TDK DOLGOZAT

Iteratív ML-EM CT képrekonstrukció GPU-ra

Molnár Balázs

Témavezető: Dr. Légrády Dávid
egyetemi docens
Nukleáris Technika Intézet
Budapesti Műszaki és Gazdaságtudományi Egyetem

BME
2013

Témakiírás

A dolgozat egy olyan CT képrekonstrukciójára alkalmas módszert mutat be, aminek gyakorlatba átültetett eredményeit eddig a tudományos életben még nem mutatták be (a probléma egy más módon történő közelítését olvashatjuk [1]-ben), az elméleti alapok viszont sok pozitívummal kecsegtetnek. A vizsgált iterációs sémát a vonatkozó szakirodalomban [2] az ún. EM (Expectation Maximization) módszerből származtatják. Alapvető feltételezés, hogy a detektálható beütések száma Poisson-eloszlású. Ezzel élve elvégezhető annak a feltételes valószínűségnek a maximalizálása, hogy egy adott (a mintát jellemző) lineáris gyengítési együttható eloszlás mellett milyen beütésszámokat detektáltunk. Az iteráció lépéseiben mindig meg kell becsülnünk az egyes voxelekbe (háromdimenziós pixel) bejutó, és kimenő részecskék számát, valamint az adott voxelben befutott átlagos úthosszat. Magát a mérést is szimulálni szükséges, az aktuálisan iterált gyengítési együttható eloszlás mellett szimulált detektor-beütésszámokat hasonlítani kell a valódi mérési adatokhoz. A TDK dolgozat célja az iteráció tesztelése, több fantom rekonstruált képének létrehozása, és ezek bemutatása. A kutatás jelenlegi fázisában az algoritmus CPU-ról GPU-ra történő átültetése az egyik legfontosabb feladat.

Motiváció

A TDK dolgozat, és a témához kapcsolódó munkáim egy, az egyetem Nukleáris Technika Intézetén (NTI) belül Légrády Dávid által irányított csoportmunka része, aminek jelentősége abban rejlik, hogy többféle rekonstrukciós algoritmusok programozásán kívül magának egy CT-berendezésnek a kalibrációja, méréstechnikájának kidolgozása is folyamatban van, tehát a rekonstrukció tesztelését a saját mérési adatainkon is végrehajthatjuk majd, valamint a projekt méréstechnikai része is profitálhat a különböző képalkotási technikákból.

Önállósági nyilatkozat

Alulírott Molnár Balázs, a Budapesti Műszaki és Gazdaságtudományi Egyetem fizikus MSc szakos hallgatója kijelentem, hogy ezt a TDK dolgozatot meg nem engedett segédeszközök nélkül, önállóan, a témavezető irányításával készítettem, és csak a megadott forrásokat használtam fel. Minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból vettem, a forrás megadásával jelöltem.

.....

Molnár Balázs

Tartalomjegyzék

1. Bevezetés

1.1 Általános bevezetés

A komputertomográfiás (továbbiakban CT[1]) képalkotás a röntgen sugárzás anyaggal való kölcsönhatásán alapszik. A kölcsönhatás sztochasztikus jellegű, a fizikai folyamatokat valószínűségekkel, pontosabban hatáskeresztmetszetekkel írjuk le. Az anyagban végbemehet abszorpció, rugalmas, és rugalmatlan szórás. A párkeltés jelensége nem játszik szerepet a CT vizsgálatok esetében, hiszen az alkalmazott röntgen sugárzás maximális energiája általában 100-200 keV. A képszintézishez a fizikai folyamatokat modelleznünk kell, így szimulációk révén következtethetünk a mérési adatokból az anyagot meghatározó skálareloszlásra, például a lineáris gyengítési együttható térbeli eloszlására. Ez azt jelenti, hogy egy kísérlet során megmérjük, hogy egy ismert intenzitású röntgen-forrással besugározva a fantomot, mennyi áthaladó fotont detektálhatunk, majd ebből az információból kiindulva képet alkotunk a fantomról. Ez az információ azonban nem elégséges, hiszen csak a vetületeket ismerjük általa. A mérés közben fellépő jelenségek szimulációjának segítségével már lehetséges az eredeti eloszlás rekonstrukciója. Több módszer is ismert ennek megoldására, legpontosabban a Monte Carlo módszerrel lehet a részecske-transzportot figyelembe venni. Hátránya, hogy nagy mennyiségű indított részecskére van szükségünk ahhoz, hogy kis relatív hibákat érjünk el. A szórások, azon belül is a rugalmas Rayleigh-szórás, és a rugalmatlan Compton-szórás Monte-Carlo módszerekkel történő szimulációjáról áttekintést [3]-ben találunk. A szórásról a dolgozatban nem lesz szó, ugyanis a programfejlesztés jelen fázisában még a szórási jelenségek szimulációjától eltekintettem, csak az abszorpcióból származó információt használtam képalkotási folyamatban.

