Pontkövető adatbázis előállítása forgóasztal segítségével

Pusztai Zoltán,[1 2] Hajder Levente[2]

[1] Elosztott Eseményekek Elemzése Laboratórium, MTA SZTAKI, 1111 Budapest, Kende utca 13-17
[2] Eötvös Loránd Tudományegyetem
email: {pusztai.zoltan,hajder.levente}@sztaki.mta.hu

Abstract

Jellegzetes pontokat követő rendszerek kvantitatív összehasonlítása jelentősen javíthat azok pontosságán, ha ehhez rendelkezésre állnak valós (ground truth, GT) adatok, melyekkel a tesztelés elvégezhető. Az egyik legnépszerűbb elérhető adatbázis a Middlebury, amely optikai áramlást számoló algoritmusok tesztelésére használható. Haszunálatakor jelentős megkötés, hogy nem tartalmaz forgó 3D-s objektumokat, ezért csak az elmozdulás követését lehet az adatbázis segítségével vizsgálni. A cikk célja egy forgóasztalos megoldás nyújtása GT adtok nagypontosságú előállítására. A munka során a legnagyobb kihívást a kamera, a projektor és a forgóasztal kalibrációja jelenti. Megmutatjuk, hogy ez a feladat egyszerű sakktábla alkalmazásával is lehetséges. Megoldásunk pontosságát a háromdimenziós rekonstrukció és a valós forgó objektumokról készített adatok minősége bizonyítják.

1. Bevezetés

A valós háromdimenziós adatokon nyugvó jellegzetes pontot követő rendszerek tesztelésére használatos megoldás létrehozása nem könnyű feladat, hiszen ezen objektumok egyszerre képesek forogni, elmozdulni, ráadásul csillanás is megjelenhet a felületükön. Nem könnyű olyan rendszert építeni, amely valós háromdimenziós adatokat képes nyújtani az ilyen rendszerek teszteléséhez. Cikkünk célja egy olyan strukturált fényt használó rekonstrukciós rendszer bemutatása, amely képes kellően precíz pontkövető adatokat előállítani a térbeli tárgyak mozgása során.

A Middlebury adatbázis[†] tekinthető jelenleg a legkorszerűbb GT jellegzetes pont generátornak. Az adatbázis több adathalmazból épül fel, melyet folyamatosan bővítenek 2002 óta. Az első adatkészleten valós objektumokon feleltetnek meg jellegzetes pontokat [25] egymással, így ez a pontok párosításának összehasonlítására használható. Később ezt a sztereó adatbázist kibővítették olyan új adathalmazokkal, melyeknél strukturált fényt [27] vagy ún. feltételes adatmezőt (condition random field) [22]-et használtak. Pixel alatti pontosságot is sikerült elérniük, ahogy azt [26]-ban olvashatjuk.

A tanulányunk célja pontok követése több kép során, a sztereó képek túlságosan nagy megkötést jelentenek számunkra. A Middlebury honlap elsősorban kétképes (sztereo) adatokat tartalmaz, bár nemrégiben egy optikai áramláson alapuló Middlebury adatbázist [4] is publikálták. Ennek az adatbázisnak a célja az optikai áramláson alapuló algoritmusok összehasonlítása. A legutolsó elérhető verzióban négy különböző fajtájú videókészlet található:

1. *Fluoreszkáló képek*: A nemmerev mozgást szín és UV kamerával vették fel. Az emberi szám által láthatatlan sűrű, fluoreszkáló valós adatot nyertek ezáltal. A lassú áramlást lefényképezték látható fény mellett, illetve a valós adatokat UV fény segítségével nyerték ki.
2. *Szintetizált adatbázis*: Élethű képeket generáltak képszintetizáló eljárással. A követett pontokat ezen rendszer segítségével számolták, hiszen a kamera és a háromdimenziós tér minden paramétere ismert.
3. *Képkocka interpoláció*: A videó képkockái közötti interpolációval sikerült előállítani a GT adatot.
4. *Sztereó képpár statikus térről*: Strukturált fény segítségével rekonstruálták a színteret [27] (Scharstein and Szeliski 2003), majd az optikai áramlást a valós sztereo adatokból számították ki.

A Middlebury adatbázis legnagyobb hátránya, hogy az objektumok csak egy irányban mozdulnak el, és nincs forgó mozgás az adatszettjeik között. Ez hatalmas megkötést je-

[†] http://vision.middlebury.edu/

lent, hiszen a jellegzetes pontok követése nagyobb kihívást jelent, ha más szemszögből látjuk őket.

Érdekes, hogy a Midddlebury soknézetes (multi-view) adatbázis [28] tartalmaz valós 3D rekonstrukciót két objektumról, azonban valós adatot nem nyertek ki belőlük. Az adatbázis másik korlátja, hogy adatszettenként csak két darab gyengén textúrázott objektumot használtak.

Habár a valós adatot lehetséges mélységi kamerával is generálni - pl. Microsoft Kinect - azonban az eszköz nem túl pontos. Egy másik érdekes síkbeli objektumokhoz használatos GT pont generáló eljárást publikáltak [10] 2011-ben, azonban mi szeretnék valódi háromdimenziós objektumokat is követni, nem csak síkbelieket.

Ezen megszorításokat figyelembe véve döntöttünk úgy, hogy egy speciális hardvert építünk a GT adatok generálására. A módszerünk forgóasztalt, kamerát és projektort használ. Bár ezen részek olcsón beszerezhetőek, az egész rendszer kiemelkedően pontos tud lenni, ahogy azt a következőkben részletezni fogjuk.

Forgóasztalos 3D szkennerek kalibrációja. Strukturált fényt használó szkenner használata relatívan olcsó és pontos alternatíva a 3D szkennerek építése során, ahogy azt a [19] munkában is olvashatjuk. Egy másik szintén nagy pontosságú lehetőség a lézer szkenner használata lenne, azonban a forgóasztal kalibrációja ebben az esetben nem lehetséges, mivel a lézerszkenner [7] csak egy 2D görbét tud rekonstruálni egyidőben. A forgóasztal kalibrációja során pedig muszáj 2D objektumokat rekonstruálni, mivel a forgás tengelye az ugyanazon tárgyról készült pontfelhők rekonstruálásával számítható ki.

Mi több, a kamera és a projektor külső és belső paramétereinek a meghatározása szintén kritikus. A kamera kalibrációja a jól ismert Zhang-féle kalibrációs eljárással [31] megoldható, viszont a projektor kalibrációja már nehezebb feladatnak bizonyult. A projektor felfogható egy inverz kameraként, amíg a kamera a három dimenziós teret vetíti egy képsíkra, addig a projektor egy két dimenziós képet vetít a térbe. Ezért a térbeli pontokat és a projektor képét meg lehet feleltetni egymásnak. Majd pixel-pixel megfeleltetést kell becsülnünk a kamera és a projektor között. Ezt segíti a strukturált fény használata..

