisson felület metszete

Abból, hogy a konvex burok térfogat szempontjából felülről korlátolja a tárgy térfogatát, következik, hogy annak a térfogata felírható a

$$V_c = V + \epsilon, \epsilon \geq 0$$

alakban, ahol V a tárgy, illetve ϵ a hibatényező térfogata.

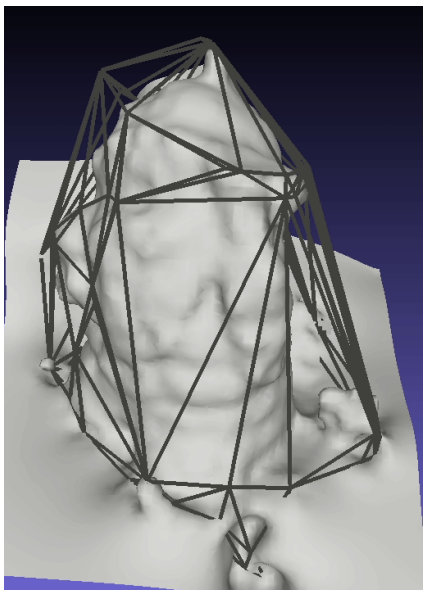
A tárgy valós térfogata felírható azon $\chi : C \rightarrow \{0, 1\}$ indikátorfüggvény kereséseként, amelyik a konvex burokban található pontokon értelmezett, egyes értékű ha az adott pont a tárgyban, vagy annak felületén helyezkedik el illetve zéró értékű ha nem. A χ indikátorfüggvény meghatározza az ϵ hibatényezőt, hiszen ϵ egyenlő annak a résztérnek a térfogatával, ahol a χ zérós értéket vesz fel. A Poisson felület a χ zéró és egy értékek közötti határvonalaként a keresett függvény egy megközelítését képezi implicit módon. A felület azért csupán megközelítés, mert nem zárt, a tárgy aljának a határvonalát nem tartalmazza. A felület irányítása ismert, következésképpen a konvex burokban található, a felület fölötti pontok nullát, a felületen, vagy az alatt elhelyezkedő pontok pedig egyes értéket vesznek fel χ-ben. A tárgy térfogatára tehát a burok felület alatti része egy jobb becslést ad.

Egy jobb konvex burok

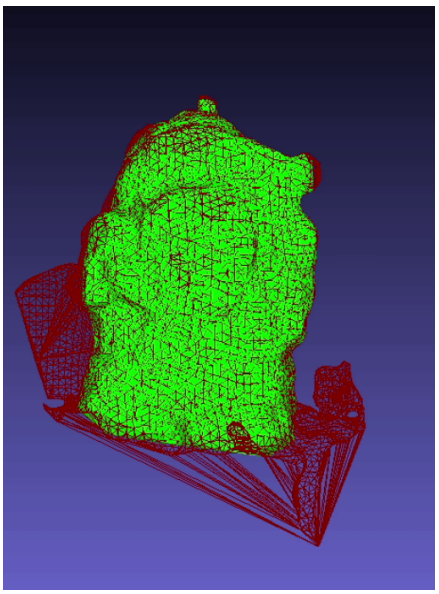
A **PMVS** által generált S_{PMVS} sűrű pontfelhőben jelentős mennyiségű zaj található, ez legfőképpen a tárgy alján megjelenő árnyékfoltoknak köszönhető. A fenti megközelítés ilyen esetben nem jelent jó becslést, hiszen a konvex burok nem illeszkedik eléggé a tárgy aljához. A **Bundle adjustment** módszerrel kapott S_{BA} pontfelhőben kevesebb zaj van jelen, ugyanis ez **AKAZE** típusú kulcspontok alapján van létrehozva, amelyek nem jelennek meg az árnyékfoltokban. Az S_{BA} azon

$$P_{BA} = \{\, p \mid \; p \in S_{BA}, \;\; p \in C\left(S_{PMVS}\right)\}$$

részhalmaza amelyik a sűrű pontfelhő $C\left(S_{PMVS}\right)$ konvex burkában található, egy jobb konvex burkot eredményez, ugyanis kevesebb zajt tartalmazhat. A zajmentességnek köszönhetően a $C\left(P_{BA}\right)$ konvex burok a tárgy alját jobban megközelíti, ezáltal csökkenti az ϵ hibatényezőt.



18. ábra. A felhő konvex burka és a hozzá illeszkedő Poisson felület



19. ábra. A piros a kezdetleges, a zöld a $C(P_{BA})$ zajmentes burokból származó megközelítése a tárgynak.

