SZÉCHENYI ISTVÁN EGYETEM
MŰSZAKI TUDOMÁNYI KAR
Informatika Tanszék

SZAKDOLGOZAT

Háromdimenziós Digitális Képfeldolgozás

Papp Szilárd

Mérnök Informatikus BSc szak

Győr, 2014

Feladatkiírás

Szakdolgozat címe: **Háromdimenziós digitális képfeldolgozás**
Hallgató neve: Papp Szilárd
Szak: **Mérnök informatikus**
Képzési szint: **BSc**
Típus: **nyilvános**

A szakdolgozat feladat leírása:

Mutassa be a képfeldolgozás alapvető fogalmait.
Ismertesse a ma használatos háromdimenziós technológiákat.
Készítsen példaprogramot, amely bemutatja a 3D pontfelhőn végezhető műveleteket.
Mutassa be a Kinect for Windows-t. Egyszerű példán mutassa be működését.
Ismertesse a 3D nyomtatás lehetőségeit.

Győr, 2014.

_____	_____
belső konzulens	tanszékvezető

Értékelő lap

Szakdolgozat címe: **Háromdimenziós digitális képfeldolgozás**
Hallgató neve: **Papp Szilárd**
Szak: **Mérnök Informatikus**
Képzési szint: **BSc**
Típus: **nyilvános**

A szakdolgozat beadható

_____	_____
dátum	aláírás (belső konzulens)

A szakdolgozat bírálatra bocsátható

_____	_____
dátum	aláírás (tanszékvezető)

A bíráló:

- Neve: Németh Zsolt
- Munkahelye: Hussarco
- Beosztása: Partner

A bíráló javaslata:

_____	_____	_____
dátum	érdemjegy	aláírás (bíráló vagy tszv.)

A belső konzulens javaslata:

_____	_____	_____
dátum	érdemjegy	aláírás (belső konzulens)

A ZVB döntése:

_____	_____	_____
dátum	érdemjegy	aláírás (ZVB elnök)

Nyilatkozat

Alulírott, Papp Szilárd, Mérnök Informatikus BSc szakos hallgató kijelentem, hogy a Háromdimenziós Digitális Képfeldolgozás című szakdolgozat feladat kidolgozása a saját munkám, abban csak a megjelölt forrásokat, és a megjelölt mértékben használtam fel, az idézés szabályainak megfelelően, a hivatkozások pontos megjelölésével.

Eredményeim saját munkán, számításokon, kutatáson, valós méréseken alapulnak, és a legjobb tudásom szerint hitelesek.

_____ _____

dátum hallgató aláírása

Összefoglalás

A háromdimenziós technológiák a 2010-es években nagy fejlődésnek indultak. Szakdolgozatomban a 3D képi információk feldolgozásával, lehetőségeivel foglalkozom példákon keresztül. A dolgozat kitér a ma használatos technológiákra adatfeldolgozási, szkennelési, nyomtatási és ember-gép kapcsolati szempontból.

Az első példában egy strukturált fényvetítésű háromdimenziós szkenner működését mutatja be a dolgozat. Ezek a szkennerek manapság már elterjedtek bizonyos szakterületeken, mint a fogászat vagy a prototípusgyártás. Képalkotás terén a statikus szkennerek közül ezt a fajtát tartják a leghatékonyabb fajtának. A megvalósítás a Brown egyetem oktatási anyagában megjelent módszer alapján, az OpenCV könyvtár segítségével készült. A feladat megoldása során a főbb szempontok a párhuzamosítás, a sebességnövelés és volt. Továbbfejlesztési lehetőségként az elsődleges egy a pontfelhőket illesztő modul volna, amely a különböző irányokból vett mintákat összeillesztve egy teljes test felületdigitalizálását eredményezné.

Második példában CT felvételek anyagvizsgálatát készítettem el. Az alumínium-kerámia kompozit felvételein a cél az anyag belsejében lévő kerámiagömb héjak detektálása volt. Az eljárás során megállapításra kerültek az egyes szeleteken látható héjak középpontjukkal és sugarukkal együtt. A felvételek kiértékelése után elegendő adat áll rendelkezésre a gömbök méretének és háromtengelyes koordinátájuk meghatározására. Az így keletkezett adatok felhasználhatóak végeselem analízisek futtatására, ami az anyagvizsgálat egyik legfontosabb kérdésköre.

Harmadik példa az emberi fül által hallott különböző frekvenciák iránykarakterisztikájának háromdimenziós modelljeit elkészítő alkalmazás. A különleges módszerrel készült felvételek eredményeiből háromdimenziós pontfelhő készül, majd ezekre a megfelelő eljárásokkal a rájuk illeszkedő felület. Az így készült modellek 3D nyomtatóval el lettek készítve. Minden egyes model egy-egy frekvencia iránykarakterisztikáit mutatja be a bal fülön reprezentálva.

Summary

The 3D technologies have gone through a significant improvement during the 2010's. This thesis is about processing 3D visual information presenting on several examples. I will write about today's technologies for data processing, scanning, printing and human interfaces.

The first example will show how a 3D scanner, based on structured light projection, functions. These devices are well-known and frequently used in some field, such as dental medicine and prototype-manufacturing. These are considered as the most efficient ones among the static types. The implementation was based on a training material of the Brown University, using the OpenCV library. The main aspects were to create a faster and more parallel solution. As further opportunity for development would be a module that adjusts a point cloud. The samples taken from different directions could be put together, resulting an object with fully digitalized surface.

The second example was made to present how CT can be used in material testing. The goal was to detect ceramic-sphere shells on pictures were taken of aluminum ceramic composite. During the process, on several slices, these shells were detected together with their center point and radius. After evaluating the records, sufficient data was available to define the size and coordinates on all three axes of the spheres. The information generated is a good starting point to run a finite element method analysis, which is an important topic in material testing.

The third example is about an application that is able to create a 3D model for the direction characteristics of the sound waves, which are hearable for the human ear. A 3D point cloud is generated using the records that were made with a special method. On this cloud a surface can be fitted. These were printed with 3D printer. Each model represents the direction characteristic of a separate frequency on the example of the left ear.

Tartalomjegyzék

1 Bevezetés 10
1.1 Témaválasztás indoklása 11
2 A digitális képfeldolgozás alapjai 12
2.1 Az OpenCV programkönyvtár 13
2.1.1 Alapvető funkciók 13
2.2 A PCL Programkönyvtár 14
3 Strukturált fényvetítéses 3D szkenner 15
3.1 A működési elv 15
3.2 Kamera kalibráció 17
3.3 Projektor kalibráció 19
3.4 A szkennelés folyamata 22
3.5 Pontfelhő rekonstrukció 24
3.6 Továbbfejlesztési lehetőségek 26
3.6.1 Megjelenítés 26
3.6.2 Multiview illesztés 27
3.7 Eredmények 28
4 Anyagszerkezet vizsgálat CT felvételeken 29
4.1 Meglévő algoritmus kiegészítéseivel tervezett módszer 29
4.1.1 Szeletek 29
4.1.2 Hough Circle – Problémák és megoldások 31
4.1.3 Analízis 32
4.1.4 Adatbázis 34
4.1.5 Analízisek összehasonlítása – legnagyobb és legkisebb detektálható körök. 34
4.1.6 A módszerből levonható következtetések 35
4.2 Rádiuszinkrementálás módszere 36
4.2.1 Belső körök 36
4.2.2 Rádiuszinkrementálás 37
4.2.3 Origó korrekció 38
4.2.4 A módszerből levonható következtetések 39
5 Iránykarakterisztikák rekonstrukciója különböző frekvenciákon 41
5.1 Pontfelhő megalkotása 42
5.2 Felület leképezése 43
5.3 Háromdimenziós megjelenítés 46
5.4 Előkészítés nyomtatásra 47
5.5 A nyomtatás 49

6 Kinect for Windows 51
6.1 A Kinect felhasználási módjai 52
6.2 Szkennelés a Kinect felhasználásával 53
7 A programok működtetése 55
8 Irodalomjegyzék 56

1 Bevezetés

A háromdimenziós technológiák térhódítása napjainkban egy re nagyobb méreteket ölt. Diplomamunkámban ennek egy kis szeletét vizsgálom, mégpedig a háromdimenziós képfeldolgozás témakörében.

Az első fejezetekben betekintést nyerhetünk a képfeldolgozás területére megismerve a felhasználási lehetőségeit. Itt nagy hangsúlyt kapnak a felhasznált programkönyvtárak, mint az OpenCV (Open Source Comuter Vision), a PCL (Point Cloud Library) és a VTK (Visualization Toolkit), de bemutatásra kerülnek a nem felhasználtak is mivel az álltaluk nyújtott lehetőségek kibővítik a háromdimenziós képfeldolgozás alkalmazási területeit. Beszélünk még a XXI. század technológiai lehetőségeiről, a felhasználási területek sokszínűségéről és a ma használatos hardverekről.

A dolgozatban e lehetőségek közül foglalkozik néhány fontosabbal. Képfeldolgozás szempontjából két nagy csoportot különböztetünk meg, a dinamikus képfeldolgozást, melyben a háromdimenziós tér időben való változását figyeljük, akár valós időben vagy analízis formájában, és a statikus képfeldolgozás, amikor a mozdulatlan teret és a benne lévő objektumokat analizáljuk. Minden bemutatott példa a dolgozatban az utóbbi, statikus képfeldolgozás területére koncentrál.

Az első ezek közül egy szkenner, mely mindösszesen egy kamerával és egy projektorral képes egy térbeli objektumok digitalizálására. Ez egy strukturált vonalvetítéses technológián alapuló eszköz, amely bizonyos területeken, mint például a fogászat, az orvostudomány, a prototípusgyártás és a műtárgy digitalizálás, már évek óta bizonyítja létjogosultságát.

A második példán keresztül a komputertomográfia által létrejövő szeletek háromdimenziós feldolgozásának egy módját láthatjuk. Az így készült képek az anyag belső szerkezetét mutatják, minden esetben 8bites színmélységben. Ám ezek nem színeket jelölnek hanem az anyag adott ponton lévő röntgen sugár áteresztő tulajdonságát.

Harmadik alkalmazás a különböző frekvenciákon hallható hang iránykarakterisztikáját analizálja. A cél itt annak szemléltetése, hogy az egyes hangmagasságokat az egyes irányokból mennyire erősen hallja az átlagos emberi fül. A látványos végeredmények műanyagból nyomtatásra kerültek.

Utolsó példában egy már-már hétköznapinak számító eszközzel a Microsoft Kinect-el foglalkozunk egy példán keresztül. Az eszköz megannyi felhasználási módja közül itt is a szkennelés bemutatására kerül sor.

1.1 Témaválasztás indoklása

Az informatika egy széles körben alkalmazott és legalább ennyi ágazattal rendelkező tudomány. A XXI. században az egyik legnagyobb fejlődésnek indult ág a háromdimenziós technológiák kérdésköre. Mind szoftveres és hardveres szempontból érdekes témakör mivel a háromdimenziós nyomtatásnak köszönhetően új lehetőségek nyíltak meg és nem utolsó sorban az emberiség figyelme is ez irányba néz. A megnövekedett igény, e technológiák folyamatos fejlődését generálja. Diplomamunka téma kiválasztása során ezen okokat, és személyes érdeklődésemet, figyelembe véve döntöttem a háromdimenziós képfeldolgozás mellett.

Az általam vizsgált alkalmazások sokrétűek. Tartalmaz a hétköznapi életben gyakran használt eszközökből, projektorból és webkamerából létrehozott szkennert, de tartalmaz olyan statikus háromdimenziós analízist melynek adatait egy CT berendezés szolgáltatja. Továbbá egy különleges feldolgozási formát melyben hanghullámok karakterisztikájának térbeli szemlétetése valósul meg. Célom ezzel is az, hogy bemutassam, milyen sok lehetőség rejlik ma a háromdimenziós technológiákban.

2 A digitális képfeldolgozás alapjai

A digitális kép diszkrét értékeket tartalmazó táblázat melyben, általánosságban pixeleket tárolunk. A pixelek fontos tulajdonsága a színmélység vagy kvantálási szint. A bináris, azaz kétszintű kvantálás során a pixelek két értéket vehetnek fel, amelyek a fehér(1) és a fekete(0) színeknek felelnek meg. Az többszintű, színmélységű képek az esetek többségében a szürkeárnyalatos képeket jelentik. A színintenzitások 8 biten ábrázolhatóak. Az RGB truecolor egy 32 biten ábrázolt additív színkeveréssel kapott színt ábrázol. A színkeverés a három csatornához rendelt többszintű képekből állítható elő. Létezik még a palettás reprezentáció, melyben a képmátrixhoz egy palettát is mellékelünk. Ilyen esetekben a képmátrix a palettaindexeket tartalmazza. A képmátrix nem csak pixeleket (picture element) hanem voxeleket (volume element) is tartalmazhat. A voxeleket a komputertomográfia által előállított szeletek reprezentálására használunk előszeretettel. A voxelek nem négyzeteket reprezentálnak, mint a pixelek, hanem kockákat, ilyen értelemben háromdimenziósnak számítanak.

Mint láthatjuk nem csak a látható fény reprezentálható képeken. A valóságba szinte minden hullámhosszon megoldható ez a megfelelő eszközzel. Hullámhossz szerinti csökkenő sorrendben:

- Rádió-hullám – radar, rádióteleszkóp, MRI
- Mikrohullám – radar
- Infravörös fény – hőkamera, infrakamera
- Látható fény – CCD kamera
- Ultraibolya képek – Vízjel a papírpénzen
- Röntgen képek (X-ray) (CT)
- Gamma képek: nukleáris medicina
- Egyéb módszer: hiper- és multispektrális képek – például infravörös és látható fény érzékelése egyidőben

2.1 Az OpenCV programkönyvtár

Az OpenCV (Open Source Computer Vision) egy nyílt forráskódú, gépi látásra, képfeldolgozásra és gépi tanulásra használatos programkönyvtár. A könyvtár tartalmazza az alapvető képfeldolgozási, a jellegzetes vonás és objektum detektáló, a videó analizáló, és a gépi tanulási funkciókat. A könyvtár ezen kívül támogatja a GPU általi feldolgozást és a legtöbb számítógépes és mobil operációs rendszert (Windows, Linux, Mac OS, iOS, Android).

2.1.1 Alapvető funkciók

- Simítás – Gauss-, medián-, és kétoldalú szűrő. Zajos képek elmosására és egyéb simításokra a legalkalmasabb a Gauss szűrő ám sebességben alulmarad a többitől.
- Küszöbölés (Threshold) – bináris, levágó, küszöbölés a nullához. A küszöbölés az egyik legegyszerűbb és legalapvetőbb szegmentálási algoritmus. Működése közben levághatjuk vagy kiemelhetjük a számunkra fontosnak ítélt részeket. A binárisnál a beállított küszöbérték feletti intenzitással rendelkező pixelek mind feketék, az alatta elhelyezkedők mind fehérek lesznek. Inverz bináris küszöböléskor a beillesztendő értékek felcserélődnek. A levágó a küszöbérték felett lévő intenzitásokat a küszöbértékkel teszi egyenlővé. A nullához való küszöbölés a küszöbérték alatti intenzitások nulla értéket vesznek fel. Inverz nullához való küszöböléskor az érték felettiek veszik fel ugyancsak a nulla értéket. Két küszöbérték is megadható thresholdoláshoz. A második (255-fekete) küszöb megváltoztatásával két oldalról végezhetjük el a szegmentálást.
- Sobel operátor – a képek első deriváltját egy 3x3–as maszk segítségével közelíti, így színátmeneteket keresve detektálja az éleket. Gyors funkció ám hátránya, hogy éles színátmeneteknél vastagodik a detektált él képe az eredetihez képest. Különlegessége, hogy minden élt csak egyszer detektál.
- Laplache operátor – a második derivált segítségével minden irányba érzékeny él keresési operátor. Elmosódott éleket is detektálni képes ám irányukat nem tudja meghatározni. Az eljárás zajérzékeny ezért érdemes Gauss simítással együtt használni.