Rengeteg projektor-kalibrációs eljárás létezik ezen a területen. A legnépszerűbb ilyen eljárások (pl. [24, 17, 30]) (i) egy már előre kalibrált kamerát használnak, (ii) majd egy mintát vetítenek a kalibrációs síkra, sarkokat detektálnak és ezek helyeit meghatározzák a térben, (iii) végezetül a 3D $\rightarrow$ 2D megfeleltetéseket a Zhang-féle kalibráció [31] adja. A módszer hátránya, hogy alacsony pontosságú, hiszen 3D sarokpontokat becsül, és a végső kalibrációt is ezek a becslések adják.

Egy másik lehetséges megoldás, hogy a projektort különböző helyzetekbe mozgatjuk [3, 13]. A mi esetünkben ez nem járható út, hiszen a projektor fix helyen helyezkedik el. Ráadásul ezen módszerek nem mondhatóak elég pontosnak.

Sakktáblát használunk mind a kamera, mind pedig a projektor kalibrációja során. Az algoritmusunk hasonlít a [19] által leírthoz. Ahogy azt a későbbiekben bemutatjuk, először a [31] eljárással kalibráljuk a kamerát, majd pedig robusztus homográfia becsléssel pont-pont megfeleltetéseket keresünk a kamera és a projektor pixeljei között. A belső projektor paramétereket szintén a [31]-ban leírt módszerrel számítjuk ki. A külső paramétereket (a relatív eltolás és forgatás a kamera és a projektor között) pedig a sztereó feladat megoldása adja. Erre a feladatra több módszer is található Hartley és Zisserman [12] könyvében. Azonban ezek nem adnak kellően pontos paramétereket, ezért itt ezeknél kifinomultabb módszert mutatunk be.

A főbb újdonság, amit bemutatunk, hogy nagyon pontos GT jellegzetes pontok generálhatóak forgó objektumok esetében is, ha a kamera-projektor rendszerhez egy forgóasztalt is hozzáveszünk. A legjobb tudomásunk szerint ez az első olyan rendszer amely ilyen pontos GT adatokat tud generálni. Maga a 3D rekonstrukció használata nem újszerű ötlet, viszont a GT adatok generálásához való felhasználása igen.

A kalibrációs eljárás több kisebb-nagyobb újdonságot tartalmaz:

- A kamera-projektor megfeleltetés robusztus (RANSAC) homográfia becslésen alapszik
- A forgóasztal kalibrációja teljesen újszerű: amíg a szokványos forgóasztal kalibrációk [14] a sakktábla kalibrációs eljáráson alapulnak [31], és a forgástengelyt a külső paraméterek által határozzák meg, mi egy olyan optimalizációs megoldást javasolunk, amely a visszavetítési hiba minimalizálásán alapszik. A pontosság ezáltal jelentősen növekedni fog. A forgóasztal kalibrációja során a kamera és a projektor külső paraméterei folyamatosan pontosabbak lesznek.

2. A SZERKEZET ÉS AZ ALGORITMUSOK

A 3D szkennerünk 3 fő alkotóelemből épül fel, ahogyan az az 1. ábrán látható. A bal oldalon egy sematikus ábra, míg a jobb oldalon a tényleges szkenner látható. Az eszköz fő elemei a kamera, a projektor és a forgóasztal. Mind a hármat be kell kalibrálnunk, hogy megfelelő pontosságot érjünk el a 3D szkennelés során. A kamera és a projektor az eszköz karjaihoz van rögzítve, de a forgóasztal mozgatható[‡], feladata a rá helyezett objektum elforgatása.

A komponensek kalibrációja adja a teljes módszer

[‡] Ezek a karok szintén mozgathatóak, de jelen esetben ezzel nem foglalkozunk

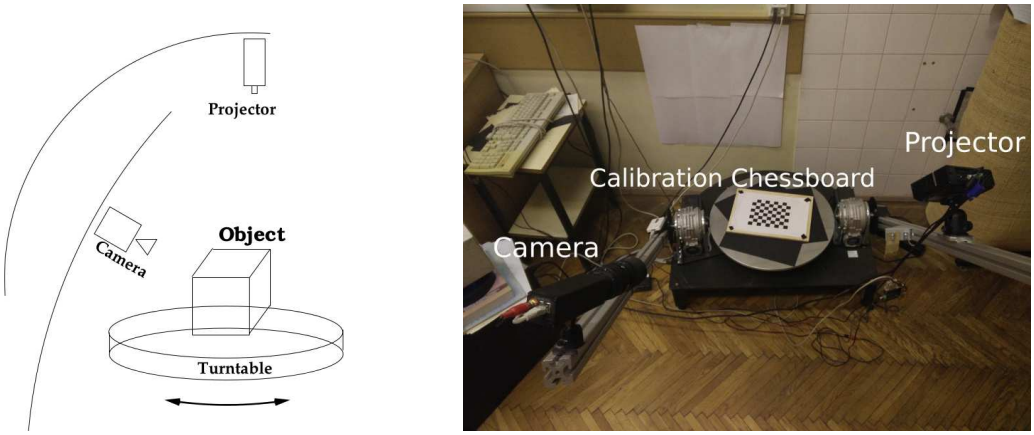

Figure 1: *A strukturált szkenner elemei: sematikus ábra (bal), kép a valódi szkennerről (jobb).*

legszűkebb keresztmetszetét. Ebben a fejezetben részletezzük, hogy hogyan lehetséges a kamerát, a projektort és a forgóasztalt kalibrálni.

A cikk a következőképpen épül fel. A szoftveres komponenseket a 2. ábra részletezi. A kamera, projektor és forgóasztal kalibrációja a 2.1, 2.1, és a 2.3 fejezetekben olvasható. A 3. fejezet elemzi az eszköz által elérhető pontosságot. Végül a 6. fejezet összefoglalja a munkánkat és annak korlátait.

2.1. A kamera kalibrációja

A kamera matematikai leírásához a lyukkamera modellt választottunk sugárirányú torzítással. Feltételezve, hogy a koordináta rendszerünk a kamerához illeszkedik, az $X \in \mathbb{R}^3$ pont vetülete a kamera síkra $u \in \mathbb{R}^2$, amit a következő egyenlet ír le:

$$u = K_c X,$$

$$K_C = \begin{bmatrix} f_x & \gamma & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix},$$

$$\tilde{u} = \begin{bmatrix} u_x[1+k_1r^2+k_2r^4] \\ u_y[1+k_1r^2+k_2r^4] \end{bmatrix},$$

$$r^2 = u_x^2 + u_y^2,$$

ahol K_C jelenti a kamera mátrixot, f_x és f_y a fókusztávolságot, (p_x, p_y) a döféspontot, γ a nyírást. A mi esetünkben csak radiális torzítást engedünk meg, ami két paraméterrel írható le: (k_1, k_2). A kamera mátrix és a torzítás paraméterei együtt adják a kamera belső paramétereit.

Egy fekete és fehér sakktáblát tartunk a kamera látószögén belül, tetszőleges pozíciókban. Képeket készítünk, meghatározzuk a sakktábla belső sarkait, majd szubpixel pontosságra finomítjuk őket. Végül a Zhang-féle [31] eljárással a kamera paraméterei meghatározhatóak.