3.4. Metrikus skálára való hozás

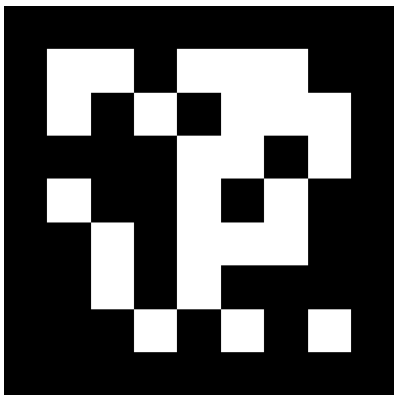
Ebben az alfejezetben a metrikus skálára hozás szükségessége, és egy referencia objektum felismerésén alapuló módszer lesz bemutatva.

Motiváció

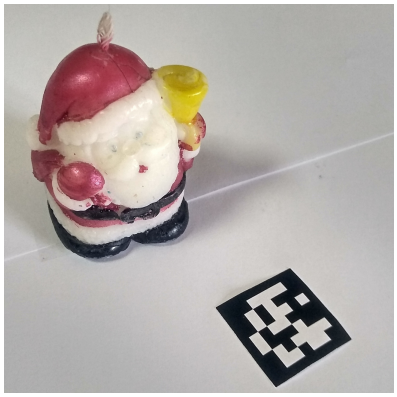
A becsült térfogat relatív természete megköveteli annak a metrikus skálára való hozását. Az S_{BA} és az ebből fakadó S_{PMVS} rekonstruált pontfelhők egy ismeretlen S hasonlósági transzformációban különböznek [14] bármely metrikus koordináta rendszertől. A térfogat a hasonlósági transzformáció rotációs és transzlációs komponenseire nézve invariáns, az s skálázási faktorra viszont nem. Az S_{BA}-beli $V_{S_{BA}}$ térfogat metrikus koordináta-rendszerben való $V_W = s^3 V_{S_{BA}}$ megfelelője s-re való tekintettel köbösen változik. Az s skálázási faktor meghatározható [15] a rekonstruált pontfelhőbeli tetszőleges két pont közötti metrikus távolság ismeretével. Elengedhetetlen az s pontos meghatározása, ugyanis a hiba ezzel egyszerre köböződik. Mindezek alapján a $V_{S_{BA}}$ relatív térfogat metrikus skálára hozható az s skálázási faktor kiszámításával.

[14] Minden W metrikus koordináta-rendszer esetén létezik legalább egy olyan $S = s[\boldsymbol{R}|\boldsymbol{t}]$ hasonlósági transzformáció, ami az S_{PMVS},S_{BA} pontfelhőbeli pontokat W-be viszi.

[15] $s = \frac{d_W(X_1,X_2)}{d_{S_{BA}}(X_1,X_2)}$ ahol $d_W(X_1,X_2)$,$d_{S_{BA}}(X_1,X_2)$ az X_1, X_2 pontok közötti valós illetve modellbeli távolság.



20. ábra. 7 × 7-es méretű Aruco marker.



21. ábra. Aruco marker mint referencia objektum.

Referencia objektum

Rekonstruált pontfelhőbeli pontok közötti valós távolság kiszámítására elengedhetetlen a kamerák helyzetéről, vagy egy referenciatárgy méretéről való plusz információ ismerete. Ha a képeket készítő készülék rendelkezik olyan szenzorokkal, amelyek a készülék térbeli elmozdulásának irányát és nagyságát mérni képesek, akkor használhatóak olyan szenzorfúziós módszerek amelyek ezen szenzorok és képek alapján megbecsülik a készülék térbeli helyzetét minden időpillanatban (*Extended Kalman filter,Visual inertial odometry*) [5]. Egy ezeknél egyszerűbb módszer az olyan referencia objektumoknak a tárggyal való egyidejű rekonstrukciója, amelyek ismert méretűek illetve a képeken könnyen felismerhetőek.

Az **Aruco** típusú markerek olyan jól meghatározott textúrával rendelkező négyzetek, amelyek képeken robusztus módon beazonosíthatóak [13]. A marker egy fekete-fehér négyzetrácsból áll, méretei 4×4-től egészen 7×7-ig terjednek. A fekete-fehér négyzetek olyan módon kell váltakozzanak, hogy a marker irányítása egyértelmű legyen. A marker sajátos textúrájának köszönhetően könnyen felismerhető a képeken, a sarkait be lehet azonosítani. A skálakülönbség meghatározásához nem elég a referenciaobjektumot csupán rekonstruálni, ismerni kell annak a modellbeli méretét. A marker $\hat{C}_i, i = 1..4$ sarkainak pontos helyzetét ismerve annak a mérete egyszerűen kiszámítható.