- Canny él detektálás – A legygakrabban használt éldetektáló algoritmus. Ötvözi magában a Gauss simítást, a Sobel operátort és a kétküszöbös thresholdolást.
- Hough vonal transzformáció – a Hough tér segítségével dolgozó él keresési eljárás. Alapgondolata, hogy egy egyenes pontjainak szinuszoid görbéi ugyan abban a pontban metszik egymást.
- Hought kör transzformáció – Hough térben körök keresése is lehetséges. általánosságban véve így egy három dimenziós térben keresünk (x, y, r). A gyakorlatban adott sugarú, de legalábbis egy intervallumon belül eső köröket keresünk. Ellenkező esetben a keresés a végtelenségig folytatódhatna.
- Mintakeresés – Egy meghatározott minta megkeresése a képen a legjobban illeszkedőt visszaadva.
- Harris sarok detektálás – a képekre egy sarkossági jellemzőt kiszámítva minden pontra meghatározza, hogy az adott képen mennyire számít az adott pont saroknak. Egy megadott küszöbérték felett ez után megkeresi a lokális maximumokat. Az így kapott pontok lesznek a képen található sarkok.

2.2 A PCL Programkönyvtár

A Point Cloud Library egy a két és háromdimenziós kép és pontfelhő feldolgozásra létrehozott nyílt programkönyvtár. A könyvtárban található funkciók alkalmasak a pontfelhőben szegmentálni, különböző pontokat regisztrálni egy másik pontfelhőben, szűrőket alkalmazva eltávolítani a zajt, megtalálni a kulcspontokat, megkülönböztetni az objektumokat egymástól, rekonstruálni a felületet és mindezeket megjeleníteni.

A pontfelhőt többféleképp is képes tárolni. Az általános tárolásban a pontok koordinátájuk vagy koordinátájuk és színük szerint találhatóak. Egy másik módszerben egy ponthierarchiát hozzárendelve tárolja az adatot. Tekinthetjük ez úgy is, hogy voxelekbe rendezi a pontokat ám a nagyobb voxelek kisebbeket is tartalmazhatnak.

3 Strukturált fényvetítéses 3D szkenner

A statikus képfeldolgozásban többféle módszer ismert a háromdimenziós felület rekonstrukcióra. Az egyik ilyen a strukturált fényvetítéses módszer. Ez a szkenner legalább egy kamerát és mindössze egyetlen projektort felhasználva képes a felületet digitalizálni. Ezeken kívül szükségünk van még egy kalibrációs mintára és természetesen egy számítógépre. Az eszközzel létrehozott felületek pontosságát a minél magasabb körültekintéssel elvégzett kalibráció, a felhasznált hardverek minősége – több kamera esetén, száma – és a fényviszonyok határozzák meg.

3.1 A működési elv

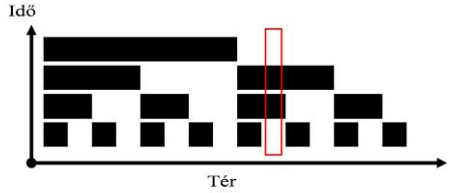
Az alapelv, hogy a projektorunkkal valamilyen sűrűségű világos és sötét vonalakat vetítünk a digitalizálni kívánt tárgyra. Ezek a vonalak a tárgyra vetítve elhajlanak a várt egyenesekhez képest a kamera képén. Ebből a torzulásból kirajzolódik a tárgy alakja (1. kép), ezáltal kiszámolható a térbeli kiterjedése is. Ahhoz, hogy ezt megtehessük a kamerát és a projektor összhangba kell hoznunk, be kell kalibrálnunk őket. A szoftveres megvalósításban az OpenCV könyvtár segítségével dolgoztam. A működési mechanikát és egyéb algoritmusokat a Brown Egyetem oktatási anyagát felhasználva készítettem el.

1. kép - A vetített vonalak elhajlása

A vetített vonalak mintáját többféleképp megadhatjuk. Az első ezek közül mikor egyetlen vonallal takarjuk ki a fehér fény elől az utat vízszintes majd függőleges irányban egyaránt úgy, hogy ez a vonal végig haladjon az adott tengelyen. Természetesen a sorrend felcserélhető és akár az egyik irány el is hagyható és a felület rekonstruálható marad úgy is,

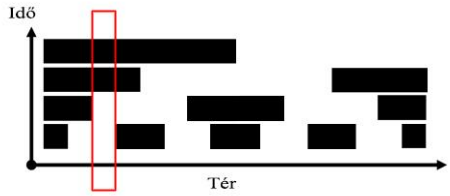
ha csak függőlegesen vetítjük a vonalat. Ezt a módszert alkalmazhatjuk úgy is, hogy egy vonallézert mozgatunk a szkennelési terület előtt.

A második módszer, hogy bináris fekete fehér vonalakat vetítünk, a tárgyra. Kétféleképp is létrehozhatunk ilyeneket. Az egyikben előbb egy kettéosztott, az egyik felén fekete a másik felén fehér képet vetítünk, majd a további képek úgy generálhatók, hogy ezt sűrítjük, tehát az előző *fekete-fehér* képet egy *fekete-fehér-fekete-fehér* kép követ. Ezt az iterációt addig folytatjuk, míg a projektor és/vagy a kamera felbontása el nem éri az egy pixelnyi vastagon vetíthető vagy látható vonalat. A másik lehetőségben szintén egy kettéosztott bináris képet vetítünk először, ám ebben a módszerben úgy sűrítjük a vonalakat, hogy az előző iterációban szereplő bináris sorozatot tükrözve illesztjük mellé így az első iterációban: {0,1}, a másodikban: {0, 1, 1, 0} értékek alapján generáljuk a vetítendő képet. Így a kód az időben tekintve és egyetlen sávra a térben minden esetben csak egyetlen bittel tér el mindkét szomszédjától (2. ábra). Az szekvencia a nevét Frank Gray-ről kapta.



Bináris kódolás
0000000011111111
1111000011110000
1100110011001100
1010101010101010

1. ábra - Bináris kódolás



Gray kódolás
0000000011111111
0000111111110000
0011110000111100
0110011001100110

2. ábra - Gray kódolás – Az oszlopok csak egy bittel térnek el szomszédjaiktól

Mindkét esetben az időben figyelve az egyes legvékonyabb csíkokat, mindig egyedi a megvilágítási kód. Legalább tíz képet kell vetítenünk vízszintes és függőleges vonalakból egyaránt, az 1024*768-as felbontáson ám a gyakorlat szerint húsz – azaz összesen negyven – kép a megfelelő mennyiség a sorok és oszlopok későbbi dekódolásához.

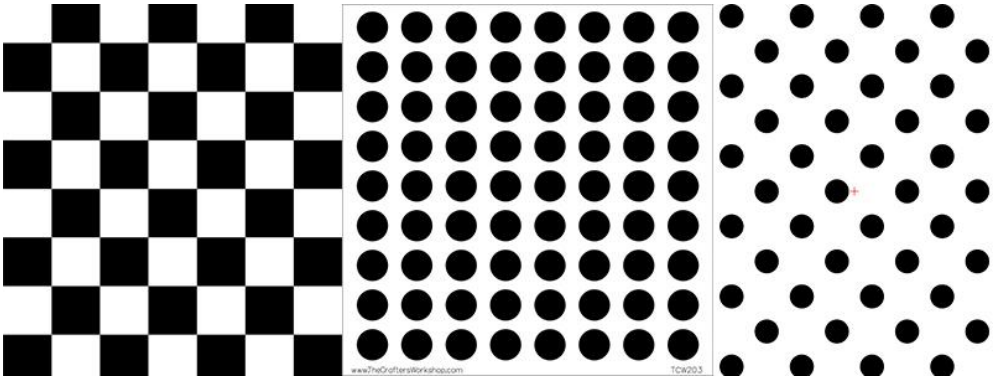
Mindkét esetben szükséges a kamera és a projektor kalibrációja, mivel a kettőjük pozíciója befolyásolja a feldolgozást. Előbb a kamerát kell bekalibrálni, majd ehhez a projektort.

3.2 Kamera kalibráció

A szkenner megvalósításához az általam használt webkamera a Microsoft LifeCam VX-3000 volt. Felbontásban 640 x 480 pixel feldolgozására képes specifikációja szerint 30 fps sebességgel. Expozíciós időben az 50Hz és a 60Hz érhető el. Ez majd a későbbiekben lesz fontos ,hogy ne interferáljon a projektorral. A webkamera viszonylag egy olcsó megoldás, és mint ilyen, hagy maga után bizonyos minőségbeli kívánni valót. A hétköznapi életben előforduló lyukkamerák többsége kisebb-nagyobb mértékben torzítják a képet. A cél a kamera kalibráció során, hogy ezt a torzulást szoftveres módon kiküszöböljük, továbbá információkat nyerjünk arról, hogy a kalibrációs tábla és a kamera milyen viszonyban van egymással, azaz milyen szögben áll a tábla és, hogy egy millimétere a valóságnak hány pixelen reprezentálódik a kamerán.

A szoftveres megvalósításhoz az OpenCV könyvtár algoritmusait használtam. Három különböző tábla segítségével kalibrálhatjuk be a kamerát ennek a könyvtárnak a segítségével.

- Klasszikus sakktábla minta – A fekete-fehér mintán a négyzetek között lévő sarkokat keressük.
- Szimmetrikus kör-rács – Sorba és oszlopokba rendezett körök melyeknek középpontjait keressük.
- Aszimmetrikus kör-rács – Ugyancsak a körök középpontjára irányul a keresés ám itt minden második oszlop a rádiusz hosszával el van tolva lefelé és mivel minden sor között pont ennyivel nagyobb a távolság az oszloptávolsághoz képest, ezért olyan mintha két rács lett volna összefésülve.

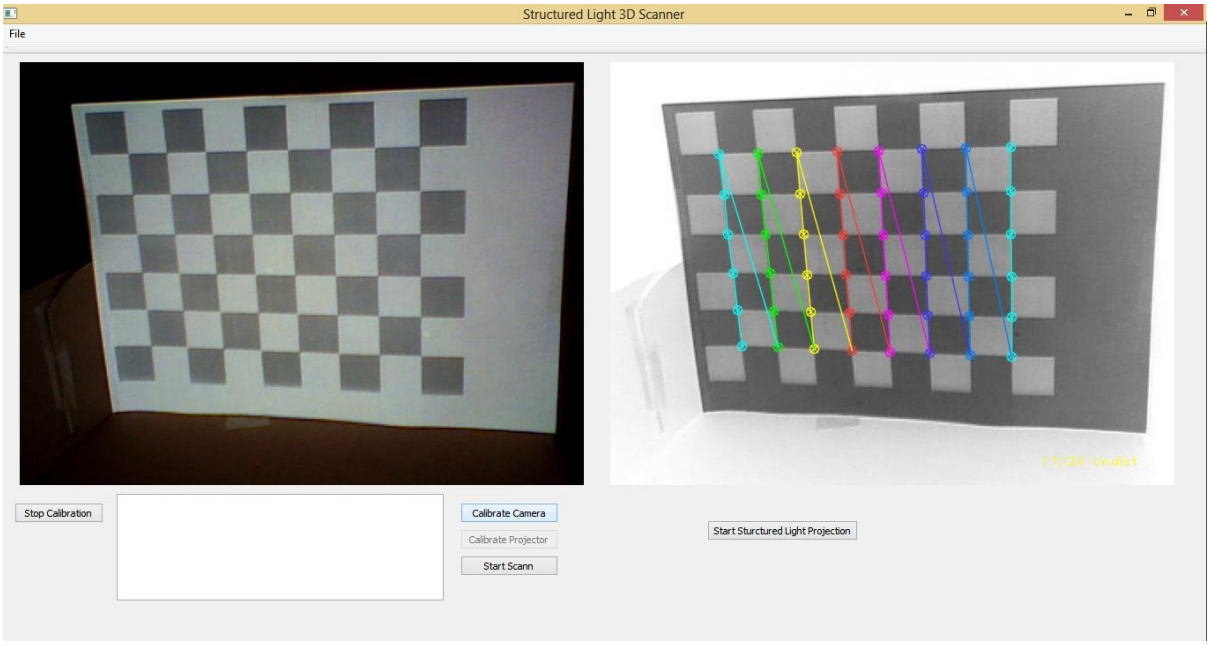


3. ábra - Kalibrációs minták

A kalibrációhoz a sakktábla mintát használjuk, méghozzá nem fekete, hanem középszürke négyzetekkel. A kalibráció gond nélkül lefut egy halványabb képen is, mivel adaptív thresholdolás is beállítható paraméterként, de ha magától mégsem találja meg, az azért van, mert a kép átlagos 8 bites intenzitása világosabb, mint a keresett négyzetek Ilyenkor egy thershold előfeldolgozással segíthetünk a kívánt eredmény elérésében.

A táblát detektáló algoritmusnak a kamera képén kívül még szüksége van a keresendő belső sarkok számára soronként és oszloponként. Visszatérésként egy logikai változót szolgáltat a keresés eredményességéről, és a egy procedúrának cím szerint átadott vektorba elmenti a megtalált sarkokat. A detektált sakokat visszarajzoljuk a kamera képére. A kalibrációhoz ezt többször meg kell ismételnünk egészen addig, amíg elegendő adatunk nem származik a sarkok helyzetéről. Ezeknek száma 15 és 25 találat között lesz elegendő.

A kalibrációs metódus ezután meghatározza az elfordítás és az elforgatás mértékének vektorait, a torzítási együtthatókat a kamera egy úgynevezett belső paraméter mátrixában, amely megadja a tábla valós pozícióját a kamerához képest. Visszatérési értékként az átlagos vetítési hibát adja meg, amelynek minél közelebb kell lennie a nullához. Ha valóban nulla ez az érték az azt jelenti, hogy a kamera CCD síkja teljesen párhuzamos a kalibrációs tábla síkjával.



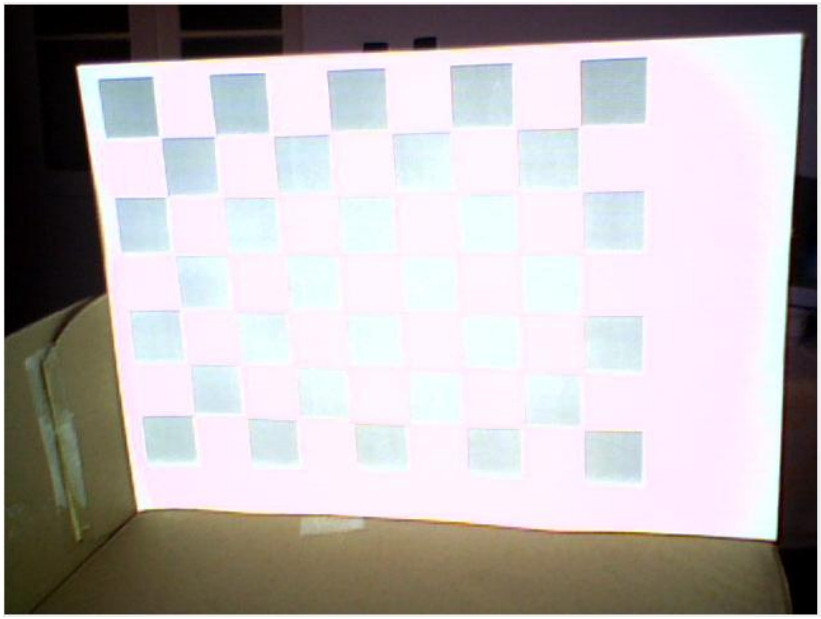
2. kép - Kamera kalibráció 6 x 8 sarkú sakktáblán

3.3 Projektor kalibráció

A szkenner megvalósításához Thoshiba TLP-S10 típusú projektort használtam. A felbontását elegendő 1280x1024-re beállítani, mivel a kameránk amúgy sem képes a túl sűrű vonalak feldolgozására. A frissítési frekvenciát ugyan úgy, mint kamera expozíciós idejét is, állítsuk 60Hz-re. Ezek mindenképp meg kel, hogy egyezzenek. Ha nem így tennénk, akkor a kivetített kép a kamerán interferenciát eredményezhet. Kevés mértékű csíkozódás is elegendő ahhoz, hogy elmossa a vetített képet, legtöbb esetben a feldolgozhatatlan szintre, vagy csak annyira, hogy hibásan detektáljuk a kivetített mintákat.