2.2. A projektor kalibrációja

Mivel a projektort felfoghatjuk egy inverz kameraként, így az szintén leírható az előbb bemutatott kamera modellel. Azonban a megfelelő projektor pixelek megtalálása, amelyekre a sakktábla sarokpontjai vetülnek a projektor szemszögéből, nem egyszerű feladat. A probléma áthidalása érdekében strukturált fény vetítünk a térbe. Ez minden egyes projektor pixelt egyértelműen elkódol. Majd minden térbeli pont esetén vissza kell fejteni az elkódolást. Ezek után a sakktáblát olyan helyekre kell helyezni, ami jól látszik mind a kamera, mind pedig a projektor szemszögéből.

Az általunk használt strukturált fény az ún. bináris Grey kódot használja, hiszen ez bizonyult a legpontosabbnak a [27] tanulmány szerint. Ezenkívül az inverz képeket is vetítjük a dekódolás segítése céljából, ezeken a képeken felcseréljük a fekete és fehér pixeleket. Legelőször egy teljesen fehér és sötét kép is vetítésre kerül, szintén azért, hogy megkönnyítsük a kódolás visszafejtését.

Mivel projektorunk maximális felbontása 1024×768, így 42 darab képet vetítünk minden egyes sakktábla pozíció esetében: a képsorozatok egy teljesen fekete és egy teljesen fehér képből; 10 képből, melyek a horizontális; másik 10 képből, melyek a vertikális koordinátákat kódolják el, és az inverz képeket is használunk. Ezek a képek egy nézőpontból készülnek, így egy képsorozatot alkotnak.

Miután az összes kép elkészült, elkezdhetjük a strukturált fény dekódolását. Először a fény direkt és indirekt intenzitását határozzuk meg pixelenként. Az eljárás részletes leírása a [21] -ban található meg. Először a minimális L_{min}

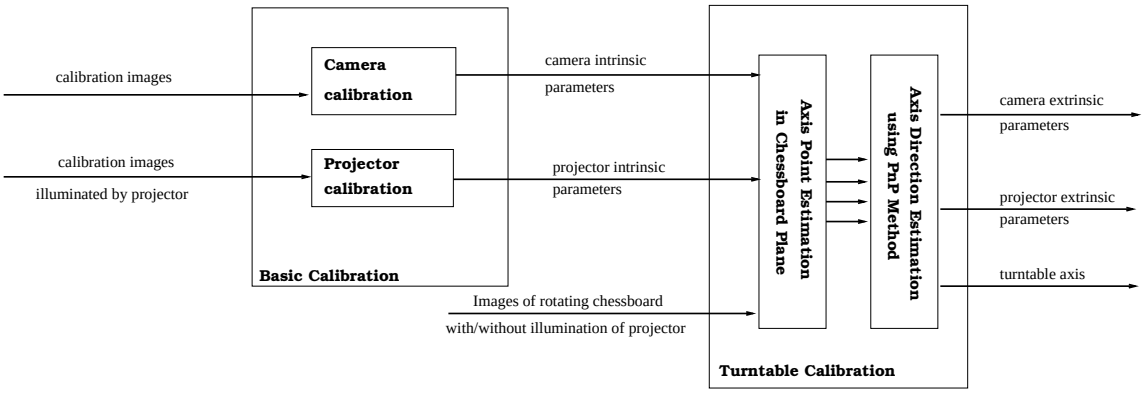

Figure 2: *A kalibrációs csővezeték szoftveres részei.*

és a maximális L_{max} intenzitást kell meghatároznunk pixelenként, majd a direkt és az indirekt értékeket az

$$L_D = \frac{L_{max} - L_{min}}{1 - B}$$

$$L_I = \frac{2(L_{min} - B * L_{max})}{1 - B^2}$$

egyenletek adják.

Ahol a B paraméter a sötét, illetve világos projektor pixelek által kibocsájtott intenzitások aránya. Ezt a két komponenst kell elválasztanunk egymástól, hiszen minket csak a projektor által kibocsájtott direkt intenzitás érdekel.

Majd klasztereznünk kell a pixeleket képpáronként, melyet a strukturált fény és annak inverze ad. 3 különböző klasztert különítünk el:

1. A pixel világít az első képen.
2. A pixel nem világít az első képen.
3. Nem lehet meghatározni.

A klaszterező szabályaink a következőek:

- $L_D < M \implies$: a pixel a 3. klaszterbe kerül,
- $L_d > L_I \wedge P_1 > P_2 \implies$: a pixel világít,
- $L_d > L_I \wedge P_1 < P_2 \implies$: a pixel nem világít,
- $P_1 < L_D \wedge P_2 > L_I \implies$: a pixel nem világít,
- $P_1 > L_I \wedge P_2 < L_I \implies$: a pixel világít,
- egyébként nem lehet meghatározni.

A pixel intenzitását az első és az inverz képen rendre a P_1 és P_2 jelöli, M beállítandó szabad paraméter, ami a mi esetünkben $M = 5$. M a túlságosan gyenge intenzitás kiszűrésében segít. A klaszterezésről többet a [29]-ban olvashatunk.

Mivel a sakktábla felváltva tartalmaz fekete és fehér négyzeteket, a sarkok dekódolás könnyen hibához vezethet. Ezek elkerülése végett homográfiákat számolunk a sarkok környezetében. 11 pixel széles ablakot használunk és mindent sikeresen dekódolt pixelt számításba veszünk. A homográfia becsléséhez a RANSAC [8] módszeren alapuló DLT homográfia becslőt alkalmazzuk, ellentétben [19]-val, ahol a robusztussággal nem foglalkoztak. Miután a homográfiát a kamera és a projektor pixelei között számoltuk ki, így arra használható, hogy a kamera pixeleket áttranszformáljunk a projektor képsíkjába. Így elérhetőek azon projektor pixelek, melyekre a sakktábla sarkok vetülnek a projektor szemszögéből, és használhatjuk a [31] módszert a projektor kalibrációjára. Fontos megjegyeznünk, hogy a projektor külső paramétereit a későbbiekben finomítani fogjuk, viszont a belső paramétereket nem változtatjuk.

2.3. Forgóasztal kalibráció

A forgóasztal kalibrációjának célja a forgóasztal tengelyének meghatározása, amely egy síkbeli pontból és egy irány vektorból áll. Ezért a szabadságfokok száma négy (kettő a síkon belüli pont meghatározása, és szintén kettő paraméter az irány).

Szerencsére a problémára több megkötés is adható. Tudjuk, hogy a tengely merőleges a forgóasztal síkjára, ezért annak iránya adott, így csak annak pozícióját kell meghatározni a forgóasztal síkjában.

A forgóasztalt kalibráltnak nevezzük, ha ismerjük azt a tengelyt, amely körül forog. Két módszert használtunk ezeknek a 3D vonalaknak a meghatározására. Először egy sakktáblát helyezünk a forgóasztalra és elforgattuk azt. Képeket készítettünk a forgatások között, majd meghatározunk a külső paramétereket, melyeket a már előre kalibrált kamera tesz lehetővé. Ez a forgó mozgás ekvivalens a fix sakktábla körüli kör pályán forgó kamera mozgásával. Kört illesztve a pontokra meghatározható a forgóasztal tengelye [14].

Azonban ez az eljárás nem bizonyult elég pontosnak. Így

egy új algoritmust dolgoztunk ki, amelyet a fejezet további részében részletezünk.