Maximum likelihood becslés

Az Aruco marker rekonstrukciója irányított módon történik. Az eredeti kulcspontpárokhoz hozzáadódnak a felismert sarkok megfeleltetései. A **Bundle adjustment** folyamán minden pontpálya egyetlen térbeli pontot eredményez. A referenciaobjektum nem biztos, hogy minden képen fel van ismerve, aminek következtében annak sarkai több, különböző pontpályán helyezkednek el, amelyek különböző térbeli pontokat eredményeznek. A C_{ij} pont az i-ik sarok j-ik pontpálya által adott mintavétele. A mintavételek alapján szükséges a $\hat{C}_i, i = 1..4$ sarkok egy becslését meghatározni. Feltételezve, hogy a mintavételi hibák egy 0 várható értékű normál

eloszlást követnek minden irányban, a **maximum likelihood** becslést az a $\hat{C}_i$ pont képezi, amelyik a mintavételektől a $\mathbf{RSS} = \sum_{j=1}^{m} d(\hat{C}_i, C_{ij})^2$ négyzetes távolságot minimalizálja [16].

$$\frac{d\mathbf{RSS}}{d\boldsymbol{x}} = \sum_{j=1}^{m} 2(\boldsymbol{x_{C_i}} - \boldsymbol{x_{C_{ij}}}) = 2m\boldsymbol{x_{C_i}} - 2\sum_{j=1}^{m} \boldsymbol{x_{C_{ij}}} = 0$$

$$\boldsymbol{x_{C_i}} = \frac{1}{m}\sum_{j=1}^{m} \boldsymbol{x_{C_{ij}}}$$

A $\frac{d\mathbf{RSS}}{d\boldsymbol{y}}, \frac{d\mathbf{RSS}}{d\boldsymbol{z}}$ esetek hasonló módon levezethetőek. Tehát a mintavételek $\hat{C}_i = \frac{1}{m}\sum_{j=1}^{m} C_{ij}$ centroidja a **maximum likelihood** becslés. A C_{ij} mintavétel pontosságát meghatározza az azt létrehozó pontpályának a w_j hossza. Egy pont helyzete annál pontosabb, minél több kép alapján jön létre. Ennek fényében a $\hat{C}_i = \frac{\sum_{j=1}^{m} w_j C_{ij}}{\sum_{j=1}^{m} w_j}$ visszavetítésszámmal súlyozott centroid jobb becslést jelent. A $\hat{C}_i, i = 1..4$ sarkok közötti távolságok ismeretében a marker oldalaira négy, átlóira pedig két mintavétel kiszámítható. Ezen mintavételek várható értéke lesz a marker d_{SBA} megbecsült mérete a modellben.

Mindezek után ismertnek tekinthető az $s = \frac{d_W}{d_{S_{BA}}}$ skálázási faktor aminek következtében a $V_{S_{BA}}$ relatív térfogat V_W metrikus megfelelője is meghatározható.