A projektor kalibráció során az előzőleg a kamera kalibráció során használt táblára vetítjük ugyan azt a kalibrációs sakktáblát, ami a megvetítendő felületen található. A projektor nem fogja és nem is kell, hogy tökéletesen rávetítse a négyzeteket a táblára. A feldolgozás során azt fontos észben tartani, hogy most az egymás felett lévő két sakktábla közül csak a vetítettet szabad detektálnunk. Ebben van segítségünkre az, hogy maga a kalibrációs tábla nem fekete négyzetekből áll, hanem középszürke árnyalatúakból. Ha erre vetítünk bizonyos előfeldolgozás segítségével kiszűrhetjük a számunkra értelmezni kívánt adatot.

A vetítés során fontos, hogy a kamera képe ne égjen ki. Ha a projektorral túl világos képet vetítünk, akkor a kamera képén azt figyelhetjük meg, hogy az egyébként sötét részek is világosak más szóval a kép kiég (3. kép).



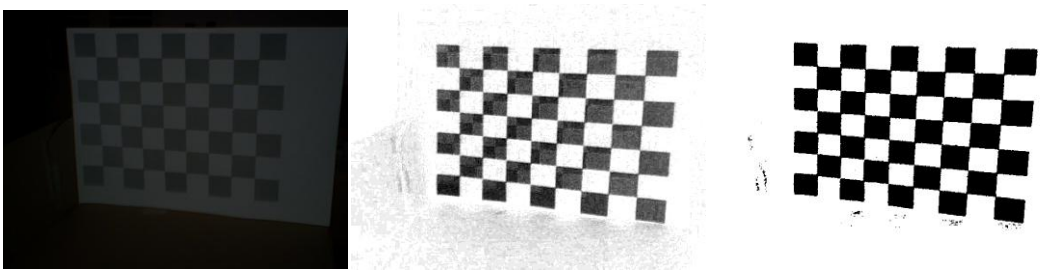
3. kép - Túl világos kép vetítése

Ezt a problémát úgy küszöbölhetjük ki, hogy a generált sakktábla színeinek intenzitását egy százalékos arányban határozzuk meg. Ez azt jelenti, hogy a beállított érték szerint sötétebb pixelekre cseréljük a világosakat. Ennek hatására a sötét és világos négyzetek közötti kontraszt egyre kisebb mértékű lesz így csak annyira szabad a megvilágítást ezzel a módszerrel korrigálni, amilyen mértékben a kamera feltétlenül megkívánja. Ezen felül a kamera képét is sötétíthetjük ugyan ezzel a módszerrel annak érdekében, hogy a kívül eső területek teljesen feketék legyenek a képen a feldolgozás során.

A kalibráció során első lépésként egy teljesen egy színű világos képet vetítünk a kalibrációs táblára. Erre azért van szükség, mert a tábla képére szükségünk lesz a későbbiekben, méghozzá olyan képnek kell lennie róla, melyen jól látszódnak a négyzetek, ezért erre a legalkalmasabb, ha a megvilágított területen a fizikailag is a táblán lévő sakktáblát detektáljuk. Abban az esetben, ha ez megtörtént, akkor ugyan azokkal a beállításokkal rávetítjük a felületre a sakktábla mintát és egy képet készítünk erről is.

Mivel ez a kép kétszer annyi sarkot tartalmaz, mint amennyit mi keresünk ezért előfeldolgozáson esik át, melynek célja, hogy eltüntessük a képről az összes olyan információt, amely nem a vetített minta adatait tartalmazza. Az előfeldolgozás lépései:

- Mindkét kép szürkeárnyalatúra konvertálása.
- Háttér kivonása a képből. Az eredményként kapott képen már nem vagy alig látszanak a fizikailag is a táblán lévő négyzetek.
- Az eredmény invertálása. Erre azért van szükség mivel a kivonás egy inverz képet eredményez.(4. kép)
- Thresholdolás. A kép még tartalmaz bizonyos mennyiségű zajt, de a legsötétebb részek mindenképp az általunk kiemelni kívánt négyzetek.



4. kép - Előfeldolgozás – Balról jobbra: háttér, vetített a hátteret kivonva, thresholdolt vetített kép

Az átalakítás után a sarkok könnyedén detektálhatók. Ebben az esetben is többször meg kell ismételnünk a lépéseket, hiszen nem mindig pontosan ugyan azokon a pontokon detektálja az algoritmus a sarkokat.

Ez után következik a projektor kalibrációs adatok átvitele a rögzített értékekből, azaz meg kell határozni a kivetítő sakktáblájához tartozó képpontokat és a torzításmentes sarokpontokat mind a kamera mind a projektor sakktábla képeire. Ez után kiszámítható a tárgy síkja és a kép síkja közötti perspektív transzformáció. Így olyan adatokat kapunk mintha a projektor szemszögéből vizsgáltuk volna a vetítés.

Miután ezekkel készen vagyunk, a kamera kalibrációs metódust kell futtatnunk, de most a projektor sakktáblájához meghatározott pontokra a projektor szemszögéből. Ebből a szempontból a projektort egy inverz kamerának tekinthetjük. Így megkapjuk a torzítás értékét, az elforgatás és elfordítás vektorokat, továbbá a vetített sík belső paraméter mátrixát, ami tartalmazza a gyújtótávolságot, a főpontot, a skálázási tényezőket.

Az összes kalibrációs tényező változatlanul felhasználható mindaddig, míg a kamerát, a projektort és a vetítési felületet nem mozdítjuk el egymástól így érdemes ezeket az adatokat elmenteni és indításkor újra betölteni. A kalibrációs táblára ettől a ponttól a gyakorlatban nincs már szükség.

3.4 A szkennelés folyamata

A sikeres kalibráció után lehetővé válik tárgyak beszkennelése. A Start Scann gombra kattintva megnyithatjuk azt az ablakot, melynek segítségével ezt elvégezhetjük. Mint eddig is, most is nagyon fontos, hogy a kamerakép fényereje és a vetített kép világossága összhangban legyenek. Első feladatkén ezért végezzük el a beállításukat nagyon körültekintően. A kalibrációs táblát, ha úgy érezzük, ki is vehetjük a megvilágított területről, ám az sem baj, ha a helyén hagyjuk.

Második lépésként a hátteret kell beszkennelnünk, amire azért van szükség, mert az ebben a fázisban létrejött pontokat a következő szkennelési fázisokban kivonhatjuk az eredményekből. Ezt már nyugodtan nevezhetjük szkennelésnek is, hiszen a háttér beolvasása semmiben sem különbözik a továbbiakban elvégzendő szkennelésektől, csak a felhasználásának a módja más.

A legcélszerűbb természetesen olyan helyen elvégezni ezt ahol a lehetőségek szerint a legnagyobb sötétség van, továbbá sokat segít, ha fekete színű az a felület, amire a beszkennelendő tárgyat helyezzük. Valamennyi háttér így is beolvasásra kerül, ám rossz fényviszonyok között előfordulhat, hogy a hátteret nem tudjuk beolvasni minden részletében és így az megjelenhet a valódi beolvasás során. Természetesen itt az is közrejátszik, hogy kis mértékben, de előfordul, hogy ugyanazt a pontot két beolvasás, egymáshoz képest más helyen detektálja. Ilyenkor két különböző pontként van jelen, így a kivonáskor nem törlődik a ponthalmazból.

A programban a beolvasást megelőzi az a szakasz, melyben létrehozzuk a Gray kódokat. Ezeket a szkenner felbontásához igazítva bináris – fekete és fehér – képként hozzuk létre. Ezek után ki kell őket vetíteni a projektorral és ezzel párhuzamosan rögzítenünk a webkamera képét, amint ez kész, máris vetíthetjük a következő képet.

A Gray kódok vetítése is két fázisból áll. Előbb kivetítjük a kódot majd annak az inverz képét is. Ezzel az eljárással megkétszerezzük a vetítendő képek számát. Az így létrejött képeknek egy puffer szolgáltat helyet. A feldolgozásuk csak az után következik, hogy mindet rögzítettük.

Az elmentett képek egy előfeldolgozáson esnek át, amiben a megfelelő formára hozzuk a pontfelhő rekonstrukciójához.

A képeken a felvételek kiértékelése során, kettesével haladva dolgozunk. Az egymás mellett lévő kép párok minden esetben ellentétes Gray kód alapján keletkeztek, inverzei egymásnak. A sötét és világos sávok határán a vetített képeken egy nagyon vékony színátmenet jelentkezik. Mivel ez az ellentétes képeken egy ellentétes irányú ám azonos pozíciójú színátmeneti gradiens, kihasználhatjuk, mikor kivonjuk őket egymásból. Ennek eredményeként egy vékony vonal rajzolódik ki (5. kép). A továbbiakban ezeket a vonalakat kell dekódolnunk. Ha ez a színátmenet nem lenne, akkor a kivonás során egy olyan képet kapnánk, melyről úgy tűnhet, hogy homogén fényű megvilágítást kapott elkészülésekor.

5. kép - Képpárok kivonása

Ez után egy skaláris értékhez hasonlítjuk a képet és leszűrünk minden olyan pontot, amelynek nagyobb vagy egyenlő a színintenzitása. Az így kapott képet egy *vagy* műveletnek vetjük alá, ahol a feltételben szereplő másik kép egy teljesen homogén fekete maszk. A továbbiakban már ezt az eredményt fogjuk a műveletek maszkolási paramétereként alkalmazni. Ezután a két eredeti képpárt fogjuk összehasonlítani egymással szintén olyan szempontból hogy melyik pont világosabb a másiknál, azaz eredményként egy olyan képet kapunk, amiben az adott koordinátán a magasabb színintenzitású lesz a pontpárok közül.

Ha az első pontpárt dolgozzuk fel, akkor készítünk egy másolatot erről a képről a következő képpárok számára, ellenben ha már létezik ilyen másolat, tehát nem az első képpárt dolgozzuk fel, akkor egy XOR műveletet hajtunk végre közöttük, majd egy skaláris értéket hozzáadunk minden pontjához. Ezekkel a műveletekkel létrejön az oszlopok és a sorok által detektált ponthalmaz. Ha összehasonlítjuk ezeket a szélességükkel és magassággukkal, akkor kapunk egy belőlük képzett maszkot, aminek az értékeit egy ideiglenes változóban gyűjtjük úgy, hogy összeadjuk a képeket. Végül az ideiglenes változót maszkként használva, NULL értékekre állítjuk a detektált sorokat és oszlopokat.

3.5 Pontfelhő rekonstrukció

A feladat ennél a fázisnál, hogy a kamera sugarában megtaláljuk a metszéspontokat a háromdimenziós síkkal. Ezt mind a vetített oszlopokra és sorokra meg kell tennünk. Ehhez normalizált pontokká kel konvertálni a kamera pontjait a gyújtópont segítségével. A kalibráció során keletkezett 3x3 –s belső kamera paraméterek közül nekünk a gyújtópontra van szükségünk, az adott kamera sugárra, amely a torzításmentes kameraképből származtatott irányvektor, valamint a projektor megfelelő oszlop és sor síkjaira. Az ezekből számított vektor-sík metszéspontok felelnek meg a háromdimenziós koordinátáknak. Ha ezek az adatok rendelkezésre állnak, akkor a metszéspont a következőképp számolható:

- *q* a kamera gyújtópontpont
- *v* a kamera sugár vektora, amely átmegy *q*-n és a keresett metszéspont felé mutat
- *w* a projektor vetített síkja amin keresünk
- *p* egy keresett pont a vetített oszlopokon

```
float n_dot_q = 0, n_dot_v = 0;
float point_cols[3];
float depth_cols;
for(int i=0; i<3; i++){
    n_dot_q += w[i]*q[i];
    n_dot_v += w[i]*v[i];
}

depth = (w[3]-n_dot_q)/n_dot_v;
for(int i=0; i<3; i++)
    p[i] = q[i] + depth*v[i];
```

Másik lehetőség a „ray-ray” triangulation, ahol a felület leképezését a kamera és a projektor sugarak metszéspontja adja meg. Ehhez szükségünk van a projektor középpontjára, a kamera gyújtópontjára és a két sugárra. Ezek segítségével kiszámolható a legközelebbi pont a két háromdimenziós vonalközött.

A„Ray-ray” háromszögelés:

- q1 a kamera középpont
- q2 a projektor középpont
- v1 a kamerasugár
- v2 a projektor sugár

```
float q12[3], v1_dot_v1 = 0, v2_dot_v2 = 0,
      v1_dot_v2 = 0, q12_dot_v1 = 0, q12_dot_v2 = 0;

    for(int i=0; i<3; i++){
        q12[i]      =  q1[i]-q2[i];
        v1_dot_v1  +=  v1[i]*v1[i];
        v2_dot_v2  +=  v2[i]*v2[i];
        v1_dot_v2  +=  v1[i]*v2[i];
        q12_dot_v1 += q12[i]*v1[i];
        q12_dot_v2 += q12[i]*v2[i];
    }

    float s, t, denom;
    denom = v1_dot_v1*v2_dot_v2 - v1_dot_v2*v1_dot_v2;
    s =  (v1_dot_v2/denom)*q12_dot_v2 - (v2_dot_v2/denom)*q12_dot_v1;
    t = -(v1_dot_v2/denom)*q12_dot_v1 + (v1_dot_v1/denom)*q12_dot_v2;


    for(int i=0; i<3; i++)
      p[i] = ( (q1[i]+s*v1[i]) + (q2[i]+t*v2[i]) )/2;
```

Ez után le kell vonni a háttér pontokat. Ezt egy feltétel alapján tesszük. Ehhez előbb kiszámoljuk a különbségét a szkennelt mélységnek és a háttér mélységének. Ha eza különbség kisebb a minimális tűrésnél és vannak is pontok azon a területen, akkor azokon a pontokon nulla értékkel töltjük fel a változókat.

3.6 Továbbfejlesztési lehetőségek

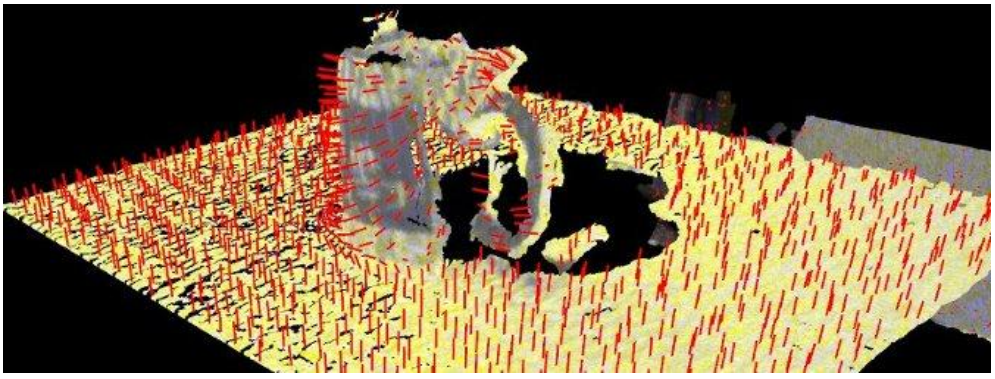
Az alkalmazás jövőbeli, kiaknázatlan lehetőségei igen széleskörűek. Bár a szoftver képes több irányból elmenteni a pontfelhőt, azokat összeilleszteni nem képes, továbbá egy léptetőmotoros megoldásban a tárgyasztal forgatását is maga végezhetné.