2.3.1. A forgóasztal kalibrációjának problémája

Adott egy ismert méretű sakktábla, amelynek sarokpontjai meghatározhatóak az ismert mintafelismerő algoritmusok segítségével, a cél a forgóasztal tengelyének a meghatározása. Ez a kalibráció része a komplex strukturált fényt használó 3D szkenner kalibrációjának, amely egy kamerából, egy projektorból és a forgóasztalból épül fel. Az utóbbit egy léptetőmotor mozgatja, amely képes pontosan elforgatni azt. A kamera és a projektor belső paraméterei ismertek, vagyis be vannak kalibrálva.

A bemenet a tengely kalibrációhoz a sakktábla felismert sarokpontjai. A sakktáblát elforgatjuk, és képeket készítünk különböző forgatások után. Majd a sakktáblát megemeljük a forgóasztalon úgy, hogy annak síkja ne változzon meg. Majd ismét elfogatjuk a sakktáblát, és ismét detektáljuk a sarokpontokat. (A sakktábla tetszőleges magasságba felemelhető. Mi csak egyszer emeltük meg a sakktáblát, azonban az eljárás tetszőleges számú emelés alkalmazására is kiterjeszthető.)

Ha azt az esetet vizsgáljuk, amikor a sakktábla és forgóasztal síkja párhuzamos, a köztük lévő távolság pedig h, akkor a sakktábla sarokpontjai leírhatóak a következő összefüggéssel:

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} x-o_x \\ y-o_y \end{bmatrix} + \begin{bmatrix} o_x \\ o_y \end{bmatrix} = \begin{bmatrix} \cos\alpha x - \sin\alpha y + o_x(1-\cos\alpha) + o_y \sin\alpha \\ \sin\alpha x + \cos\alpha y - o_x \cos\alpha + o_y(1-\cos\alpha) \end{bmatrix} \quad (1)$$

ahol α jelöli a jelenlegi elfordulást. Fontos megjegyezni, hogy a h tetszőleges magasság nem változtat az egyenleteken. Az X és Y betűk térbeli koordinátákat jelölnek, míg a kisbetűs megfelelőjük (x és y) a kétdimenziós koordináták a képtérben.

2.3.2. Az algoritmus

A javasolt, tengelyt kalibráló algoritmus két fő lépésből épül fel:

1. A tengely középpontjának a meghatározása sakktábla síkjában; és
2. a kamera és a projektor külső paramétereinek a finomítása.

A $[o_x, o_y]^T$ tengely meghatározása a sakktábla síkjában. A tengely meghatározásának a célja, hogy kiszámítjuk a $[o_x, o_y]^T$ sakktábla síkbeli koordinátákat. Egy alternáló algoritmust javaslunk a megoldásra, ami két lépésből áll:

Homográfia-lépés. Sík-sík homográfiát becslünk minden képre. A sarokpontok 2D helyzete a képeken ismertek. Ezen a 2D koordinátákat meg lehet határozni a sakktábla síkjában az 1. egyenlettel. Ha homogén koordinátákat is használunk, akkor felírhatjuk, hogy

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim H \begin{bmatrix} \cos\alpha x - \sin\alpha y + o_x(1-\cos\alpha) + o_y \sin\alpha \\ \sin\alpha x + \cos\alpha y - o_x \cos\alpha + o_y(1-\cos\alpha) \\ 1 \end{bmatrix}. \quad (2)$$

A DLT módszert használjuk numerikus finomításos lépésekkel [12] a homográfia becslésére. A feladatot a linearizált 2. egyenletet oldja meg:

$$E(\alpha, x, y, o_x, o_y) = E_1(\alpha, x, y, o_x, o_y) + E_2(\alpha, x, y, o_x, o_y)$$

ahol

$$\begin{aligned} E_1(\alpha, x, y, o_x, o_y) = \\ uh_{31}(\cos\alpha x - \sin\alpha y + o_x(1-\cos\alpha) + o_y\sin\alpha) + \\ uh_{32}(\sin\alpha x + \cos\alpha y + -o_x\cos\alpha + o_y(1-\cos\alpha)) + \\ uh_{33} - \\ h_{11}(\cos\alpha x - \sin\alpha y + o_x(1-\cos\alpha) + o_y\sin\alpha) - \\ h_{12}(\sin\alpha x + \cos\alpha y + -o_x\cos\alpha + o_y(1-\cos\alpha)) - \\ h_{13} \end{aligned}$$

és

$$\begin{aligned} E_2(\alpha, x, y, o_x, o_y) = \\ vh_{31}(\cos\alpha x - \sin\alpha y + o_x(1-\cos\alpha) + o_y\sin\alpha) + \\ vh_{32}(\sin\alpha x + \cos\alpha y + -o_x\cos\alpha + o_y(1-\cos\alpha)) + \\ vh_{33} - \\ h_{21}(\cos\alpha x - \sin\alpha y + o_x(1-\cos\alpha) + o_y\sin\alpha) - \\ h_{22}(\sin\alpha x + \cos\alpha y + -o_x\cos\alpha + o_y(1-\cos\alpha)) - \\ h_{13} \end{aligned}$$

Szerencsére a probléma lineáris. A koordináta-rendszer origója és skálája tetszőlegesen változtatható. Ahogyan azt [12]-ban olvashatjuk, a súlypont és a kvázi egyenközű skála választása pontos megoldást ad. A hibafüggvény $E(\alpha, x, y, o_x, o_y)$ felírható minden sakktábla sarokpont és minden forgatás esetén. Ezért a minimalizációs problémát a következőképpen formalizálhatjuk:

$$\arg\min_H \sum_{i=1}^{G_x} \sum_{j=1}^{G_y} \sum_{k=1}^{N} E(\alpha_k, x_{i,\alpha}, y_{j,\alpha}, o_{x,\alpha}, o_{y,\alpha}).$$

ahol $a_k \in [0, 2\pi]$, $x_i \in [0, G_x]$, $y_i \in [0, G_y]$, és G_x,G_y a sakktábla dimenziója.

(Lehetséges értékek a (x_i, y_j)-re a $(1,1)$,$(1,2)$,$(2,1)$,...stb.) A probléma túlhatározott homogén lineáris marad, és ezért optimálisan megoldható.