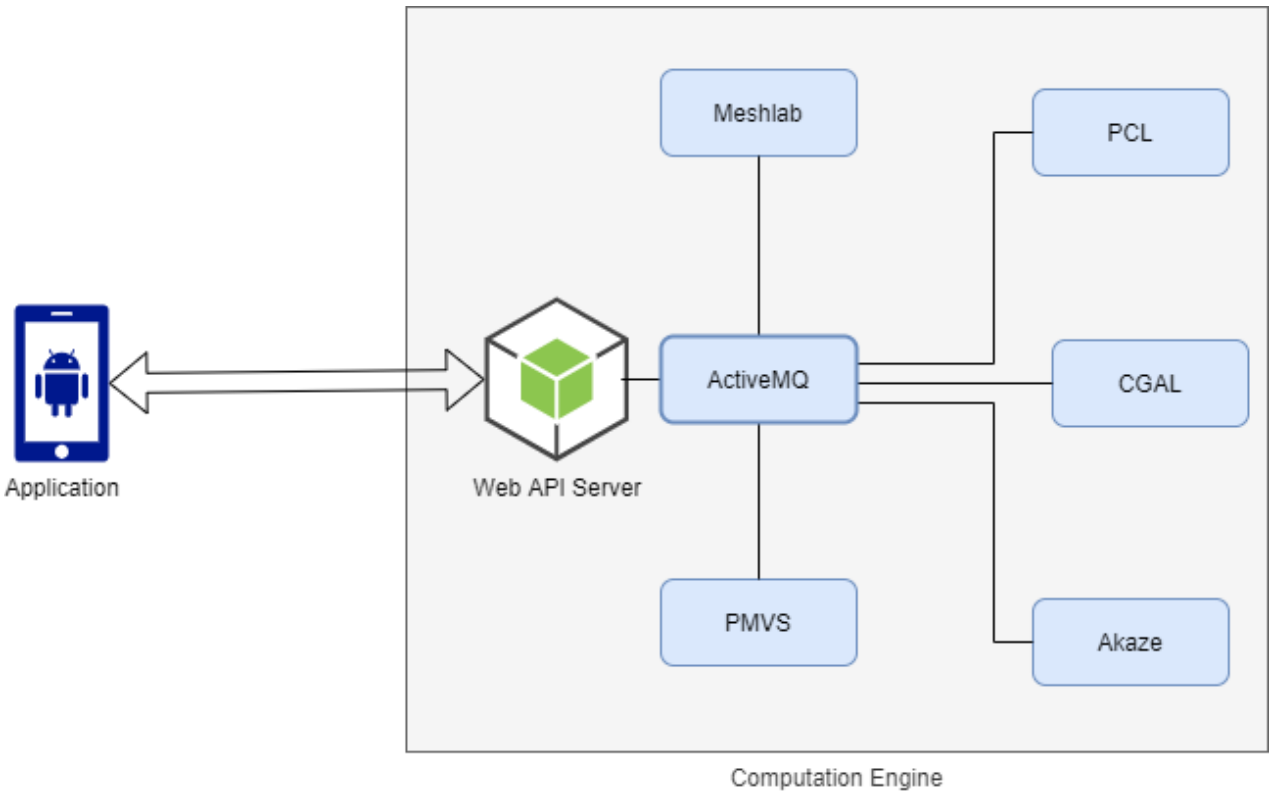
Összefoglaló

Az előző alfejezetekben egy felület térfogatának kiszámítására való eljárás és a zárt felületek szükségessége volt bemutatva. Ez után a konvex burok mintegy zárt felületként egy felső korlátot szabott a tárgy térfogatára vonatkozólag. Ezt követően két finomítási módszer segítségével ez a felső korlát csökkent. Az első módszer a burok Poisson felület fölötti részének hibatényezőként való felírása, a második pedig a **Bundle adjustment**-beli zajmentes pontfelhőből származó jobb minőségi konvex burok használata volt. Utolsó lépésként a felismert referenciaobjektum méretének a **maximum likelihood** becslésével a kapott relatív térfogat metrikus skálára van hozva.

4. Architektúra

A Volumiz3D rendszere egy Android kliensből és egy mikroszerviz alapú backend szerverből áll. Az **Android kliens** szerepe, hogy segítse a felhasználót a fényképek elkészítésében, majd ezt követően a képeket a webszerverhez továbbítsa feldolgozásra. Ezen felül az Android kliens segítségével követheti a felhasználó a feldolgozás folyamatát, majd megtekintheti a feldolgozás eredményeként kapott térfogatot és háromdimenziós modellt is.

A második fő része a rendszernek a **képfeldolgozási réteg**. Ezt kívülről a Node.js-ben íródott **web API szerver** segítségével lehet elérni. A webszerver implementációja a REST API standard szerint íródott, melynek szerepe, hogy a szükséges végpontokat biztosítsa az Android kliens számára. A backenden belül találhatóak a képfeldolgozás szakaszait elvégző modulok is, valemint egy ActiveMQ [30] üzenet bróker, amely a modulok közötti kommunikációt teszi lehetővé. A képfeldolgozás szakaszait elvégző modulok a következőek: AKAZE, GGAL, PCL, PVMS, valamint a Meshlab modul, melyek az általuk használt képfeldolgozási megoldás alapján lettek elnevezve. Az egyes modulok különálló virtualizált konténerben futnak, melyekből szükség esetén további példányokat lehet indítani, elkerülve az esetleges terhelésből adódó problémákat. Az említett modulokat a dolgozat további fejezeteiben bővebben taglaljuk.



22. ábra. A Volumiz3D rendszer felépítése.