Megjelenítés szempontjából lehetőség van a mélység térképen kívül a pontfelhő interaktív ablakban való megjelenítésére is. Egy ilyen megoldást eszközölve, újabb lehetőségek tárulnak elénk a feldolgozásban is. Adattípusok terén a gyakorlatban leginkább használatos fájlformátumokba való mentés nagyban javítana a felhasználhatóságon. Ilyen formátumok az STL, a VTK, és az OBJ.

3.6.1 Megjelenítés

A pontfelhőt, a felületetet és az éleket megjeleníteni az elsődleges, ha dolgozni akarunk velük. Ebben ismét segítségünkre lehet a Point Cloud Lybrary. Saját vizualizációs modulja segítségével könnyedén megjeleníthetünk akár egyszerre több nézőpontot is. Valós időben válthatunk át a pontfelhők megjelenítéséből a hálok vagy a teljes felület kirajzolására, természetesen csak abban az esetben, ha azokat az adott nézethez hozzá rendeltük.

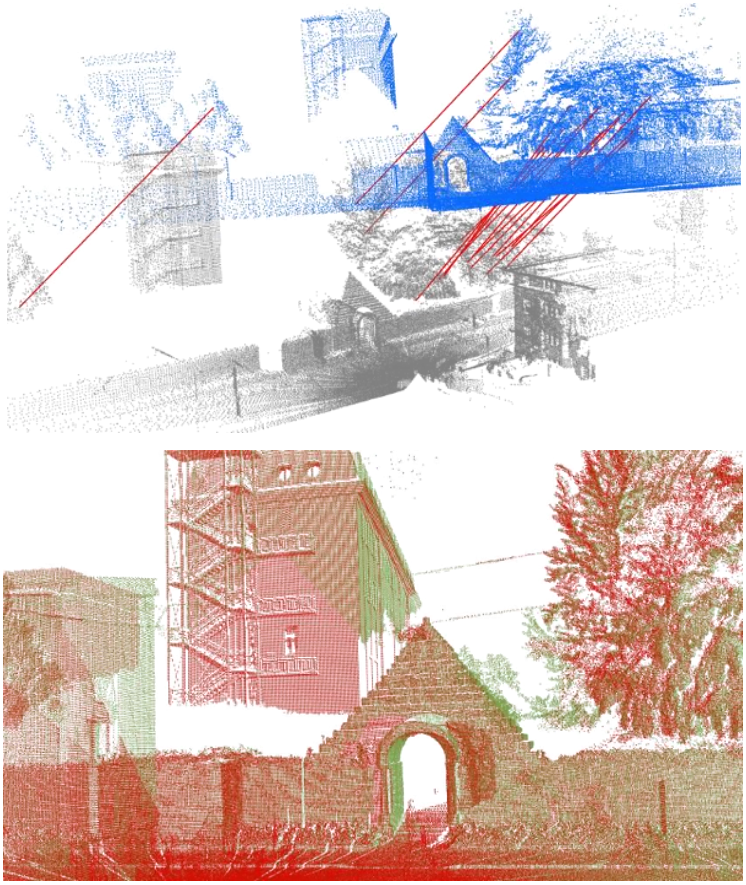
A megjelenítéssel együtt új funkciókat is építhetünk a programba. Az egér és a billentyűzet figyelésével különböző szegmentációs eljárások építhetők a programba. Az egérkurzort és billentyűkombinációkat alkalmazva létrehozható lenne olyan eljárás, melynek során a kijelöléshez hozzá vagy éppen elvehetnénk elemeket. Ezek külön vonatkozhatnak csak pont, csúcs, él vagy felület.



6. kép PCL Viewer működés közben

3.6.2 Multiview illesztés

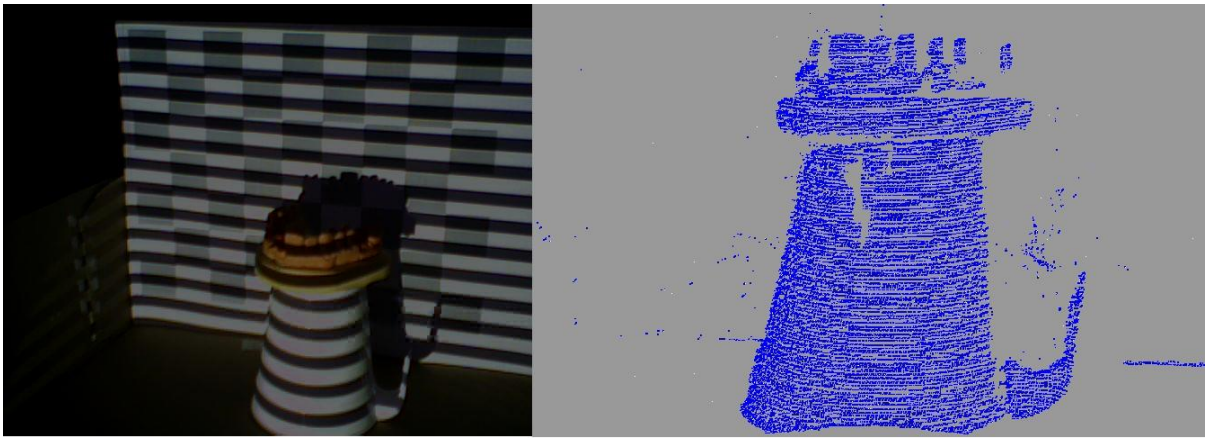
Több oldalról készített felvételek összeillesztésére a felületdigitalizálás során minden esetben szükség van. Ezt hivatott megvalósítani a Point Cloud Library regisztrációs modulja. Működése közben két ponthalmazt hasonlít össze. Azonosítja a megfelelő pontokat amelyek között csökkenti a távolságot és a megfelelő mértékben elforgatja, beigazítja a helyére(6.kép). A folyamat addig folytatódik míg nincsenek a megadott tűrésen belül a pontok.



7. kép - Multiview illesztés

3.7 Eredmények

A szkennert egy foglenyomat beolvasásával teszteltem látható, hogy elég sok pontot megtalált. A sötétebb részeit a lenyomatnak nem érzékelte a feldolgozás során így azok hiányoznak. A gyakorlatban erre szkenn púdert használnak, amely egy nagyon vékony matt fehér réteggel vonja be a mintát.



4. ábra - Foglenyomat szkennelése és eredménye

4 Anyagszerkezet vizsgálat CT felvételeken

Az alumínium-kerámia kompozit anyagvizsgálata során a cél a kerámia héjak detektálása volt. A megoldás gyakorlati hasznossága, hogy az ebből származó adatok felhasználásával véges elemes szimulációk futtathatók a kompozit további kutatásában. A feladattal TDK dolgozat keretein belül volt szerencsém foglalkozni. A feladat témavezetője Kozma István és Dr. Zsoldos Ibolya – társ-témavezető – volt. A helyi megmérettetésen a Dorogi Gábor Mechatronikai Mérnök hallgatóval készült közös munkánkat a zsűri a megtisztelő első hellyel illette a Gépészeti Szekcióban. A továbbiakban a dolgozat általam készített részeit ismertetem.

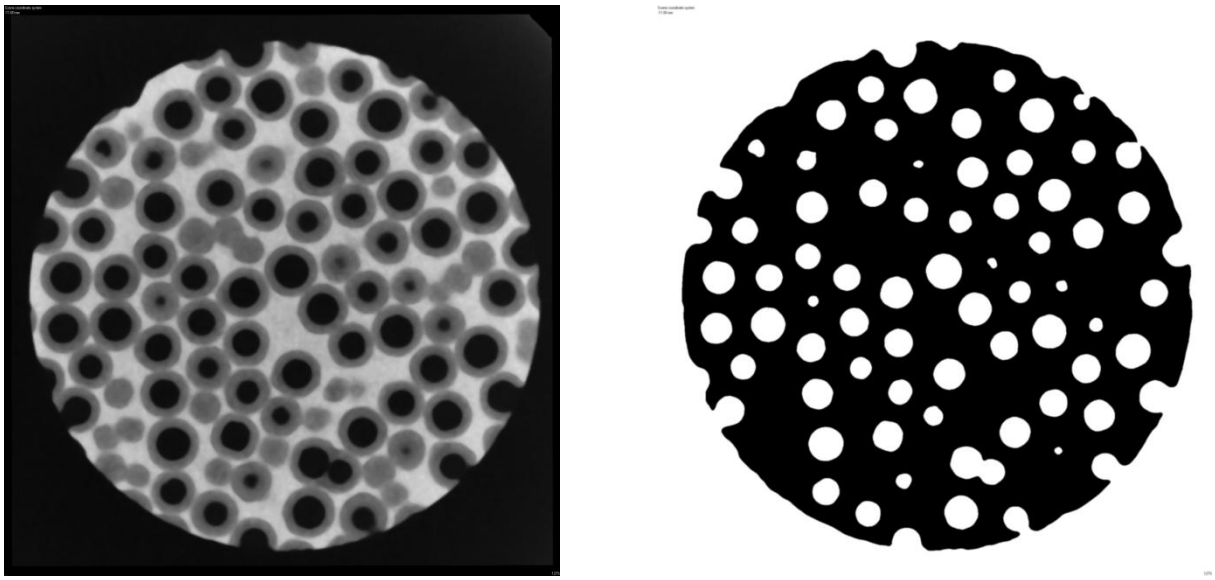
4.1 Meglévő algoritmus kiegészítéseivel tervezett módszer

Mindenekelőtt megpróbáltam a Hough Circle algoritmus önálló használatából a lehető legjobb körív-felismerést elérni.

A CT felvételek kiértékelése Hough transzformáció segítségével, feladat specifikus eljárások bevezetését vonta maga után, mind a szeletek egyenkénti kiértékelése, mind a teljes analízis szempontjából. A szeletek fő problémái a voxelek pixelekké alakítása során keletkező elmosódás, az összeérő alakzatokból adódó szegmentálás szükségessége és, hogy az alakzatok csak körszerűek, nem pedig körök. A teljes analízis szempontjából gömbök detektálása a célunk, ehhez azonosítani kell a gömbökhöz tartozó köröket a szeleteken. Utóbbi probléma magában hordozza annak megoldását is.

4.1.1 Szeletek

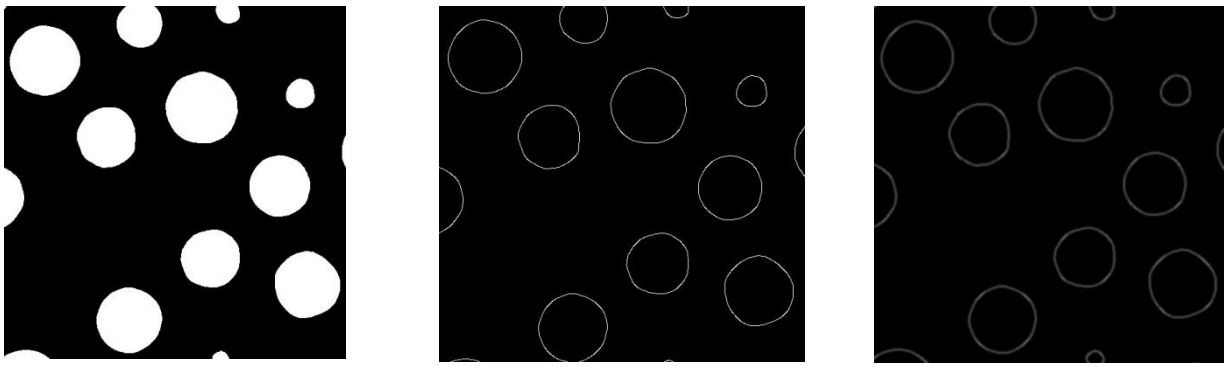
Az elmosódások CT-képek készítésekor a voxelek jelegéből adódnak. Az elmosódásokat képfeldolgozási módszerekkel könnyedén javíthatjuk. Első lépésben threshold algoritmus segítségével kiemelhetjük a számunkra fontos információt tartalmazó részeit a képnek és elkülöníthetjük a lényegtelen és zavaró objektumokat (5. ábra). Az így kapott képen a sötét körök egységes, homogén színt kapnak, akárcsak a világos fém részei is.



5. ábra Az eredeti kép és a threshold algoritmus alkalmazása utáni

Második lépésként a threshold-dal szegmentált képeken meg kell keresnünk a határvonalakat. Erre a rendelkezésre álló él-kereső algoritmusok közül a Canny félét találtam a legjobbnak. Az így kapott képen, fekete alapon fehér színnel egy pixel vastagon kirajzolódnak a szegmentált alakzatok körvonalai.

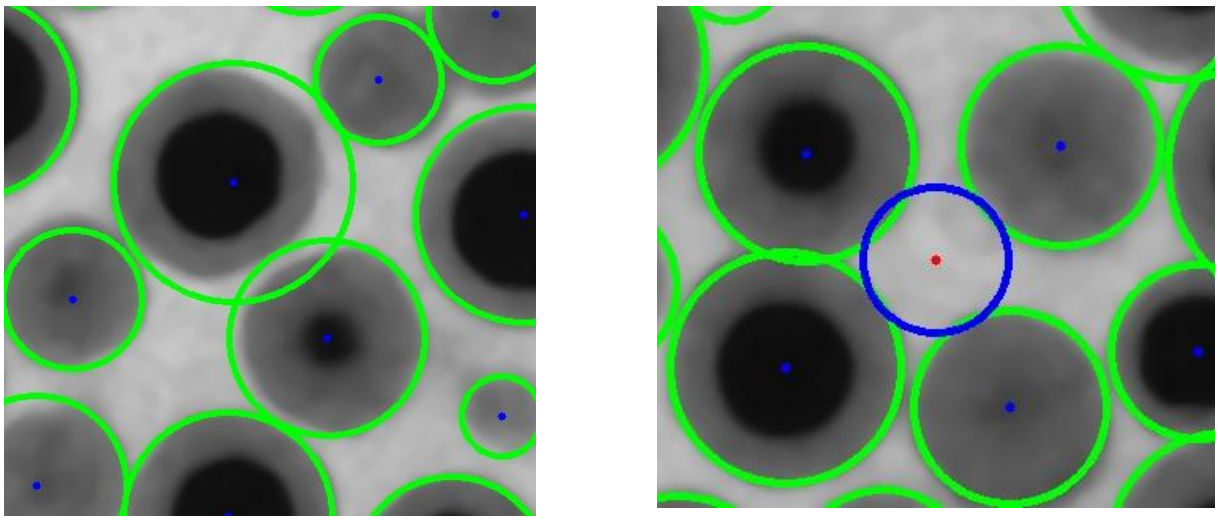
Ezzel még nem fejeződtek be az átalakítások, mivel az egy pixel vastag vonalak nem elég simák, nagyon erősek a szögek az ívelt vonalakon, illetve előfordulhatnak felesleges élek is, ami hibás keresést eredményez. Utolsó lépésként tehát el kell simítani az éleket, méghozzá a lehető legkisebb mértékben, hogy az adat ne sérüljön az eljárás folyamán. Ehhez Gauss-féle élsimítást alkalmaztunk 3x3-as mátrix konvolúcióval (6. ábra).