Tengely-lépés. A lépés célja az $[o_x, o_y]^T$ tengely meghatározása. A fenti két egyenlet lineáris a tengely koordinátájára nézve. Ezért az egyenletek homogén lineáris egyenletrendszert alkotnak: $A\,[o_x, o_y]^T = b$, ahol

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

A elemi kifejtve:

$$\begin{aligned} a_{11} &= h_{11} - h_{11}\cos\alpha - h_{12}\sin\alpha - \\ &\quad u(h_{31} - h_{31}\cos\alpha - h_{32}\sin\alpha), \\ a_{12} &= h_{11}\sin\alpha + h_{12} - h_{12}\cos\alpha - \\ &\quad u(h_{31}\sin\alpha + h_{32} - h_{32}\cos\alpha), \\ a_{21} &= h_{21} - h_{21}\cos\alpha - h_{22}\sin\alpha - \\ &\quad v(h_{31} - h_{31}\cos\alpha - h_{32}\sin\alpha), \\ a_{22} &= h_{21}\sin\alpha + h_{22} - h_{22}\cos\alpha - \\ &\quad v(h_{31}\sin\alpha + h_{32} - h_{32}\cos\alpha), \end{aligned}$$

Az ismert b vektor elemei:

$$b = \begin{bmatrix} b_{11} - b_{12} \\ b_{21} - b_{22} \end{bmatrix}$$

ahol

$$\begin{aligned} b_{11} &= h_{13} + \\ &h_{11}(x\cos\alpha - y\sin\alpha) + h_{12}(y\cos\alpha + x\sin\alpha), \\ b_{12} &= h_{33}) + \\ &u(h_{31}(x\cos\alpha - y\sin\alpha) + h_{32}(y\cos\alpha + x\sin\alpha), \\ b_{21} &= h_{23} + \\ &h_{21}(x\cos\alpha - y\sin\alpha) + h_{22}(y\cos\alpha + x\sin\alpha), \\ b_{22} &= h_{33} + \\ &v(h_{31}(x\cos\alpha - y\sin\alpha) + h_{32}(y\cos\alpha + x\sin\alpha)). \end{aligned}$$

A fenti egyenleteket felírhatjuk minden sakktábla sarokpont és minden forgatás esetére. Ezért mind a homográfia, mind pedig a tengely számítása túlhatározott. Így a paramétereket nagyon pontosan meg lehet becsülni. Érdekes, hogy a homográfia helyzetének becslése homogén, a tengely becslése pedig inhomogén lineáris probléma. Ezek a lineáris becsléselméletből jól ismert [6] eljárásokkal megoldhatóak.

A két lépést egymás után kell lefuttatni. Mind a két lépés minimalizálja az algebrai hibát, ezért az eljárás konvergál a legközelebbi lokális minimumhoz. Sajnos a globális optimumot nem lehet elméletben garantálni. De úgy tapasztaltuk, hogy az algoritmus a megfelelő megoldáshoz konvergál. A konvergencia relatív gyorsnak mondható, a tesztjeink során $20-30$ iteráció volt szükséges a minimum eléréséhez.

Kezdeti paraméterek. A javasolt alternációs eljárás két kezdeti értéket igényel (o_x és o_y). A pontos megoldást a tengely középpontja adja az o_x és o_y értékekre. Az algoritmus tapasztalataink szerint nem túl érzékeny a kezdeti értékekre. Mi több, kipróbáltuk jóval kifinomultabb eljárásokat is. Ha a kamera középpontját megbecsüljük a PnP (Perspective n Points) algoritmus által, mint pl. az EPNP [15] algoritmus, akkor a kamera középpontjai a forgatás során egy kört [14] formálnak, ahogy azt a fejezet elején említettünk. A kör középpontja szintén egy jó kezdeti értéket ad. Azonban úgy tapasztaltuk, hogy a megfelelő megoldást akkor is megtalálja az algoritmusunk, ha a kezdeti középpont egy tetszőleges pont a sakktáblán belül.

2.3.3. A tengely középpont becslése a globális térben.

Az első algoritmus megbecsüli a tengely pozícióját a sakktábla koordináta rendszerében. De a sakktábla különböző helyzetekbe és különböző magasságokba helyezzük. Az algoritmus célja – melyet ebben a fejezetben részletezünk – hogy a forgó sakktáblát a globális koordináta rendszerbe helyezzük, és meghatározzuk a projektor külső paramétereit (pozíció és orientáció). A globális koordináta rendszer a kamerához van igazítva, így a kamera külső paramétereit nem kell külön megbecsülnünk.

A kalibráció során összesen két sakktábla képsorozatot készítünk. A külső pozíciók könnyen meghatározhatóak. Ha a sík 3D koordinátái ismertek, a 2D pozíciókat határozzuk meg, majd a projektív paraméterinek a meghatározását PnP problémának nevezzük. Matematikailag a PnP optimalizáció a következőképpen írható le:

$$\arg\min_{R,t} \sum_{i=1}^{G_x} \sum_{j=1}^{G_y} \sum_{k=1}^{N} Rep\left(R, t, \begin{bmatrix} u_{i,\alpha} \\ v_{j,\alpha} \end{bmatrix}, \begin{bmatrix} x'_{i,\alpha} \\ h \end{bmatrix}\right)$$

ahol a Rep függvény így definiálható:

$$Rep\left(R, t, \begin{bmatrix} u_i \\ v_j \end{bmatrix}, \begin{bmatrix} x'_i \\ y'_j \\ h \end{bmatrix}\right) = \left\| DeHom\left(R \begin{bmatrix} x'_i \\ y'_j \\ h \end{bmatrix} + t\right) - \begin{bmatrix} u_i \\ v_j \end{bmatrix} \right\|_2^2 .$$

A használt aposztróf (') azt jelenti, hogy a sakktábla koordináta rendszere a $[o_x, o_y]^T$ pontokra fekszik rá. A $DeHom$ függvény a dehomogenizált 2D vektorokat állítja elő, a térbeli vektorokból szokásos módon származtatva: $DeHom([X,Y,Z]^T) = [X/Z, Y/Z]^T$.

Vannak megoldások, amelyek megbirkóznak a síkbeli

pontokkal. Mi az EPnP [15] algoritmust használtuk. Így a relatív transzformáció a sakktábla síkok és a kamera között kiszámítható. Ezeket $[R^1,t^1]$ és $[R^2,t^2]$ jelöli. A sakktábla magassága pedig megmérhető. Az általánosság megsértése nélkül az első sík magassága nullának mondható: $h_1 = 0$. (Az a legegyszerűbb, ha az első sorozat során a sakktáblát a forgóasztalra helyezzük.) Majd a második sakktábla magassága könnyen lemérhető a forgóasztalhoz képest.

Egy paraméter becslése kimerítő kereséssel is megoldható, mi is ezt a módszert választottuk. A legjobb érték a forgatáshoz, ami a PnP visszavetítési hibájára minimalizálja:

$$\arg\min_{R,t} \sum_{i=1}^{G_x} \sum_{j=1}^{G_y} \sum_{k=1}^{N} \left(Rep^1 + Rep^2 \right)$$

ahol

$$Rep^1 = Rep\left(R^1, t^1, \begin{bmatrix} u_{i,\alpha_k} \\ v_{j,\alpha_k} \end{bmatrix}, \begin{bmatrix} x'_{i,\alpha_k} \\ y'_{j,\alpha_k} \\ 0 \end{bmatrix} \right)$$

$$Rep^2 = Rep\left(R^2, t^2, \begin{bmatrix} u_{i,k} \\ v_{i,k} \end{bmatrix}, \begin{bmatrix} x'_{i,\alpha+\alpha_k} \\ y'_{j,\alpha+\alpha_k} \\ h \end{bmatrix} \right)$$

Az összefüggésben a felső index a sakktábla sorszámát jelöli. A jobb és bal oldal közötti kapcsolatot az jelenti, hogy a térbeli pontokat ugyanolyan szöggel forgatjuk el, de a fix magasságot ($\Delta\alpha$) hozzá kell adni minden forgatáshoz. (Ez az elrendezés látható a 3. ábrán.) A $\Delta\alpha$ hatása a második mátrixot tekintve:

$$R^2 = \begin{bmatrix} \cos\Delta\alpha & -\sin\Delta\alpha & 0 \\ \sin\Delta\alpha & \cos\Delta\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} R^1 \qquad (3)$$

A minimalizációs probléma szintén PnP, ezért ez is megoldható az EPnP algoritmus [15] alkalmazásával. A $\Delta\alpha$-ra vonatkozó becslés kimerítő kereséssel található meg.