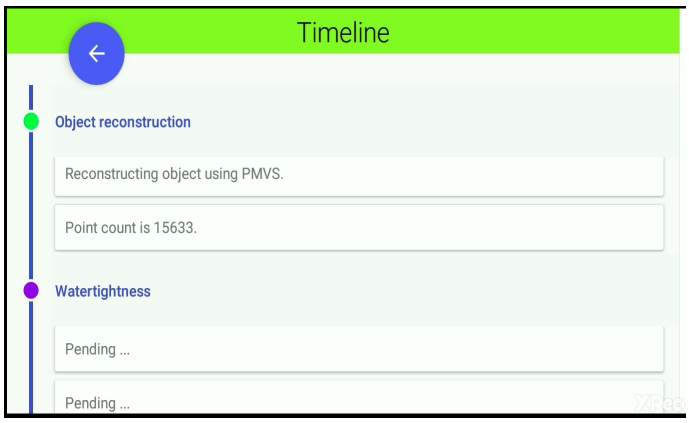
4.1. Android kliens

Ebben a fejezetben az Android kliens funkcionalitásai lesznek bemutatva. Az alkalmazás megnyitásakor a kamera nézet jelenik meg, amely az alkalmazás fő nézetét képezi. Egy adott tárgyról készített képeket, egy az alkalmazáson belül létrehozott projekt segítségével lehet egybefogni. A projekt létrehozására a *Projects* menüponton belül van lehetőség, melyhez szükséges megadni a projekt nevét, a marker információit és a készíteni kívánt képek felbontását. A projekt létrehozása után a kamera nézethez navigálva a 23-as ábrához hasonló kép fogad bennünket. A bal felső sarokban látható a projekt neve, mellette pedig egy érték, ami az eddig elkészített képek számát jelzi. A kamerán kék pontokkal vannak jelölve a kulcspontok, a jobb felső sarokban pedig látható az érzékelt kulcspontok száma. Ezek nem felelnek meg a képfeldolgozási réteg által használt kulcspontoknak, mivel azokat nem lehetne valós időben követni. Ezek a kulcspontok csak irányadóak, melyek az elmozdulás követésében segítenek a felhasználónak. A felismert markereken megjelenik a marker azonosítója. A felismerés feltétele, hogy a marker mind a négy sarka jól látható kell legyen. A felület jobb oldalán három gomb látható, az első a beállítási lehetőségeket jeleníti meg, a második gombbal fényképeket készíthetünk, valamint a harmadikkal elküldhetjük az elkészített képeket feldolgozásra.

Az első fénykép elkészítése után, az alkalmazás vektorok berajzolásával segít a felhasználónak abban, hogy a képek közötti elmozdulás mértéke megfelelő legyen. A berajzolt vektorok összekötik a legutolsó és a jelenlegi képen látható kulcspontok helyzetét, melyek hossza színekkel is reprezentálva van. A rövid vektorok hidegebb, míg a hosszabbak melegebb színűek. Ha az elmozdulás túl nagy a vektorok piros színűvé válnak. A cél az, hogy a felhasználó az elmozdulást egy optimális intervallumon belül tartsa, azaz ezek a vektorok zöld színűek maradjanak.

Elkészítés után a képek, a küldés gombra kattintva, a képfeldolgozási réteghez lesznek továbbítva. A *Status* nézeten belül egy idővonal segítségével követni lehet a feldolgozás folyamatát, ahogy a 24-es képen látható. Itt megjelenik a képfeldolgozás minden lépése. Bal

23. ábra. Az első kép elkészítése után az alkalmazás jelzi a felhasználónak az elmozdulás mértékét.

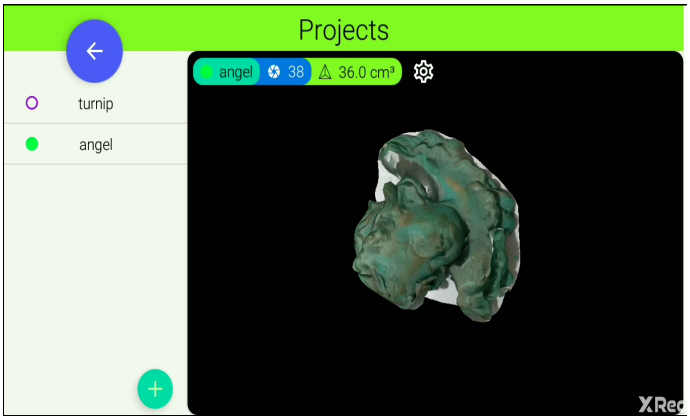


24. ábra. A képfeldolgozásnak minden lépése követhető az alkalmazásban, annak jelenlegi állapotával és részeredményekkel.

oldalon egy idővonalon színes körök jelzik a lépések állapotát: az üres várakozást jelent, a lila azt, hogy folyamatban van, a zöld azt, hogy sikeresen végre lett hajtva a művelet és a piros pedig azt jelzi, hogy hiba történt. A feladatok alatt plusz információ található a műveletek végrehajtásával kapcsolatban.

A képfeldolgozás végeztével megjelenik a kiszámított térfogat, amely megtekinthető emellett a projekt nézetben is, ahogy ez a 25-ös képen is látható. Bal oldalon látható az eddig létrehozott projektek listája, jobb oldalon pedig a jelenlegi projekthez tartozó eredmények. A fő képernyőhöz hasonlóan megjelenik a projekt elnevezése, mellette a képek számával és a tárgy térfogatával. Fekete háttérrel megjelenik a létrehozott, textúrás háromdimenziós modell, amely automatikusan forog, így a felhasználó bármely szögből megtekintheti. Lehetőség van továbbá a kurzor segítségével manuálisan is forgatni a tárgyat.

A megvalósításhoz az OpenCV [23] natív C++ implementáció lett felhasználva *Java Native Interface*-n keresztül, amely biztosítja a kulcspontok követését valós időben. A létrehozott modell megjelenítéséhez a jPCT 3D motor [19] lett használva.



25. ábra. A projektek listája látható bal oldalon, jobb oldalon pedig a jelenleg kiválasztott projekt eredménye.

4.2. Képfeldolgozási réteg

Az alábbi fejezetben a backendhez tartozó modulok kerülnek részletes bemutatásra.

4.2.1. Webszerver

A kliens és a képfeldolgozási réteg közötti kommunikáció biztosítására szolgál. A kommunikáció úgy valósul meg, hogy az Android kliens REST alapú HTTP kéréseket küld a webszerver végpontjaihoz. Három fő végpontot támogat az implementáció: */images*, */jobs* és */models*.