6. ábra: Threshold algoritmus utáni kép, Canny algoritmus utáni kép, élsimított kép

4.1.2 Hough Circle – Problémák és megoldások

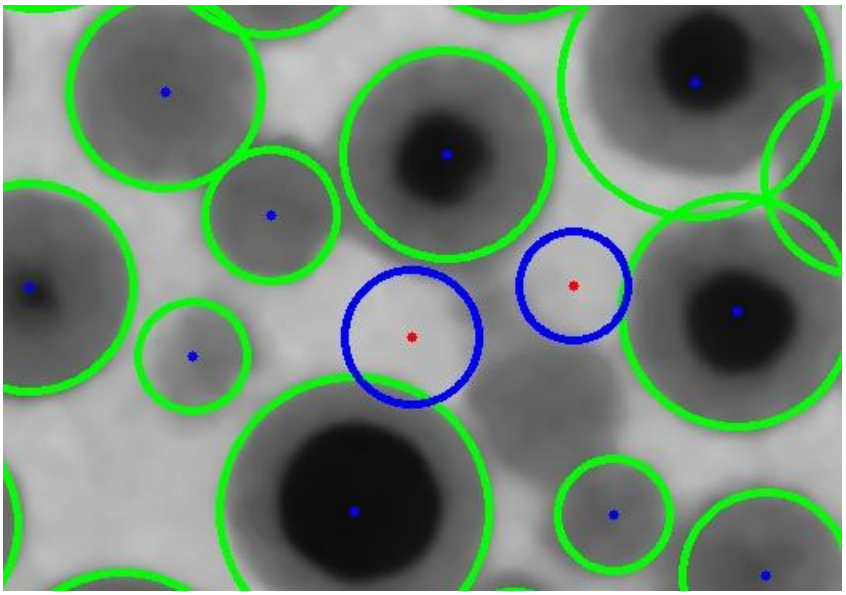
A CT rétegfelvételek (szeletek) az előzőekben ismertetett módosítások után alkalmassá válnak, hogy köröket keressünk rajtuk. A Hough transzformációból képzett körkereső algoritmus (lásd: fejezetcím) megfelelő beállításokkal megközelítőleg jó eredményeket hoz, viszont bizonyos esetekben nincs egészen jó helyen a kör középpontja, ami hibás rádiuszt is okoz Másik hiba, amikor az analízis olyan kört talál, amelyet nem mi nem szeretnénk vizsgálni (7. ábra).



7. ábra: Rosszul detektált középpont, fals kör

Az első probléma oka, hogy a felvételeken szereplő alakzatok csak körszerűek, így az algoritmus az egész alakzat egy olyan ívéhez igazítja a kört, amelyet először megtalál. Hangsúlyos, hogy amit először talál, mivel ha talál egyet, ahhoz bizonyos távolságban nem keres újat. Ez egy olyan beállítás, amelyet a körkereső algoritmus paraméterként kap, és azért van rá szükség, mert ellenkező esetben rengeteg kört képes illeszteni egyetlen alakzathoz.

A szükségtelenül detektált körök előfordulását az előző bekezdésben említett eljárás nem tudja megszüntetni, mivel egészen más okból jönnek létre. Ezek valójában tényleg körök, melyek valójában a képen vannak, csak az emberi szem hajlamos elsiklani felettük, mivel ezek úgy jönnek létre, hogy a körök közötti területen rajzolódik ki, vagy egyszerűen olyan pozíciót vesznek fel egymáshoz képest, ami egy új, nagyobb kört alkot.



8. ábra: Rosszul detektált körök azonosítása

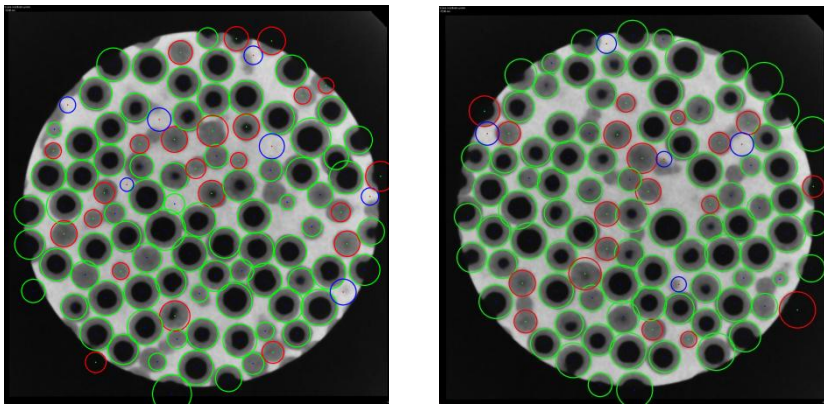
Szerencsére ezeknek a közös tulajdonságoknak köszönhetően szűrhetők is lesznek, ha a kerámiagömbök sajátosságait ismerjük: A nagyobb körök egyszerűen a méretüknél fogva, kiugró értékként azonosíthatók. A kisebbek pedig minden esetben nagyobb részt világos pixelekből álló régiókat jelölnek ki az eredeti képen, és emiatt könnyű elkülöníteni őket.
A módosítások után már, csak a vizsgálat célját meghatározó alakzatokhoz tartozó köröket kapjuk meg minden CT felvételen.

4.1.3 Analízis

A metszetek egyenkénti kiértékelése után az együttes vizsgálatukból meghatározzuk a fém mátrixban lévő kerámia gömbök elhelyezkedését és méretét.

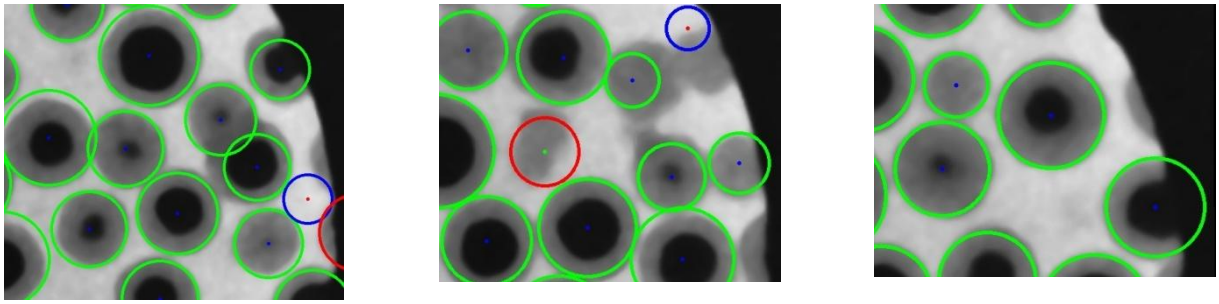
A gömbök azonosítása az egymás mellett lévő szeletek kiértékelésével történik. Megvizsgáljuk az egyes szeleteket úgy, hogy a szomszédos szeleteken talált köröket összehasonlítjuk. Ha egy bizonyos intervallumon belül egy szomszédos szeleten is megtalálható egy kör, akkor azokat egy kerámia gömb részének tekinthetjük. Miután ezt megismételjük minden szeleten, a legtöbb, az előzőekben már detektált kör valamelyik detektált gömb részét képzi. Mivel minden gömbről legalább két szeletünk áll rendelkezésre. Két metszetből bármely gömb adatai elemi geometriai módszerekkel számítható.

Az analízis során megmaradnak olyan körök a metszeteken, melyek egy gömbhöz sem lettek hozzárendelve. Ezek az ún. „árva körök” a legnagyobb számban az első és az utolsó szeleten találhatók (9. ábra). Ennek egyértelmű oka az, hogy nekik csak az egyik oldalról van szomszédos szeletük, így olyan esetben mikor egy gömb az első szeleten még szerepel, de a másodikon már nem, – mivel valahol a kettő között van a gömb legalsó pontja – akkor az árva kör jelenléte nem jelenhet hibát. A probléma akkor van ha egy árva kör nem egy szélső szeleten jelentkezik (10. ábra). Ennek több oka is lehet. Az első, hogy olyan apró a gömb hogy csak egyetlen szeleten volt egyértelműen detektálható. Lehetséges még az is, hogy a fent említett a metszetek méréséből adódó középponti hiba oly mértékű, hogy az a teljes analízisben használt intervallumon kívül esik.



Zöld: Gömbként detektált
Piros: Árva kör
Kék: Rosszul detektált kör

9. ábra: Első szelet, utolsó szelet



10. ábra: Megelőző metszet, árva kör egy közbenső metszeten, következő metszet

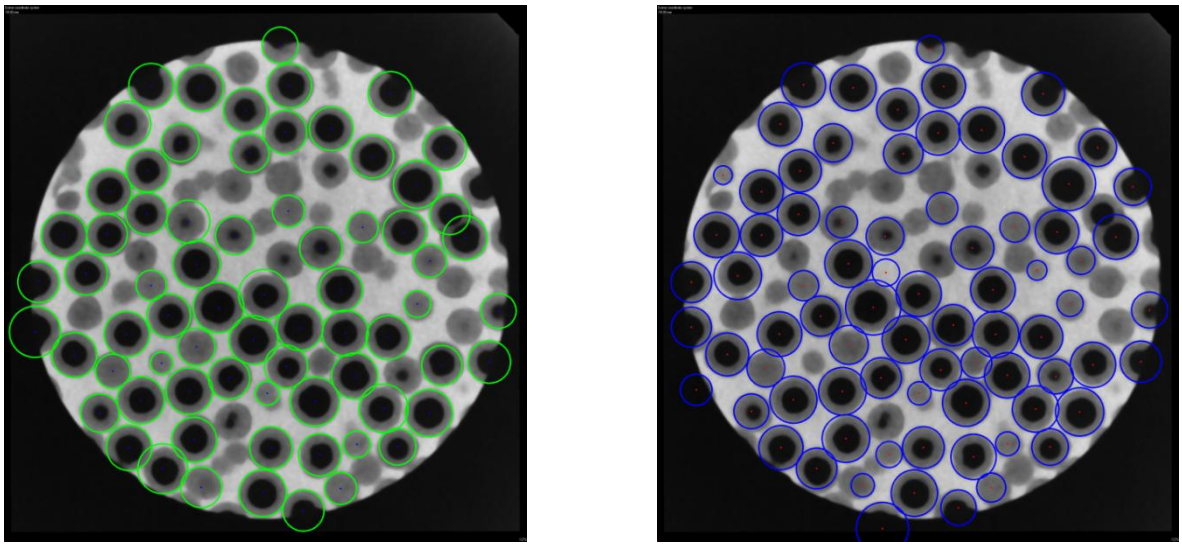
4.1.4 Adatbázis

Az analízisből származó minden adat egy adatbázisba van tárolva. A felépítését tekintve úgy tárolja az adatot, hogy képes legyen egy teljes analízis adatait teljes mértékben elkülöníteni.
Az adatbázis legkisebb tárolt adata a kör. Ezek metszetekhez és gömbökhöz vannak rendelve, azok pedig analízishez. Kezdetben minden kör adatait rögzítjük, a fals és az árva körökét is. Később csak azoknak a köröknek adunk azonosítót, amelyek egyértelműen gömbmetszetként értelmezhetőek a szomszédos metszetek kiértékelése során.

4.1.5 Analízisek összehasonlítása – legnagyobb és legkisebb detektálható körök.

Az adatbázisnak köszönhetően, ugyanazon CT felvételeket más-más beállításokkal futtatva is analizáltuk, az eredményeket összehasonlítottuk. Az összehasonlításban az első vizsgált analízisnek a módszer legjobb eredményt hozó analízisét választottuk (szemrevételezéssel értékelve). Másodiknak pedig egy olyant, ami ugyanazokkal a beállításokkal fut egyet kivéve. Ez a paraméter két detektált középpont közötti minimális távolságot határozza meg. A második analízisben ez a lehető legkisebbre, azaz egyre van állítva. Az így keletkezett analízisben minden körszerű objektumhoz több hozzá illeszkedő kör tartozik.

Ha rendelkezünk két ilyen adathalmazzal, könnyedén megkereshetjük az egy objektumhoz tartozó legnagyobb és legkisebb köröket. Ezek segítségével korrigálhatjuk az első analízis értékeit, attól függően, hogy mire van szükségünk. A legkisebb detektálható kör az objektumon belül, a legnagyobb az objektum körül helyezkedik el. Kettejük átlagából számított középpont és sugár bizonyos mértékig alkalmas a módszer hibáinak kijavítására (11. ábra).



11. ábra: Első analízis eredmény, átlagolásból származó eredmény.

4.1.6 A módszerből levonható következtetések

A módszer a feladat jellege miatt, a várakozásokkal ellentétben, nem hozott egyértelműen elfogadható eredményeket. A legnagyobb problémát az egymást metsző körök jelentik. Ezeknél az eseteknél egyik javító módszer sem változtatott. A gömbhéjak metszetén a belső körök keresésekor a Hough Circle körkeresés jó eredményt ad. Ennek oka a levegő és kerámia közötti erős kontraszt, így a belső körök detektálása jó alapja lehet egy másik módszernek.

A további módszereknél ezért abból indultunk ki, hogy a belső körök a Hough Circle algoritmussal meghatározva már ismertek, és csak a küldő körök keresésére fejlesztettünk saját algoritmusokat.

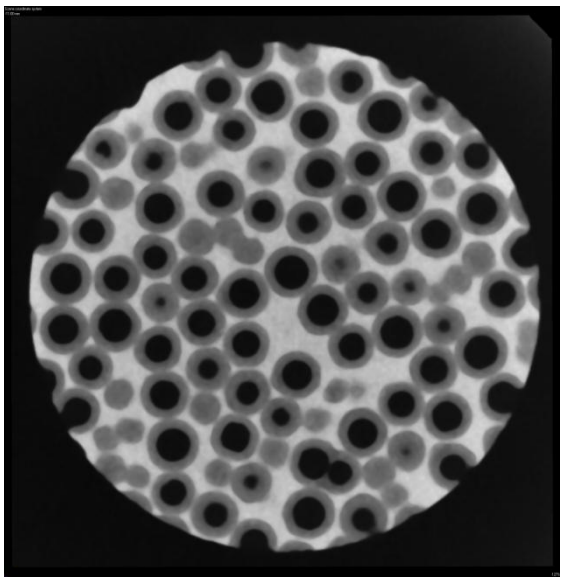
4.2 Rádiuszinkrementálás módszere

A rádiusz inkrementálás módszerének tervezésekor a fő szempont az volt, hogy az algoritmus a lehető legjobban detektált középpontból kiindulva növelje a sugarat és közben korrigálja az origó koordinátáját.

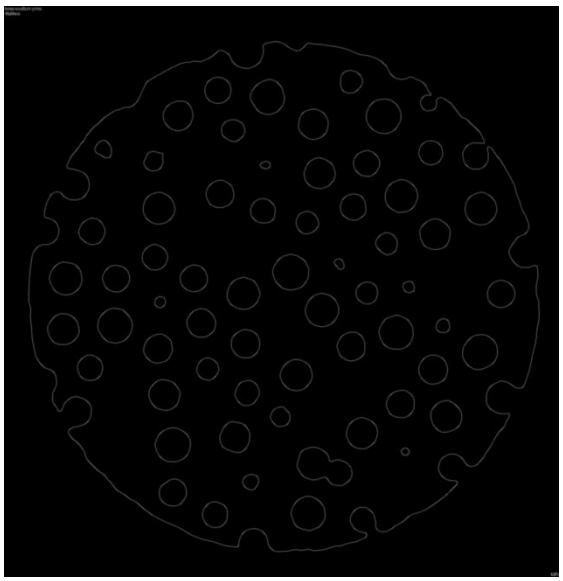
4.2.1 Belső körök

Az algoritmus első feladata, hogy a képet úgy alakítsa át, hogy a kerámia kompozitban lévő apró légbuborékokat fel lehessen ismerni, azaz szegmentálja őket. Azért esett erre a választás, mert a kerámia és a levegő részek között a legerősebb a kontraszt, ebből következik, hogy könnyebben detektálható.