Végül a projektor külső paraméterei a PnP algoritmus által határozható meg a projektor képei által detektált sarokpontokra. A kapott projektorparamétereket transzformáljuk a kamera külső paraméterek inverzével, hiszen a globális koordináta rendszerünk a kamerához van igazítva.

2.4. Objektum rekonstrukció

Az objektumok rekonstruálása hasonló a projektor kalibrációjához. Ebben az esetben a rekonstruálandó tárgyat helyezzük a forgóasztalra a sakktábla helyett. Strukturált fényt vetítünk rá, képeket készítünk, utána forgatjuk az objektumot. Az eljárást addig ismételjük, amíg az objektum vissza nem ér a kezdő helyzetébe. Majd dekódoljuk a projektor pixeleket a kibocsájtott strukturált fény segítségével minden egyes képsorozaton. Miután ez kész, Hartley-Strum triangulációt [11] használunk a megfelelő kamera-projektor pixelek meghatározására. Kiszámítjuk ezeket minden nézőpontra, majd a kapott pontfelhőket összeillesztjük.

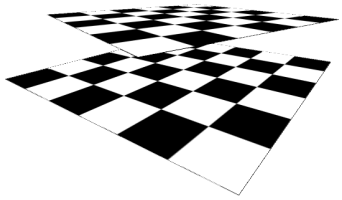

Figure 3: *A sakktáblának első helyzetének az állása. Mivel az egyenesek nem párhuzamosak, ezért a $\Delta\alpha$ relatív szöget becsülni kell.*

3. Eredmények

Az eljárásunk fő előnye az, hogy a GT adat generális teljesen automatikus. Ezért tetszőleges számú tárgyat tudunk rekonstruálni. Itt 4 különböző tárgyat mutatunk be, melyek jól követhető jellegzetes pontokat tartalmaznak. Ezek a következőek:

- **Dinoszaurusz**. Tipikus rekonstrukciós tárgy a dinoszaurusz, amely különböző cikkekben megtalálható, mint pl. [9]. Szerencsére a gyerekek szintén szeretek a dinoszauruszokat, és az egyik szerző fiának volt egy műagyag dinója, melyet rekonstruálni tudtunk. Ezért a dinoszauruszt a tesztjeink közé vettük.
- **Flakon.** A műanyag flakon kiváló tesztelésre, hiszen részletgazdag textúrát tartalmaz.
- **Plüss kutya.** Nagy kihívásnak számít a puha játék pontjainak követése, hiszen nem tartalmaz sík felületet. Ezért használtuk fel ezt is a tesztjeink során.
- **Poszter.** Az utolsó tárgyunk egy újság egyik oldala lett. Ez egy egyszerű, textúrázott sík. A jellegzetes pontkövetőket hatékonyan lehet követni ezen a példán keresztül, két ok miatt is: nincs csillogás és a jellegzetes pontok elmozdulása leírható sík-sík homográfiával.

A tesztek során a tárgyakat a forgóasztal forgatja körbe, a két pozíció közötti elfordulás 3°. A GT adatot generáló algoritmusunk kiértékelő része két modellt követ. (i) Az elsőben megkeresi azon jellegzetes pontokat a képen, amelyeket képes rekonstruálni. A pontok közül egyenletes mintavételezéssel, rácsszerűen választja ki a tesztelő eljárás a pontokat. (ii) A kiválasztott pontokat az első képen a követő (tracker) határozza meg. Mi SIFT [18] pontokat használtunk a tesztjeink során, de bármely tetszőleges pontkövető használható. A legjobb N pontot követi aztán a tesztelő eljárás.

A követendő pontokat ezután rekonstruáltuk a strukturált fény segítségével. Majd a térbeli pontokat elforgattuk a forgóasztal tengelye körül az előre beállított szöggel, és visszavetítettük őket a következő képre. Ezt az eljárást minden megmaradt képre megismételtük. A visszavetítés után kapott kétdimenziós jellegzetes pontok adják a végső valós adatot az összehasonlításhoz.

A képsorozatokat a 4– 7. ábrák szemléltetik. A rekonstruált tárgyak háromdimenziós modelljei szintén láthatóak, kivéve a Poszter teszt esetén, hiszen az síkbeli objektum és így nem érdekes a térbeli rekonstrukciója. A 3D modelleket színezett pontfelhők reprezentálják, azonban a színeknek nincs befolyása a rekonstrukcióra. Csupán a látványosság kedvéért használtuk őket.

A kiszámolt valós adatok a négy tesztünk esetében a 8–11. ábrán láthatóak. Az első sor az egyenletesen elosztott jellegzetes pontokat mutatja, míg a második sorban a 8–7. képeken a SIFT pontokat láthatjuk sárga színnel. Automatikus pontkövetőt is használtunk (BruteForceMatcher – OpenCV), és a segítségével meghatározott pontokat pirossal rajzoltuk a képekre. A rendszerek összehasonlítása nem szerves része az írásunknak, csak azt szeretnénk bemutatni, milyen könnyedén lehetséges az összehasonlítást elvégezni.

A GT adatokat vizuálisan elemeztük és nem találtunk pontatlanságot köztük. Úgy véljük, hogy a rendszerünk pontossága egy pixel alatti. Ez jelentősen kevésnek mondható, hiszen a kameránk felbontása 2592×1936 (5 Mpixel).

Figure 7: *Két kép a 'Poszter' képsorozatból.*

4. Az ismert pontkövető/megfeleltető rendszerek összehasonlítása

Bár írásunknak nem célja a pontkövető rendszerek összehasonlítása, mi lefuttatunk párat a legismertebbek közül, melyeket megtaláltunk az OpenCV programkönyvtárban [§]. Mindegyik követő több részlépésből épül fel, külön lehetséges a pontokat kinyerni, leírni és összepárosítani őket. Azonban ezek generálása és leírása ugyanazon módszerrel történik a mi példák során. A követő (párosító) algoritmus lehet különböző, mi a legpontosabb eredményt adó módszereket választottuk ki.