Az **images** végpontra tölti fel a képeket a kliens, ezzel kezdeményezve feldolgozásukat. A képeken kívül kiegészítő információkat is elküld az Android kliens, melyből az első egy, a felhasználó számára generált, azonosító. Ezen kívül továbbítva van a használt referencia objektum mérete és típusa, melyeket a projekt létrehozásakor ad meg a felhasználó, valamint a telefon által használt kamera fókusztávolsága is.

Ha a kérés megfelel a fentieknek, akkor a webszerver létrehoz egy munkafolyamatot, és annak az azonosítóját visszaküldi a válaszban. A létrehozott munkafolyamat adatai továbbítva lesznek a backend többi moduljai felé. A létrehozás után bármilyen kérés érkezik ezzel a munkafolyamattal kapcsolatban, az ugyancsak propagálva lesz a többi modul felé.

A **jobs** végpont információt szolgáltat a képfeldolgozás állapotáról, a fentebb említett munkafolyamat azonosító alapján. Itt az is ellenőrzésre kerül hogy a kérést küldő eszköz ugyanaz-e mint, amely a munkafolyamatot létrehozta. A válaszból kiderül, hogy a kép feldolgozásának melyik fázisában vagyunk, illetve az is ha hiba lépett fel.

A **models** végpontnak akkor van jelentősége ha a képfeldolgozás a végéhez ért. Ha ez megtörtént, akkor erre a végpontra küldött kéréssel a kliens a munkafolyamat és a felhasználó azonosító alapján megkapja a létrehozott modellt, az ahhoz generált textúrát, és annak a becsült térfogatát.

A webszerver a backend-en belüli kommunikációt STOMP (Streaming Text Oriented Messaging Protocol) [31] használatával valósítja meg. A munkafolyamathoz szükséges információkat továbbítja az üzenetbrókernek, amelyen keresztül eljut a megfelelő modulhoz. A képek eljuttatása a modulokhoz a docker által nyújtott shared volume-ok használatával történik.

4.2.2. ActiveMQ

Ahogy arról már szó esett, a kommunikáció megvalósításához egy üzenetbrókert került felhasználásra. A választott megoldás az **Apache ActiveMQ** [30], nyílt forráskódú, platform független üzenetbrókere. Az Apache implementációja megfelel a Java Messaging Service standardoknak, így követi a "publish-subscribe" modellt. A kommunikáció fő eleme a téma

(topic), amely segítségével csoportosítani lehet a publikált üzeneteket. Ezekre a témákra a fogadó fél feliratkozik, azaz fogyasztó (consumer) lesz.

A "publish-subscribe" modellnek két verziója van: az elsőben minden üzenetet megkap minden feliratkozó, még akkor is ha az nem volt feliratkozva a publikálás időpontjában, míg a másikban minden publikált üzenetet csak egy feliratkozott fogyasztó kap meg. A kommunikáció a backend moduljai között, az utóbbi megoldás használatával lett megvalósítva.

4.2.3. Szervizek

A Volumiz3D rendszer backendje több egymástól független modul összessége, amely microszervíz architektúra mintájára épült. Minden egyes modul önállóan működik, és a képfeldolgozás egy-egy szakaszát foglalja magába. A modulok közötti kommunikáció az ActiveMQ üzenetbrókeren keresztül történik. Mindegyik modulhoz tartozik egy topic, amelyen publikálva lesznek a hozzá tartozó feladatok. A feladat elvégzése után, a modul továbbküldi az eredményt egy üzenet formájában. Ez a fajta kommunikáció hozzájárul, hogy a rendszer könnyen skálázható legyen. Sok kérés esetén a jobban leterhelt szervizekből újabbakat elindítva könnyen megsokszorozható a rendszer teherbíróképessége. Az új szerviz példányok feliratkoznak a hozzájuk tartozó topicra és elkezdik a feladatok feldolgozását. A következőkben a képfeldolgozási modulok kerülnek bemutatásra, a végrehajtási sorrendet követve.

A képfeldolgozás első modulja az (1.5)-ös és a (2.1)-es részben leírt módszerekre épül. Első lépésként szegmentálás hajtódik végre az összes bemeneti képre, ahogy az az (1.6)-os alfejezetben bemutatásra került. A szegmentált képeken az **AKAZE** algoritmus meghatározza a kulcspontokat, majd a **bundle adjustment** algoritmus felhasználja ezeket. A bundle adjustment eljárás eredményeként létrejött pontfelhő skálája a (3.4)-es fejezet alapján van kiszámítva. Mindezek végeztével előkészíti a PMVS-hez szükséges fájlokat, majd ezeket egy üzenet formájában továbbítja a megfelelő szerviznek.

A második modul egy **PMVS** implementációt alkalmaz (lásd a (2.2)-es fejezetben), amely egy sűrű pontfelhőt hoz létre. A létrehozott sűrű pontfelhőt, és a bemenetként kapott ritka pontfelhőt egyaránt továbbítja a következő modul felé.