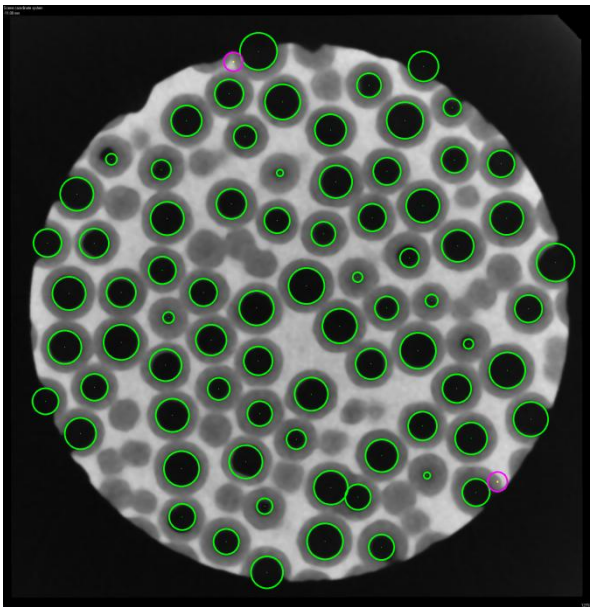
A belső körök detektálására ugyanúgy, előbb threshold eljáráson majd Canny élkeresésen és Gauss simításon esik át a CT szelet, akárcsak a meglévő algoritmusok kiegészítésével alkalmazott módszernél. Beállításokban különbözik annyiban, hogy a threshold alsó határértéke a kerámia és a levegő, színintenzitásai közé esik. Az átalakítás után a belső körök egyszerűen detektálhatók (12. ábra).



12.a, ábra Normál kép



12.b, ábra Átalakítás utáni kép



12, c. ábra Detektált belső körök az eredeti képen

4.2.2 Rádiuszinkrementálás

Miután meghatároztuk a kiindulási középpontokat és sugarakat, elkezdődhet a sugár növelése. A növekedést a megfelelő pontban le kell állítani, meg kell határozni, hogy hol van a vége kerámia gömbhéjaknak, és hol kezdődnek a fém részek. A kettő között található egy átmeneti rész is, amit szintén figyelembe kell venni a leállításkor.

Az algoritmusunk minden egyes inkrementáláskor a körvonalon lévő pixelek színintenzitását vizsgálva dönti el, hogy a körvonal elérte-e a kerámia szélét. Ezeket az intenzitásokat előfordulásuk arányában vizsgáljuk. Megkülönböztetünk négy ilyen számosságot.

- Magas intenzitásúak – intenzitás <130 *
- Alacsony intenzitásúak – intenzitás <= 130
- Extrém magas intenzitásúak – intenzitás > 163**
- Extrém alacsony intenzitásúak – intenzitás < 20***

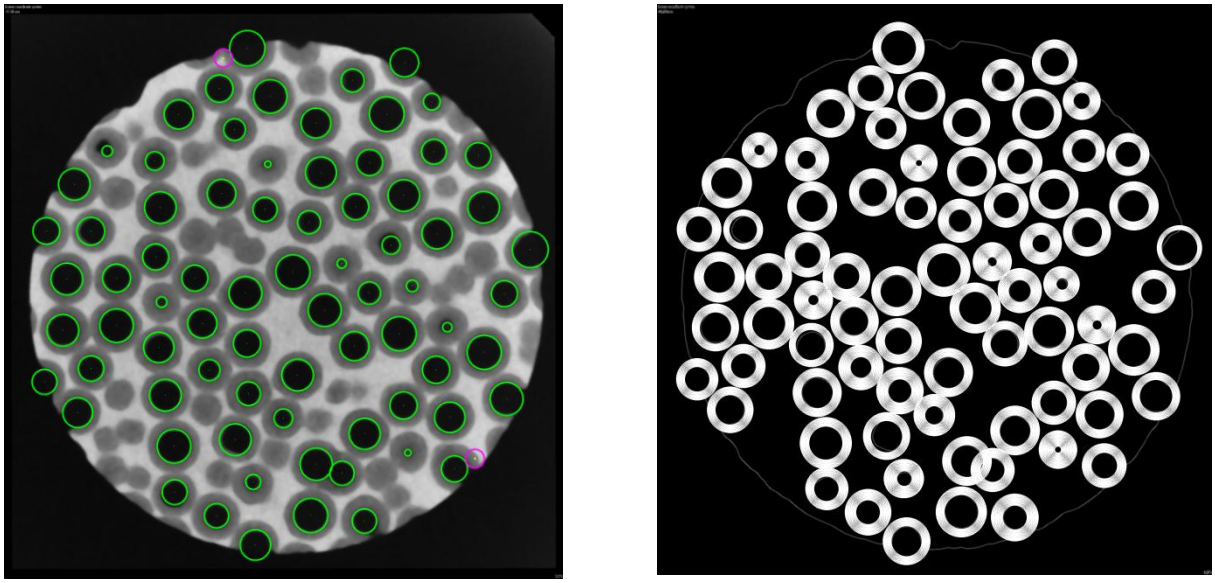
* A 130-as intenzitás a kerámia és fém részek közötti átmenetben van.

** A 163-as intenzitás már egyértelműen fém részeknél jelentkezik.

*** A 20-as intenzitás már egyértelműen levegő részeknél jelentkezik.

Az eljárás ezen intenzitások számosságának egymáshoz viszonyított aránya alapján számol és dönti el, hogy elértük-e a kerámia szélét. A magas és alacsony intenzitások arányának el kell érni az 1:1 arányt. Az extrém nagy és alacsony intenzitásoknak az 1:5, és az extrém kicsi és alacsony intenzitásoknak a 4:5 arányt. (13. ábra)

Utóbbi kettő csak akkor érvényes, ha már találtunk legalább egy extrém magas körvonalon lévő pontot. Egyébként csak az első aránypárt vizsgáljuk. Ha találtunk ilyen extrém magas pontot és ez a két intenzitás arány nem megfelelő, akkor tovább fut a keresés, mivel ez azt jelenti, hogy sok alacsony intenzitásunk van, de mégis találtunk egy egyértelműen fém részt reprezentáló pontot. Ez azt is jelenti egyben, hogy a középpont rossz helyre került, így előbb korrigálnunk kell és csak az után szabad tovább növelni a rádiuszt, de csak akkor, ha az még mindig szükséges.



13. ábra: Belső körök - inkrementált kép

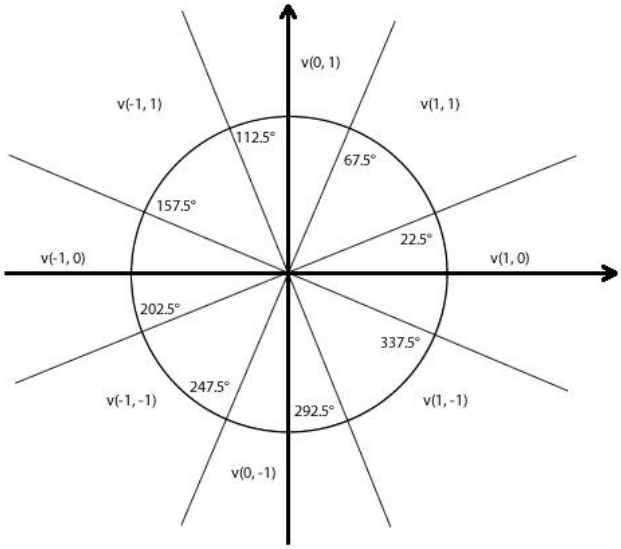
4.2.3 Origó korrekció

Amikor a gömbhéj külső és belső köre nem koncentrikus, a külső és belső körök középpontja nem esik egybe, akkor a külső kör középpontját javítani kell. A korrekció úgy zajlik, hogy a körvonali pixelek alacsony intenzitású pontjai felé egyszerre csak a lehető legkisebb léptékben mozduljon el. Ez után a vezérlést az inkrementálás visszakapja és eldönti, kell-e még korrigálni a középponton.

Az irányt úgy határozzuk meg, hogy a középpontból az alacsony – 130–nál kisebb – intenzitások felé mutató vektorok eredőjét számoljuk ki. Ennek az eredőnek meghatározzuk a meredekségét, ami egyenlő a vektor szögének tangensével.

Ezt a tangens értéket alapul véve nyolc irányba osztjuk be az elmozdulást 45°-os intervallumokban.

- 0° - 22.5° és 337.5° - 0° intervallumban: v(1, 0) szerinti elmozdulás
- 22.5° - 67.5° intervallumban: v(1, 1) szerinti elmozdulás
- 67.5° - 112.5° intervallumban: v(0, 1) szerinti elmozdulás
- 112.5° - 157.5° intervallumban: v(-1, 1) szerinti elmozdulás
- 157.5° - 202.5° intervallumban: v(-1, 0) szerinti elmozdulás
- 202.5° - 247.5° intervallumban: v(-1, -1) szerinti elmozdulás
- 247.5° - 292.5° intervallumban: v(0, -1) szerinti elmozdulás
- 292.5° - 337.5° intervallumban: v(1, -1) szerinti elmozdulás
- A kör középpont eltolások addig folytatódnak, míg az inkrementálás úgy nem látja, hogy megfelelő helyen van a középpont. (15. ábra)

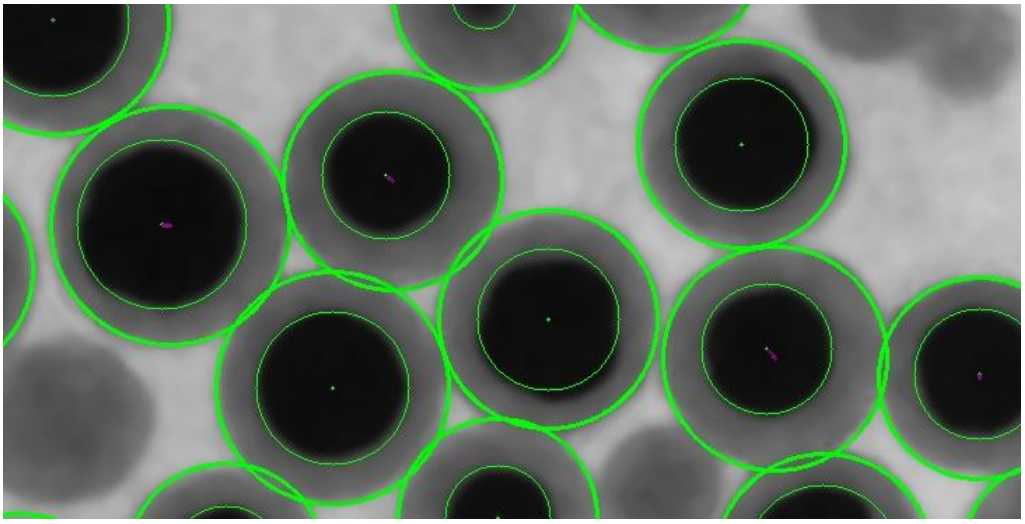


14. ábra - Az irányvektorok intervallumai

4.2.4 A módszerből levonható következtetések

Ezzel a módszerrel kapott körök jól illeszkednek, olyan szélsőséges esetekben is, amikor az egyes alakzatok nagyon közel vannak egymáshoz, vagy metszik egymást, vagy amikor a belső légbuborék középpontja nem egyezik meg a külső héj origójával. A módszer a metszeti keresésen javít, a metszetek együttes vizsgálata, így még pontosabbá válik.

A módszer működésénél fogva csak olyan alakzatokkal dolgozik, melyeknek létezik belső körük. A többi alakzat detektálása is lehetséges, akkor, ha a threshold alsó értékét a kerámia és fém területek közti színátmenet legsötétebb értékére állítjuk

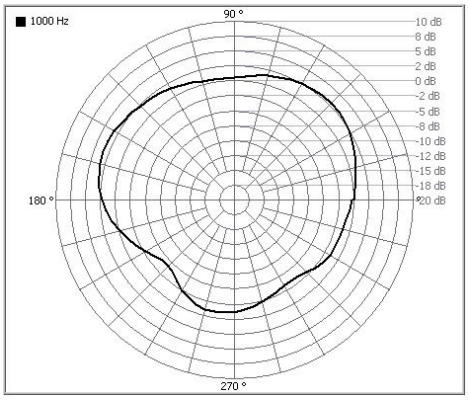


15. ábra – Az elmozdulások.

5 Iránykarakterisztikák rekonstrukciója különböző frekvenciákon

Ebben a feladatban a cél az volt, hogy az emberi fül által hallható frekvenciák iránykarakterisztikáját minden irányból három dimenzióban jelenítsük meg felületként. Az elkészült modell minden esetben a bal fülre értendő frekvenciák karakterisztikáját mutatja úgy, hogy a középponttól az adott irányba a karakterisztika erősségével megegyező mértékkel távolodunk. A modellek közepén egy fej található a megfelelő irányba pozícionálva. A megalkotás során a szempont a szemléltetés, lehetőség szerint úgy, hogy jól láthatóan ábrázoljuk számítógépen és egy másik végeredményként nyomtatható is legyen műanyagból.

Az adatok kinyerése mérések folytán történt. A mérés úgy zajlott, hogy először függőlegesen mérte végig a fejet úgy, hogy pont a fejjel szemben indult a mérés majd 5°-onként elindult felfelé körbe a műfej körül, míg vissza nem ért a kiinduló állapotába. Ezután egy vízszintes tengely körül forgatva az egyik irányba 10°-ot folytatódott a mérés. Ezt a második forgatási tengelyt úgy kell elképzelni, hogy egy pontja a tarkó közepe egy másik pedig az orr. A forgatás addig folyt az egyik irányba, míg el nem érte az összesen 90°-os elfordulást. Így a fejet körülvevő tér fele meg lett mérve. Ezt követően visszatértek a kiindulási ponthoz és elindultak a másik irányba újabb 90°-nyi elfordulásig megismételve. Így a fejet körülvevő teljes tér feldolgozásra került. A elforgatásoknak köszönhetően a 90° - os vízszintes síkról két mérési adat is rendelkezésre áll.



16. ábra - 1kHz frekvencia iránykarakterisztikája a függőleges síkon

A feldolgozás széles tartományon történt a 125Hz – től a 16kHz – ig bezáróan. Az adatokat pontosvesszővel elválasztott CSV fájlban bocsátották rendelkezésemre.

5.1 Pontfelhő megalkotása

Első lépésként az adatok előkészítését végezzük el. Mivel az adatok nem a megszokott formában elérhetők, ezért olyan sorrendbe kell őket tenni, hogy feldolgozhatóvá váljanak. A jobboldalra való forgatásból kiindulva az egymás alá ez után a 100°-os következne ám a következő felvétel már más irányban történt, ezért a következő a másik irány 90°-os mérési eredménye majd ezután csökkenő sorrendben haladva a nulláig. Az adatok viszont így tükrözve lennének ám ennek a megoldását már nem az előkészítési fázisban érdemes megoldani.

Az előkészítés során egy lineáris interpolációt is érdemes alkalmazni a 10°-os lépések sűrítése érdekében. Mivel a feladat során a fő szempont a szemléltetés ezért ez csak segítségünkre van. Az interpoláció után minden irányba 5°-os szögben található a következő adat.

A pontok felvétele a pontfelhőbe úgy történik, hogy előbb kiszámoljuk egy síkra csak az *x* és az *y* koordinátákat majd a *z* koordinátát 0-ra állítjuk. Ezek után csak ezt a síkot kell elforgatni a megfelelő szögben. A megfelelő szög a 90°-ig egyszerűen 5°-onként nő. A 90° után azaz a először a 95°-nál az adatunk egy tükrözött adat, ezért 175°-ot adunk az elforgatáshoz hogy elérjük a 270°-ot ami a 90°-nak felel meg ha a másik irányba forgatnánk a síkot ami pont a soron következő a feldolgozásban.

```
double v = rowData[j].toDouble();
pcl::PointXYZ basic_point;
basic_point.x = cosf(pcl::deg2rad(j*5.0))*(20+v);
basic_point.y = sinf(pcl::deg2rad(j*5.0))*(20+v);
basic_point.z = 0.0;
```

:

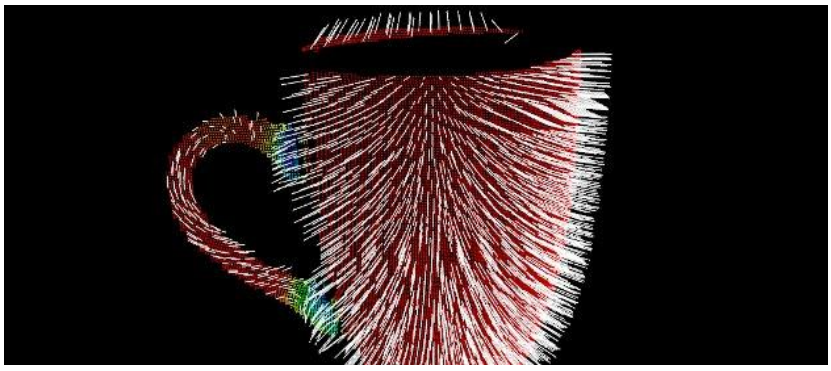
```
// x szerint
basic_point.x = _x;
basic_point.y = _y * cosf(pcl::deg2rad(alfa)) - _z * sinf(pcl::deg2rad(alfa));
basic_point.z = _y * sinf(pcl::deg2rad(alfa)) + _z * cosf(pcl::deg2rad(alfa));
basic_cloud_ptr->points.push_back(basic_point);
```

8. kép - A koordináták kiszámítása

5.2 Felület leképezése

Első feladatunk, ha pontfelhőre felületet szeretnénk illeszteni, hogy meghatározzuk a pontokhoz tartozó normálisokat. A normálisokat a gyakorlatban magukra a felületekre lehet kiszámítani, mivel a normális a felület egy pontjához tartozó érintősíkra merőleges vektor. A normálisokat pontfelhőből csak becsülhetjük. A becslésre a Point Cloud Library NormalEstimation osztályát használjuk.