A következő pontmegfeleltető algoritmusokat használtuk:

1. Scale Invariant Feature Transform (SIFT) [18]
2. Speeded Up Robust Features (SURF) [5]
3. KAZE [2]
4. Accelerated KAZE (AKAZE) [1]
5. Binary Robust Invariant Scalable Keypoints (BRISK) [16]
6. ORB (Oriented FAST and Rotated BRIEF) [23]

Több megfeleltető használatával is leteszteltük a pontdetektorokat/leírokat. Az OpenCV-ben a brute-force párosítón kívül megtalálható még a FLANN (Fast Approximate Nearest Neighbor) követő [20]. Ebben az esetben a brute-force módszeren alapuló megfeleltető minden egyes jellegzetes pontot összehasonlít az első képen, minden egyes ponttal a második képről, és kiválasztja a legkisebb távolsággal rendelkezőt ezek közül. Az 12. ábrán 'BF_L1' jelenti azt, hogy a nyers erő (brute-force) megfeleltetőt használtunk L_1-es normával, míg a 'BF_H1' jelenti azt, hogy ugyanazt alkalmaztuk a Hamming távolsággal. L_2 normát használtunk azon algoritmusoknál ahol 'BF_L2' vagy 'BF_H2' felirat található.

Minden egyes teszten összehasonlítottuk a riválisokat követőket. Egy követő hibáját a követett és a valós pontok közötti távolság alapján számítjuk ki. Minden kép esetén ezt átlagoljuk, és ezen átlagokat vizsgáljuk a későbbiekben. A medián értékeket szintén számításba vesszük. Az átlag és a medián a 12. ábrán láthatóak.

A pontkövetők részletes összehasonlítása nem célja az írásunknak. Tudjuk, hogy több információra van szükség egy ilyen összehasonlítás elvégzéséhez, azonban írásunk célja csak az, hogy bemutassuk, a kvantitatív összehasonlítás lehetséges. Egy ennél részletesebb összehasonlítás a közeljövőben publikálunk.

5. Korlátok és jövőbeli tervek

A fő célunk az volt, hogy valódi forgó tárgyakról nyerjünk adatokat. A forgóasztalos megoldásunkkal nem tudunk egyenesen mozgó kamerákat szimulálni, de más adatbázisok (mint a híres Middlebury) képesek erre, ezért a mostani megoldásunkat egyesíteni lehetne a többi adatbázissal. Ettől függetlenül a szerkezethez tartozik még két mozgó kar, amelyen a kamera, illetve a projektor nyugszik, így újszerű nézőpontokat is el tudunk érni a rendszerünkkel. Ez csak akkor lehetséges, ha a karokat szintén bekalibráljuk, ez az egyik jövőbeli tervünk.

A rendszerünk másik hátránya, hogy egyes tárgyak esetén lehetséges, hogy a tárgy önmaga takarásába kerül. Ezt a hardver nem képes észlelni, ehhez felület illesztése szükséges. Ezért a jövőben folytonos felületet szeretnénk illeszteni a pontfelhőre. Ha ezek minősége megfelelő lesz, akkor ez segíthet a önmagukat takaró tárgyak problémájában.

[§] http://opencv.org

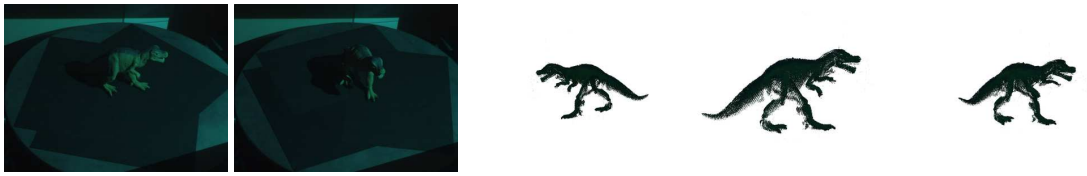

Figure 4: *Két kép a 'Dinoszaurusz' képsorozatból és a tárgy térbeli pontfelhője három nézőpontból.*

Figure 5: *Két kép a 'Plüss Kutya' képsorozatból és a tárgy térbeli pontfelhője három nézőpontból.*

Figure 6: *Két kép a 'Flakon' képsorozatból és a tárgy térbeli pontfelhője három nézőpontból.*

Figure 8: *A GT követett pontok a 'Flakon' képsorozatokon. Felső sor: a pontokat egy egyenletes lépésközű négyzetrácsról választottuk ki. Alsó sor: a SIFT által kiválasztott pontok.*

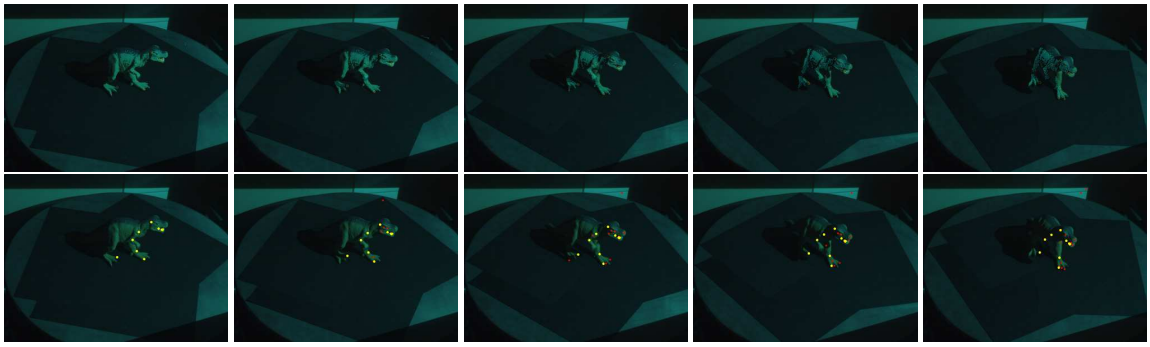

Figure 9: *A GT követett pontok a 'Dinoszaurusz' képsorozatokon. Felső sor: a pontokat egy egyenletes lépésközű négyzetrácsról választottuk ki. Alsó sor: a SIFT által kiválasztott pontok.*

Figure 10: *A GT követett pontok a 'Plüss Kutya' képsorozatokon. Felső sor: a pontokat egy egyenletes lépésközű négyzetrácsról választottuk ki. Alsó sor: a SIFT által kiválasztott pontok.*

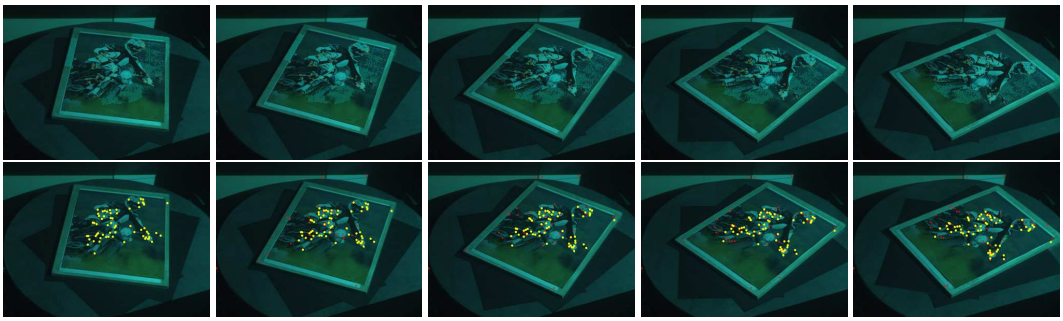

Figure 11: *A GT követett pontok a 'Poszter' képsorozatokon. Felső sor: a pontokat egy egyenletes lépésközű négyzetrácsról választottuk ki. Alsó sor: a SIFT által kiválasztott pontok.*