A harmadik szerviz a **PCL** (Point Cloud Library) [26] könyvtárból használt megoldásokat tartalmazza. Első lépésként a bemeneti pontfelhőn egy egyenletes eloszlású mintavételezést hajt végre. A mintavételezett pontfelhőnek meghatározza a konvex burkát, illetve egy Poisson felületet illeszt rá (lásd a 2.3. fejezetekben). A létrehozott felületnél és konvex buroknál biztosítja az őket alkotó síkok normáinak megfelelő irányítását.

A **CGAL** modul a képfeldolgozási rétegen belül a létrehozott felületek metszését végzi el. A metszetnek kiszámolja a térfogatát majd az első modulban kiszámolt értékkel skálázza.

A **Meshlab** modul a létrehozott metszetre viszi fel a textúrát a képek alapján. A textúrás

metszet és a térfogat továbbításra kerül az API szerver felé, és ezzel a munkafolyamat befejezettnek tekinthető.

4.2.4. Hibakezelés

Hiba esetén a modul egy üzenetet küld az API szerver felé, amely tartalmazza a részfeladat azonosítóját, és a hiba forrását. Mind a kiváltott hibák, mind a modulok által végrehajtott műveletek naplózva vannak, amelyek naplózási szintek szerint követhetőek. Hiba esetén a munkafolyamat leáll, és erről a felhasználó értesítést kap. Ha valamely hiba folytán a szolgáltatást biztosító konténer összeomlik, akkor az automatikusan újraindul, amely biztosítja a rendszer robusztusságát.

5. Eredmények és következtetések

Következtetések és továbbfejlesztési lehetőségek

Témavezetői visszaigazolás/véleményezés

Igazoljuk, hogy Bándi Nándor és Tunyogi Rudolf - Bálint hallgatók *Volumiz3D: Térfogatbecslés képek alapján* című dolgozatukat az alulírottak szakmai irányításával készítették el, és javasoljuk az említett tudományos munka bemutatását a 2020-as XXIII. reál- és humántudományi Erdélyi Tudományos Diákköri Konferencián (ETDK).

dr. Sulyok Csaba, tanársegéd
Babeş–Bolyai Tudományegyetem, Matematika és Informatika Kar

Szabó Zoltán, szoftverfejlesztő
Codespring

Farkas Eszter, szoftverfejlesztő
Codespring

Kolozsvár, 2021. február 8.

Hivatkozások

[1] *A comparative analysis of SIFT, SURF, KAZE, AKAZE, ORB, and BRISK*. 2018. URL: https://www.researchgate.net/publication/323561586_A_comparative_analysis_of_SIFT_SURF_KAZE_AKAZE_ORB_and_BRISK (utolsó elérés dátuma: 2020. márc. 31.)

[2] Sameer Agarwal, Keir Mierle, et al. *Ceres Solver*. http://ceres-solver.org.

[3] K. S. Arun, T. S. Huang, és S. D. Blostein. „Least-Squares Fitting of Two 3-D Point Sets". *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-9. évfolyam, 5. szám (1987), 698–700. oldal.

[4] C. Bradford Barber, David P. Dobkin, és Hannu Huhdanpaa. „The quickhull algorithm for convex hulls". *ACM Transactions on Mathematical Software (TOMS)* 22. évfolyam, 4. szám (1996), 469–483. oldal. DOI: 10.1145/235815.235821.

[5] Javier Civera, Andrew J. Davison, és María Martínez Montiel José. *Structure from motion using the extended kalman filter*. Springer, 2012.

[6] M. Daum és G. Dudek. „On 3-D surface reconstruction using shape from shadows". *Proceedings. 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No.98CB36231)*. 1998, 461–468. oldal.

[7] Csetverikov Dmitrij. *Digitális képelemzés alapveto algoritmusai*. URL: https://people.inf.elte.hu/redragon/Digit%e1lis%20k%e9pelemz%e9s/lec07_thr_prn_4.pdf (utolsó elérés dátuma: 2020. ápr. 1.)

[8] Csetverikov Dmitrij. *Digitális képelemzés alapvető algoritmusai*, oldal 100–109. URL: https://www.inf.elte.hu/dstore/document/297/Csetverikov_jegyzet.pdf.

[9] *Epipolar Geometry*. 1999. URL: http://www1.cs.columbia.edu/~jebara/htmlpapers/SFM/node8.html (utolsó elérés dátuma: 2020. márc. 12.)

[10] Tsung-Pao Fang és L.a. Piegl. „Delaunay triangulation in three dimensions". *IEEE Computer Graphics and Applications* 15. évfolyam, 5. szám (1995), 62–69. oldal. DOI: 10.1109/38.403829.

[11] Pablo Fernández Alcantarilla. „Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces". 2013. szept. DOI: 10.5244/C.27.13.