Miután létrehoztuk az első lépésben meg kell adni neki a pontfelhőt, amivel dolgozhat. A feldolgozás során egy kd-fa adatszerkezet segítségével keresi meg a normálisokat. A fát rendeljük az objektumhoz üresen. A keresés közben majd feltöltődik, de mivel nincs felület amin kereshetnék – azaz a pontok nincsenek összekötve egymással – ezért ezzel helyettesíthetjük azt. Ez után a keresési rádiuszt kell megadni az objektumnak. Ez azt jelenti, hogy csak a keresési távolságon belül lévő szomszédjait veszi számításba egy adott pont normálisának becslésekor.
Ezt követően kiszámolja a normálisokat. Először veszi a legközelebbi szomszédjait. Kiszámítja a felületi normálist a szomszéd és pont alapján, majd ha leellenőrzi, hogy merre néz a szomszéd normálisa. Ha az ellentétes a pontéval, akkor épp a rossz irányba mutat, ezért megfordítja.



9. kép - Normálisok a pontfelhőn

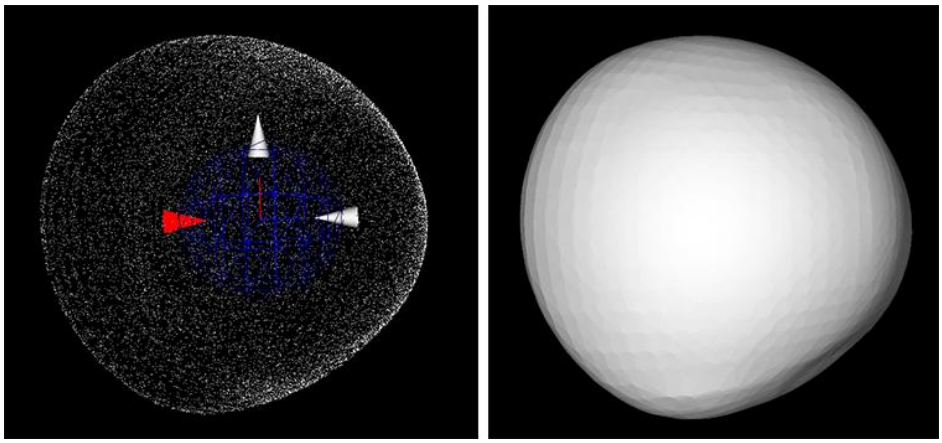
A Normálisok becslése után konkatenáljuk a pontfelhőt és a hozzá tartozó normálisokat, hogy egy adathalmazban legyenek. A meglévő adatokat többféleképp felhasználhatjuk felület rekonstrukcióra. Számos algoritmus ismert ebben a témában, de egyik sem biztosít hibátlan eredményt.

A pontfelhőt konkáv hálóval is körül vehetjük. Mint neve is mutatja egy konkáv hálót fog a pontok köré illeszteni olyan módon, hogy belül ne maradjon pont. Probléma csak akkor van, amikor mégis léteznek a pontfelhőben olyan pontok, melyek belül vannak a többihez viszonyítva. Lehetnek ezek hamis pontok, mérésből vagy előfeldolgozásból származó hibák, ám teljesen elronthatják a rekonstrukciót.
Másik eset, amikor a tárgy belül üreges. Ilyenkor szegmentálni kell a belső pontokat együtthatóikkal együtt.

A hamis pontok leszűrésére használjuk a PassThrough osztály példányát. A szegmentációhoz a SACSegment osztály nyújt segítséget. Az szegmentációban talált elszeparálására pedig a ProjectInliners ad segítséget. Így normalizálódik a felhő és az összetartozó pontok az adatszerkezetben is egymás után találhatók.

A konkáv háló ez után az üreges alakzatokra is tudjuk alkalmazni. Az iránykarakterisztikák kiértékelése során, a magasabb frekvenciákon volt erre szükség. Míg alacsony frekvenciákat a fülünk szinte minden irányból megfelelően hallja, a frekvencia növekedésével ez nagymértékben megváltozik. Bizonyos frekvenciákat hátulról szinte nem is hall az ember. Ezek a sajátosságok a mérések eredményein is látszódtak és bemélyedéseket hoztak létre a felületen.

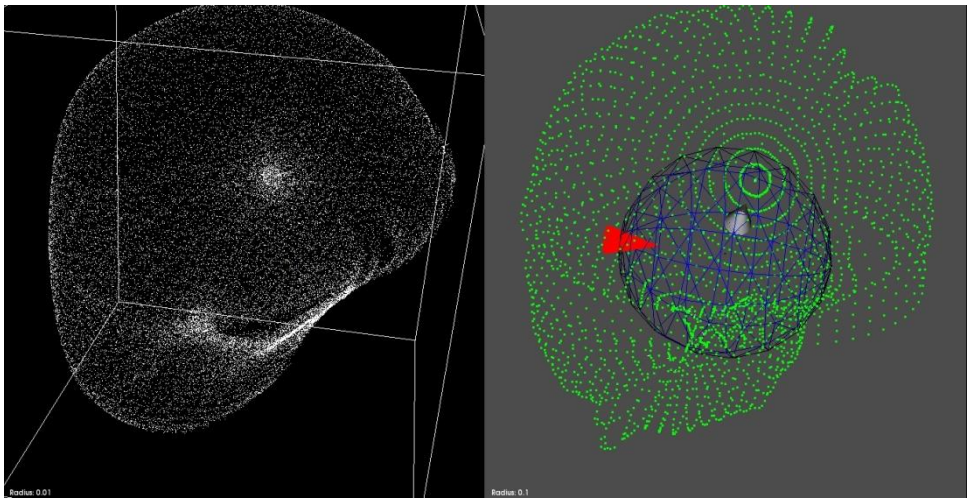
Konkáv hálót használva a felületrekonstrukció minden esetben teljes, azaz nincsenek benne lyukak, kihagyott területek. A z alábbi képen (10. kép) jól megfigyelhető a pontfelhő és a ráillesztett konkáv háló. A bal oldali képen egy sematikus objektum reprezentálja az emberi fejet. A vizsgált bal fül pirossal lett jelölve.



10. kép - Simított pontfelhő és hozzá tartozó konkáv hálós felületillesztés

A pontfelhő optimalizálása során sokszor kevés adatból dolgozunk vagy csak nem elég részletes az adathalmaz a számunkra. Ilyen esetekben az adatot simítani kell. Erre jó megoldás a MovingLeastSquares osztály. Az MLS objektum egy kd-fa segítségével egy beállított távolságon belül keresi a szomszédokat és egy szintén beállított lépéstávolsággal pontokat ad a felhőhöz. Ennek a lépésköznek körülbelül az ötödének kell lennie a keresési távolságnak. Az osztály a jellege miatt tartalmaz olyan metódusokat, melyek nem engedik, hogy egy pont normális nélkül jöjjön létre.

Mivel a feladat megoldása során fontos volt, hogy kinyomtatható legyen 3D nyomtatóval ezért igen fontos volt a felületi pontok sűrűsége. Ezért minden kipróbált felület rekonstrukciót megelőzött egy simítási fázis (10. kép, 11. kép).



11. kép Pontfelhő simítása MLS alkalmazásával

A GreedyProjectionTriangulation egy gyors algoritmus felület illesztésére. Feltételezi, hogy különböző sűrűségű területek egymástól elkülönülve léteznek és, hogy a sűrű területek simábbak a területek közti átmenetnél. A feladat megoldása során alkalmazható ez az eljárás ám még a simított és sűrű ponthalmazon is nagyon éles csúcsokat eredményezett. Ez annak az eredménye, hogy az algoritmus nem válogat sokat a lehetőségek közül és kapzsi módon a lehető legegyszerűbb megoldásokat választja. Olyan esetekben érdemes használni, amikor a feldolgozási minőség kevésbé fontos a futási időhöz képest.

5.3 Háromdimenziós megjelenítés

A Point Cloud Library segítségével a háromdimenziós objektumkat könnyedén megjeleníthetjük a PCLVisualizer osztály segítségével. A megjelenítőhöz több hozzáadhatunk pontfelhőt, normálisokat, poligont, poligon hálót, és egyszerű objektumokat, gömböt, téglatestet, vonalat, gúlát, 3D-s szöveget és síkot. Egy nézet többet is tartalmazhat ezekből, de össze is rendelhetjük őket. Ilyen esetben előbb megadjuk a pontfelhőt, aztán a pontfelhőhöz tartozó normálisokat. Fontos, hogy a normálisokat minden esetben hozzá kell rendelni egy ponthalmazhoz. Az objektumok mind egyedi névvel elláthatók, hogy hivatkozhatóak legyenek. Az egyedi neveknek köszönhetően renderelési paramétereket adhatunk meg az objektumainknak.

Ezen kívül több nézetablakot jelölhetünk ki. Ha több viewport-ot használunk, akkor megjelenítendő objektumok hozzáadásakor a nézetablakot is jelölni kell. A nézetek méretét és pozícióját is megadhatjuk. Ezt a tengelyeken elfoglalt kezdő és végpontjaival adhatjuk meg, de ezek az értékek nem koordináták, hanem nulla és egy közötti valós számok. A nézetek így nagyon sokszínűek lehetnek. Készíthetünk, ha kell egy nagyot és mellé egy oszlopba három kisebbet, hogy ezzel is teret adjunk a fontosabb adatoknak.

Használat közben is sok szolgáltatás érhető el a PCLVisualizer-ben. Megjeleníthetünk vonalzót, ami segít minket a méretek meghatározásában, de eltüntethetjük a felületet vagy a hálót is. Készíthetünk egy gombnyomásra képernyőkép mentés PNG formátumban, növelhetjük vagy csökkenthetjük a pontok méretét és átválthatunk két fajta sztereo 3D-s módba is.

5.4 Előkészítés nyomtatásra

A 3D nyomtatás egyre elterjedtebb a hétköznapokban is. A nyomtatáshoz szükséges fájlok létrehozására több különböző program áll rendelkezésre. A feladat egyszerűnek tűnhet ám a céljaink szerint a kinyomtatott objektumoknak valamilyen módon reprezentálniuk kell, hogy az ember bal fülével hallható tartományok iránykarakterisztikáit modellezzük. Ehhez a legegyszerűbb módszer egy emberi fejet belerakni a modellbe a megfelelő pozícióba.

Mivel ezt látnunk is kell a külső héj mindenképp átlátszó műanyagból kell, hogy készüljön a belső, ha van rá lehetőség készülhet nem átlátszó műanyagból. Mivel ez nem volt megoldható, ezért a fej része üreges maradt. Mivel a 3D nyomtaók az üregeket támaszanyaggal töltik a modelleket ketté kellett vágni a fej közepének mentén, hogy a támasz anyagot ki lehessen mosni nyomtatás után. A kinyomtatott darabokat összeragasztva az üregek jól láthatóak az átlátszó műanyagnak köszönhetően.

Ezeknek a modelleknek a létrehozásában az általam fejlesztett program biztosítja számunkra a pontfelhőket. Ezek a pontfelhők minden darabja az MLS algoritmus segítségével simított.

A további feldolgozás során megkaphatjuk, a ráillesztet felületet. Ehhez a MeshLab nevű programot hívjuk segítségül. Ebben a programban lehetőség van a számunkra nem kívánatos pontokat kijelölni és eltávolítani. Miután ezt megtettük az RIMLS funkciójának segítségével alkossuk meg a felületet. Az így kapott adatot exportáljuk STL formátumba. Ennél a pontnál készen vagyunk a felülettel.

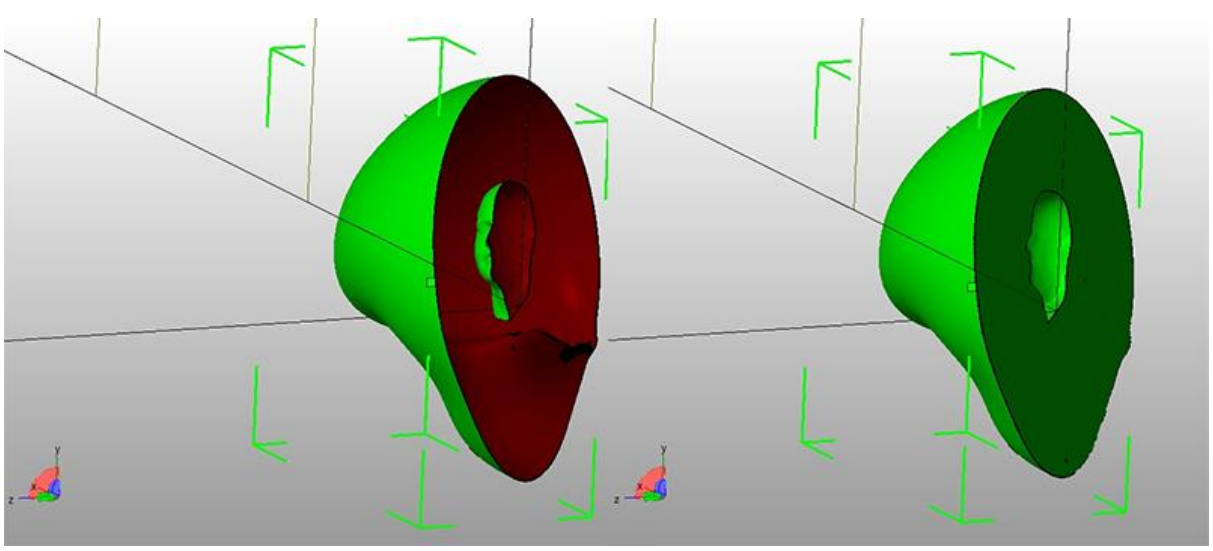
A következő lépésben a GOM Inspect programmal dolgozunk tovább. Első feladatunk a fej STL fájljának betöltése. Alakítsuk előbb át a fejet úgy, hogy beleférjen az objektumokba és hogy a megfelelő orientáltságú legyen, továbbá legyen olyan pozícióban, hogy a fülek a vízszintes tengelyen foglaljanak helyet. Mentés után ezt a fejet kell minden objektum megalkotásánál használnunk.

A jól pozícionált fej mellé ez után már betölthetünk egy modellt. Egy munkamenetben használva egyszerre fognak megjelenni a képernyőn. Készítsünk mindkettőről egy másolatot és párosítsuk őket. A párosítás során a egy-egy fej-modell párosból egyetlen összetartozó objektumot kapunk. Mindkettő ilyen modellt vágjuk ketté a megfelelő tengely mentén és

egyiknél a pozitív, a másiknál pedig a negatív oldalt tartsuk meg. Ez után kimenthetjük a modelleket. Ezek már csaknem készen állnak a nyomtatásra, de két nagyobb probléma van velük. Az első, hogy a vágások síkjában megszakad a zárt felület, a másik, hogy a fej normálisai nem az objektumból kifelé mutatnak (12. kép – bal oldal). Ez annak az eredménye, hogy az átalakítások során az eredetileg belső fele került a modell külső felére.