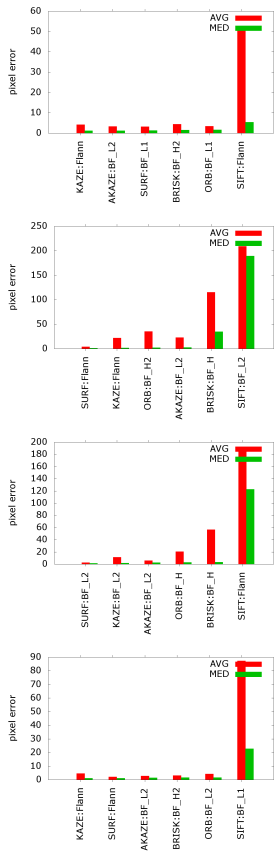

Figure 12: *A pontkövetők hibája két képsorozaton. Avg: átlag, Med: medián. A teszt tárgyak felülről lefele: 'Flakon', 'Dinoszaurusz', 'Plüss Kutya' és 'Poszter'.*

6. Összefoglalás

A bemutatott újszerű GT adatelőállító eszköz teljesen automatikusan képes forgó tárgyakról pontkövetési adatokat gyűjteni. A fő újdonságunk, hogy forgóasztalt használtunk, és megmutattuk, hogyan lehetséges ennek az eszköznek a pontos kalibrációja. Indokoltnak láttuk annak megmutatását is, hogy a 3D szkenner nem csak 3D pontfelhőt képes generálni, hanem valós, jól követett 2D-s koordinátákat is. A GT adatokat publikussá tettük, és elérhetőek a weboldalunkon ¶

References

1. P. F. Alcantarilla, J. Nuevo, and A. Bartoli. Fast explicit diffusion for accelerated features in nonlinear scale spaces. In *British Machine Vision Conf. (BMVC)*, 2013. 8

2. Pablo Fernández Alcantarilla, Adrien Bartoli, and Andrew J. Davison. Kaze features. In *ECCV (6)*, pages 214–227, 2012. 8

3. Hafeez Anwar, Irfanud Din, and Kang Park. Projector calibration for 3d scanning using virtual target images. *International Journal of Precision Engineering and Manufacturing*, 13(1):125–131, 2012. 2

4. Simon Baker, Daniel Scharstein, J.P. Lewis, Stefan Roth, MichaelJ. Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *International Journal of Computer Vision*, 92(1):1–31, 2011. 1

5. Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110(3):346–359, 2008. 8

6. Åke Björck. *Numerical Methods for Least Squares Problems*. Siam, 1996. 6

7. C. Bradley, G.W. Vickers, and J. Tlusty. Automated rapid prototyping utilizing laser scanning and free-form machining. *CIRP Annals – Manufacturing Technology*, 41(1):437–440, 1991. 2

8. M. Fischler and R. Bolles. RANdom SAmpling Consensus: a paradigm for model fitting with application to image analysis and automated cartography. *Commun. Assoc. Comp. Mach.*, 24:358–367, 1981. 4

9. A. W. "Fitzgibbon, G. Cross, and A." Zisserman. "automatic 3D model construction for turn-table sequences". In *"3D Structure from Multiple Images of Large-Scale Environments, LNCS 1506"*, pages "155–170", "1998". 7

10. Steffen Gauglitz, Tobias Höllerer, and Matthew Turk. Evaluation of interest point detectors and feature descriptors for visual tracking. *International Journal of Computer Vision*, 94(3):335–360, 2011. 2

11. R. I. Hartley and P. Sturm. Triangulation. *Computer Vision and Image Understanding: CVIU*, 68(2):146–157, 1997. 7

12. R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003. 2, 5

13. Peter Sturm Jamil Draréni, Sébastien Roy. Geometric video projector auto-calibration. In *Proceedings of the IEEE International Workshop on Projector-Camera Systems*, pages 39–46, 2009. 2

14. Csaba Kazo and Levente Hajder. High-quality structured-light scanning of 3D objects using turntable. In *IEEE 3rd International Conference on Cognitive Infocommunications (CogInfoCom)* , pages 553–557, 2012. 2, 4, 6

¶ http://web.eee.sztaki.hu

15. V. Lepetit, F.Moreno-Noguer, and P.Fua. Epnp: An accurate o(n) solution to the pnp problem. *International Journal Computer Vision*, 81(2):155–166, 2009. 6, 7

16. Stefan Leutenegger, Margarita Chli, and Roland Y. Siegwart. Brisk: Binary robust invariant scalable keypoints. In *Proceedings of the 2011 International Conference on Computer Vision*, pages 2548–2555, 2011. 8

17. Jiarui Liao and Lilong Cai. A calibration method for uncoupling projector and camera of a structured light system. In *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 770 – 774, 2008. 2

18. David G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision*, ICCV '99, pages 1150–1157, 1999. 7, 8

19. Daniel Moreno and Gabriel Taubin. Simple, accurate, and robust projector-camera calibration. In *2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission, Zurich, Switzerland, October 13-15, 2012*, pages 464–471, 2012. 2, 4

20. Marius Muja and David G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *In VISAPP International Conference on Computer Vision Theory and Applications*, pages 331–340, 2009. 8

21. Shree K. Nayar, Gurunandan Krishnan, Michael D. Grossberg, and Ramesh Raskar. Fast separation of direct and global components of a scene using high frequency illumination. *ACM Trans. Graph.*, 25(3):935–944, 2006. 3

22. Christopher J. Pal, Jerod J. Weinman, Lam C. Tran, and Daniel Scharstein. On learning conditional random fields for stereo - exploring model structures and approximate inference. *International Journal of Computer Vision*, 99(3):319–337, 2012. 1

23. Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, pages 2564–2571, 2011. 8

24. Filip Sadlo, Tim Weyrich, Ronald Peikert, and Markus H. Gross. A practical structured light acquisition system for point-based geometry and texture. In *Symposium on Point Based Graphics, Stony Brook, NY, USA, 2005. Proceedings*, pages 89–98, 2005. 2

25. D. Scharstein and R. Szeliski. A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. *International Journal of Computer Vision*, 47:7–42, 2002. 1

26. Daniel Scharstein, Heiko Hirschmüller, York Kitajima, Greg Krathwohl, Nera Nesic, Xi Wang, and Porter Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In *Pattern Recognition - 36th German Conference, GCPR 2014, Münster, Germany, September 2-5, 2014, Proceedings*, pages 31–42, 2014. 1

27. Daniel Scharstein and Richard Szeliski. High-accuracy stereo depth maps using structured light. In *CVPR (1)*, pages 195–202, 2003. 1, 3

28. Steven M. Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2006), 17-22 June 2006, New York, NY, USA*, pages 519–528, 2006. 2

29. Yi Xu and Daniel G. Aliaga. Robust pixel classification for 3d modeling with structured light. In *Proceedings of the Graphics Interface 2007 Conference, May 28-30, 2007, Montreal, Canada*, pages 233–240, 2007. 4

30. K. Yamauchi, H. Saito, and Y. Sato. Calibration of a structured light system by observing planar object from unknown viewpoints. In *19th International Conference on Pattern Recognition*, pages 1–4, 2008. 2

31. Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000. 2, 3, 4