[12] Yasutaka Furukawa és Jean Ponce. „Accurate, Dense, and Robust Multi-View Stereopsis". *IEEE Trans. on Pattern Analysis and Machine Intelligence* 32. évfolyam, 8. szám (2010), 1362–1376. oldal.

[13] S. Garrido-Jurado et al. „Automatic generation and detection of highly reliable fiducial markers under occlusion". *Pattern Recognition* 47. évfolyam, 6. szám (2014), 2280–2292. oldal. DOI: 10.1016/j.patcog.2014.01.005.

[14] Jason Geng. *Structured-light 3D surface imaging*. 2011. URL: https://www.osapublishing.org/DirectPDFAccess/123030EB-FD65-6BAD-DE7B80F5DFFC1C9A_211561/aop-3-2-128.pdf?da=1&id=211561&seq=0&mobile=no (utolsó elérés dátuma: 2020. márc. 12.)

[15] Amal Gunatilake et al. „Real-Time 3D Profiling with RGB-D Mapping in Pipelines Using Stereo Camera Vision and Structured IR Laser Ring". *CoRR* abs/1907.12172. évfolyam, (2019). arXiv: 1907.12172. URL: http://arxiv.org/abs/1907.12172.

[16] Richard Hartley és Andrew Zisserman. *Multiple View Geometry in Computer Vision*. 2. edition. New York, NY, USA: Cambridge University Press, 2003. ISBN: 0521540518.

[17] Hugues Hoppe. „Poisson surface reconstruction and its applications". *Proceedings of the 2008 ACM symposium on Solid and physical modeling - SPM 08* (2008). DOI: 10.1145/1364901.1364904.

[18] Tushar Jadhav, Kulbir Singh, és ADITYA ABHYANKAR. „Volumetric 3D reconstruction of real objects using voxel mapping approach in a multiple-camera environment". *TURKISH JOURNAL OF ELECTRICAL ENGINEERING & COMPUTER SCIENCES* 26. évfolyam, (2018. márc.), 755–767. oldal. DOI: 10.3906/elk-1704-144.

[19] *jPCT 3D Engine*. URL: http://www.jpct.net/about.html (utolsó elérés dátuma: 2020. ápr. 15.)

[20] Prof.Fei Fei Li. *Epipolar Geometry*. 2011. URL: http://vision.stanford.edu/teaching/cs231a_autumn1112/lecture/lecture9_epipolar_geometry_cs231a.pdf (utolsó elérés dátuma: 2020. márc. 12.)

[21] William E. Lorensen és Harvey E. Cline. „Marching cubes: A high resolution 3D surface construction algorithm". *ACM SIGGRAPH Computer Graphics* 21. évfolyam, 4. szám (1987. jan.), 163–169. oldal. DOI: 10.1145/37402.37422.

[22] Jorge Nocedal és S. Wright. *Numerical optimization*. Springer-Verlag New York, 1999.

[23] *Official OpenCV Documentation*. URL: https://docs.opencv.org/2.4/ (utolsó elérés dátuma: 2020. ápr. 15.)

[24] Adrien Bartoli Pablo F. Alcantarilla Jesús Nuevo. „Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces". (). URL: http://www.bmva.org/bmvc/2013/Papers/paper0013/paper0013.pdf (utolsó elérés dátuma: 2020. ápr. 1.)

[25] Nagy Péter. *Digitális képanalízis*. URL: https://mbft.hu/assets/sejtanalitika/digitaliskepanalizisnp.pdf (utolsó elérés dátuma: 2020. márc. 12.)

[26] *Point Cloud Library*. URL: http://docs.pointclouds.org/.

[27] Meixiang Quan és Songhao Piao. „Robust visual-inertial SLAM: combination of EKF and optimization method". *CoRR* abs/1706.03648. évfolyam, (2017). arXiv: 1706.03648. URL: http://arxiv.org/abs/1706.03648.

[28] Richard J. Radke. *Computer Vision for Visual Effects*. Cambridge University Press.

[29] Noah Snavely, Steven M. Seitz, és Richard Szeliski. „Photo tourism”. *ACM Transactions on Graphics* 25. évfolyam, 3. szám (2006. jan.), 835. oldal. DOI: 10.1145/1141911.1141964.

[30] Bruce Snyder, Dejan Bosnanac, és Rob Davies. *ActiveMQ in action*. évfolyam 47. Manning Greenwich Conn., 2011.

[31] *Streaming Text Oriented Messaging Protocol*. URL: https://stomp.github.io/ (utolsó elérés dátuma: 2020. ápr. 4.)

[32] John A. Vince. *Rotation transforms for computer graphics*. Springer, 2011.

[33] Cha Zhang és Tsuhan Chen. „Efficient feature extraction for 2D/3D objects in mesh representation”. *Proceedings 2001 International Conference on Image Processing (Cat. No.01CH37205)* (). DOI: 10.1109/icip.2001.958278.