A problémát a Geomagic Stúdió segítségével oldhatjuk meg. Az inteligens kijelölési lehetőségeknek köszönhetően a fej egy kattintással kijelölhető, mivel külön objektumként úgymond lebeg a modell közepén. A normálisok megfordítása ugyancsak egyszerű feladat, mivel az eszköztár jól strukturált, könnyen megtalálható a kívánt funkció. Ha megfordítottuk a normálisokat, akkor a felület bezárása a marad hátra.

Az eszköztárban válasszuk ki a hídillesztő eszközt és állítsuk simára a fajtáját. Ezzel az eszközzel a vágási felület lyukas részét rácsozzuk be. Pár darab ilyen híd segítségével összekötjük a körvonalat a fej körvonalával. Ebben a fázisában már nem lebeg a modell közepén a fej, így a Fill Hole eszköz segítségével lezárhatjuk a felületet (12. kép – jobb oldal).



12. kép Normálisok megfordítása és felület zárása

A lezárt felületen érdemes még lefuttatni a Mesh Doktor nevű automata hibakereső és hibajavító funkciót. A Mesh Doktor megtalálja a dupla csúcsokat, a kisebb lyukakat, a túl éles szögeket és az átfedő hidakat. Ezt az előkészítési folyamatot minden modellre alkalmazzuk és ez után STL formátumban kimentve nyomtathatók is az objektumaink.

5.5 A nyomtatás

A nyomtatáshoz használt 3D nyomtató az Objet30 nevet viselő eszköz. A hozzá tartozó szoftver segítségéve a nyomtatóra ráhelyezhetjük a modellek képét. A szoftver ez után egy nagyon pontos becslést képes adni arról, hogy hány gramm műanyagra lesz szükség a nyomtatáshoz, külön a modellhez és külön a támaszanyaghoz. Becslést ad arra kifolyólag is, hogy mennyi időt vesz igénybe a nyomtatási procedúra.

13. kép - Objet30 Pro

Az Objet30 nyomtató működéskor először egy keverék réteget ken fel a támaszanyag és a nyomtatási műanyagból a tálcájára. Ez egy igen vékony fóliaszerű réteg, ami megvédi a tálcát. Ez után vékony rétegenként felkeni fotopolimert a megfelelő helyekre, visszafelé jövet az olvasófej mellé helyezett UV lámpa fényére megköt az anyag.
Ebből a mechanikából következik, hogy minél magasabb egy tárgy annál többször kell az olvasófejnek végigjárnia az útját. Így sok lapos tárgy kinyomtatása akár rövidebb lehet, mint egy magas tárgyé. Ezek ismeretében, ha az olvasófej NULL pozíciójához legközelebb kezdjük a nyomtatandó tárgyakat a munkaasztalra rakni, ügyelve arra, hogy a tárgyak legkisebb dimenziója legyen nyomtatáskor a magasság és, hogy arra az oldalára fordítsuk ahol kevesebb alátámasztást igényel, nagy mértékben növelhetjük a hatékonyságot, mind anyag költség mind gépóra szempontjából.

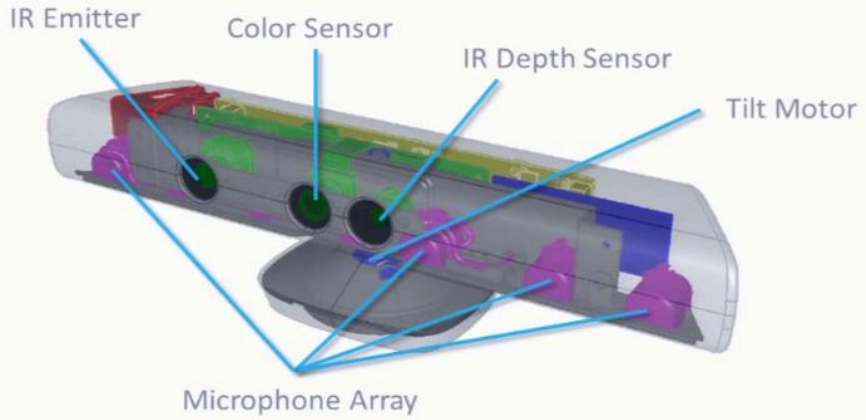
A támaszanyag olyan helyekre kerül ahol bár nincs pontja a modellnek, de a következő rétegek során nyomtatni fogunk fölé. Ennek köszönhetően mikor a nyomtatás ehhez a részhez érkezik lesz alatta támaszték, ami megtartja, míg megszilárdul. A támaszanyag mátrixa enyhén lyukacsos, rácsos szerkezetű, könnyen leszedhető gumiszerű anyag.

A kinyomtatott műanyagról, ha lemossuk, az alatta lévő műanyag mattabb lesz, mint az olyan részeken, ahol nem volt támaszanyag. Ez alól csak azok az esetek kivételek mikor matt felületűre nyomtatjuk a műanyagot. Ez a mi esetünkben nem lehetett opció, hiszen a belső üreg nem látszódna megfelelően egy matt hatású átlátszó műanyag belsejében.

14. kép - A kinyomtatott modellek

6 Kinect for Windows

A Kinect 2010 óta jelen van a piacon, először a Microsoft által fejlesztett konzolra az Xbox 360-ra jelent meg. Windowsos verziójának kiadására csak 2012-ben került sor. Eredetileg játékok vezérlésére fejlesztették így a háromdimenziós letapogatásra nem fektettek nagy hangsúly. A fő szempont a szkeleton felismerése és annak mozgásának a gyors lekövetése volt. A szkeleton az emberi test vázát hivatott reprezentálni. Minden nagyobb testrész megkülönböztetve az eszköz alkalmas az emberi test pozíciójának meghatározására a testrészek méreteinek elemzésére és a mozgásuk lekövetésére.



18. ábra - A Kinect szenzorai

Az eszközben a köztudattól eltérően csupán két kamera foglal helyet. Az egyik ilyen kamera egy átlagos 640x480 felbontásra képes RGB kamera. Ez csak az eszköz által látható objektumok színéért felel. Ha nem használjuk érdemes kikapcsolni. A másik kamera egy infravörös mélységérzékelő szenzor. A harmadik kamerának tűnő komponens egy infravörös jelet sugárzó emitter. Ennek feladata, hogy infravörös pontokat vetítsen az előtte lévő térre, ezeket a jeleket a mélységérzékelő kamera dolgozza fel. Rendkívül jó ötlet a fejlesztőktől, hogy ezt így oldották meg ugyanis egy sztereókamerás rendszer nem tudna működni sötétben vagy rossz fényviszonyoknál drasztikusan csökkenne a teljesítménye. Az egykamerás infravörös rendszer viszont minden fényviszonyban megállja a helyét és a felhasználó számára láthatatlanok a vetített infravörös pontok. Működése ezért sokban hasonlít a strukturált fényvetítéses szkennerre, mivel a pontok itt is egy meghatározott struktúra alapján vetítődnek és a torzulásuk alapján számítható ki a látótérbe helyezett test kiterjedése.

Játékvezérlés során hangokkal is irányíthatunk bizonyos funkciókat. A Kinect itt is előrébb lépett. Az eszközben helyet foglal két mikrofon is, amik együttműködve meg tudják határozni a hang irányát. A dupla mikrofonoknak köszönhetően a zajszűrésre is lehetőség van. Ha a hangok a szkeletonok irányából érkeznek az eszköz felé, akkor jogosan azt hozzájuk társíthatják az alkalmazások és el is különítheti azokat a felhasználók szerint. Így jöhet létre az is, hogy egyszerre, egy időben több felhasználó utasítását képes végrehajtani az erre felkészített alkalmazás.

Az eszköz így működése közben három adatfolyamot képes szolgáltatni. Az erőforrások kihasználása érdekében a nem használt szenzorokat. A készülékben egy gyorsulásmérő is található. Ez minden irányban képes mérést végezni. Segítségével követni tudjuk a kamera elmozdulását, így jobb feldolgozást biztosítva olyan megoldásokban ahol az eszköz egy mozgó egységre van illesztve, esetleg maga a felhasználó mozgatja. További segítséget nyújt a kamerába szerelt motor, ami a középállásából 27°-al felfelé és lefelé egyaránt képes elforgatni.

15. kép - Az emitter által kivetített pontok

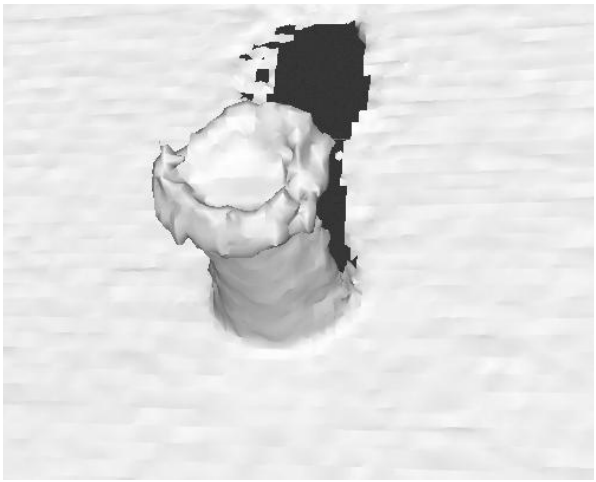
6.1 A Kinect felhasználási módjai

A Kinect egy rendkívül sokoldalú eszköz. A szkeleton detektálásnak köszönhetően a gesztusnyelvet használva irányíthatunk alkalmazásokat, vagy akár egy intelligens szobát is esetleg robotot vagy csak robot karokat. A szkeletonnal való irányításnak csak a képzeletünk szabhat határt. A szkeleton és az RGB kamera segítségével Blue box felvételeket készíthetünk egyszerűen, Blue box nélkül, és akár a ruházatunkat is kicserélhetjük virtuálisan.

Az 1.7-es Kinect SDK-val rengeteg új lehetőség nyílt a fejlesztők előtt, ugyanis ekkor került bele a Face tracking, a Kinect fusion és az OpenCV támogatás. Az emberi fejet előbb is képes volt felismerni a rendszer, ez logikusan következik a szkeleton meglétéből, de csak a fej helyzetét volt képes meghatározni, az arc gesztusait nem volt képes feldolgozni. A Face tracking ezt a feladatot hivatott ellátni. A Kinect fusion egy másik irányban folyt fejlesztés eredménye. Ennek segítségével az eszköz alkalmassá vált testek háromdimenziós rekonstrukciójára. Az OpenCV 2.4.x. támogatás megjelenése az SDK-ban egy új szintre emelte a képfeldolgozási lehetőségeket. Az elérhető funkciók száma megsokszorozódott és a témában jártas fejlesztők nagy többsége hosszú ideje ismerheti a függvénykönyvtárat.

6.2 Szkennelés a Kinect felhasználásával

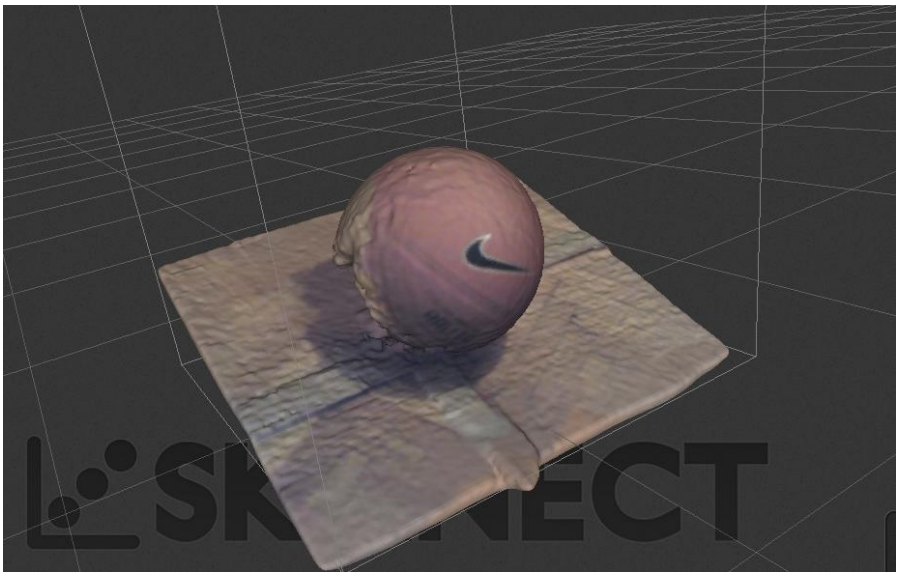
A szkennelés megvalósításához a Skanect nevű programot választottam. Ez az okos szoftver könnyedén együttműködik a Kinectel for Windows, ezen kívül támogatja a Kinect for Xbox360-t, az Asus Xtion-t, a Primesense Carmine-t és a 2014 júliusában forgalomba kerülő Structure Sensor-t ami a többi egymáshoz igen hasonlóval ellentétben egy mobilis iPad-re vagy laptopra szerelhető külön erőforrást nem igénylő igen kisméretű eszköznek ígérkezik. A szoftver ezen kívül támogatja a Cuda 2.0 kompatibilis grafikai kártyákat a gyorsabb feldolgozás érdekében



16. kép - Kinect-el szkennelt fog lenyomat

A rendszer minimális követelménye a Kinect SDK 1.8-as verziója. Ezek után jól bevilágított térben szkennelhetünk. Előre érdemes beállítani a szoftvert a szkennelendő objektum

méreteihez, így nagyon kis térben is használhatjuk úgy is, hogy a látótérben sok más objektum is található. Szkennelés közben a tárgyat forgassuk át saját tengelye körül, ne túlzottan gyors tempóban. A Kinect nagyobb tárgyak szkennelésére alkalmas leginkább. Részletes és kis objektumok csak nagyon jó fényviszonyok mellett készíthetők.



17. kép - Kinectel szkennelt kosárlabda

7 A programok működtetése

Felhasznált programkönyvtárak és fejlesztői eszközök:

- Qt 5.1.1 – Fejlesztői környezet. Az összes user interface-k használata miatt telepítése kötelező.
- OpenCV 2.4.5. – Programkönyvtár. A programok által használ könyvtárak (.dll-ek) az adott program könyvtárában találhatók.
- PCL 1.6. – Programkönyvtár. A programok által használ könyvtárak (.dll-ek) az adott program könyvtárában találhatók. A könyvtár tartalmaz harmadik fél által kiadott könyvtárakat is: Boost, Eigen, FLANN, OpenNI, Qhull, VTK.

Ha a programok a Qt 5.1.1 telepítése után sem indulnak, érdemes telepíteni a fennt említett könyvtárakat is.

8 Irodalomjegyzék

[1] Kató Zoltán, Czúni László: Számítógépes látás. Typotex Kiadó, 2011

[2] Jean-Yves Bouguet. Camera Calibration Toolbox for Matlab.

[3] Arturo de la Escalera and Jose María Armingol - Automatic Chessboard Detection for Intrinsic and Extrinsic Camera Parameter Calibration, Open Access Journal 2010

[4] Douglas Lanman and Gabriel Taubin - Build Your Own 3D Scanner:3D Photography for Beginners - SIGGRAPH 2009

[5] Yishi Guo - Multiple Camera Support for CCV - Google Summer of Code 2011

[6] http://www.inf.u-szeged.hu/~kato/teaching/

[7] http://www.pointclouds.org/

[8] http://docs.pointclouds.org/

[9] http://opencv.org/

[10] http://docs.opencv.org/opencv_tutorials.pdf

[11] http://mesh.brown.edu/byo3d/

[12] http://www.microsoft.com/en-us/kinectforwindows/