Az orvosi gyakorlatban a CT igen elterjedt diagnosztikai eljárás, így többféle algoritmus is létezik a kép rekonstrukciójára [1]. A módszerek alapvetően két csoportba sorolhatók: iteratív és analitikus. Az analitikus megoldások közül a legelterjedtebb a szűrt visszavetítés, az egyszerűsége és gyorsasága miatt. Az iteratív eljárások előnye az analitikusokkal szemben a kevesebb műtermék, illetve a valósághűbb modellezés lehetősége.

[1] Computed Tomography

1.2 Transzmissziós ML algoritmus általában

A dolgozatomban bemutatott rekonstrukció a Maximum Likelihoood (továbbiakban ML) alapötletét használja fel. Tekintsük egy transzmissziós kísérletnél Y-t a mért beütések számának valószínűségi változó vektoraként (továbbiakban vvv). Jelölje μ a rekonstruálandó skálareloszlást (lineáris gyengítési együttható), és vegyük figyelembe μ-t úgy, mint egy paramétert, ami mellett Y-t mérünk. Egy adott μ mellett mért Y vvv súlyfüggvénye legyen g(y|μ), aminek a maximumát keressük μ szerint, vagyis azt a skálareloszlást, ami mellet y a legnagyobb valószínűséggel felelne meg a valódi mérésünk eredményének. Ez általában nem könnyű, ugyanis az Y adatok nem tartalmaznak elegendő információt a függvény maximalizálásához, hanem egy teljes adathalmaz vetületét adják csak. Ezt a teljes adathalmazt, amiből a rekonstrukciót maradéktalanul végre lehetne hajtani, jelölje X vvv, és legyen ennek súlyfüggvénye f(x|μ). Ha a mérendő térfogatot felosztjuk voxelekre[2], akkor a legkézenfekvőbb választás, ha X tartalmazza az egyes voxelekbe belépő részecskék számát. Ekkor ugyanis egy projekció esetén a j. voxel μ_j abszorpciós értékét egyértelműen meg lehet határozni az X_j illetve az X_{j+1}, vagyis a belépő és kilépő fotonok számának, segítségével. Egy LoR[3] által érintett voxelek száma legyen m-1, így igaz lesz az $Y=X_m$ egyenlőség (az utolsó voxelt elhagyó részecskék száma X_m). Tekintsük most a

$$H\left(\mu \mid \mu_n\right) = E\left(\ln f(X \mid \mu) \mid Y, \mu_n\right) - \ln\left[g\left(Y \mid \mu\right)\right]$$

függvényt. Megmutatható a Jensen egyenlőtlenség alkalmazásával, hogy H maximuma μ szerint éppen μ_n-nél van, vagyis $\max_{\mu}\left\{H\left(\mu \mid \mu_n\right)\right\} = H\left(\mu_n \mid \mu_n\right)$. A bizonyítás részleteit mellőzve belátható (felírva a $H\left(\mu_n \mid \mu_n\right) - H\left(\mu \mid \mu_n\right)$ mennyiséget, és a logaritmus függvényre alkalmazva a Jensen egyenlőtlenséget), hogy ha úgy választjuk meg μ_n-t, hogy $f\left(x \mid \mu_n\right) > f\left(x \mid \mu\right)$, akkor H-nak valóban $\mu=\mu_n$-nél van maximuma. Állítsunk elő tehát egy $\left\{\mu_n\right\}$ sorozatot, és minden lépésben maximalizáljuk $E\left(\ln f(X \mid \mu_{n+1}) \mid Y, \mu_n\right)$ értékét μ_{n+1} szerint. Ekkor H-t tekintve kimozdultunk $H\left(\mu_n \mid \mu_n\right)$-ből $H\left(\mu_{n+1} \mid \mu_n\right)$-be, így az előzőek alapján H kisebb lett. Ez maga után vonja azt, hogy $\ln\left[g\left(Y \mid \mu_{n+1}\right)\right] > \ln\left[g\left(Y \mid \mu_n\right)\right]$, vagyis megvalósítottuk a Maximum Likelihood módszert. Az EM módszer összességében

[2] Háromdimenziós pixel
[3] Line of Response - válaszegyenes

visszavezeti a likelihood becslést várható érték maximalizálásra, ahol a teljes adathalmaz Y melletti várható értékét kell optimalizálni a gyengítési együttható eloszlás szerint.

Egyetlen dolog hiányzik még a fenti módszer konvergenciáját tekintve: még nem mondtuk meg, hogy a log-likelihood függvénynek milyen szélsőértéke van. Természetesen azt szeretnénk, ha maximuma lenne, vagyis $\ln\left[g\left(Y\mid\mu\right)\right]$ μ-ben konkáv lenne. Ez pontosan akkor teljesül, ha második deriváltmátrixa negatív definit. A következőkben beláthatjuk, mikor is teljesül ez a szimulációnkban.

Jelölje most Y_i az i. projekciónál a detektort elérő fotonok számának vv-ját. Modellezzük Y_i-t Poisson eloszlású vv-ként[4], vagyis eloszlásfüggvénye legyen

$$g(y_i \mid \mu) = \frac{\lambda_i^{y_i}\left(\mu\right)\cdot e^{-\lambda_i(\mu)}}{y_i!}$$

egy adott μ paraméter mellett. Az Y_i valószínűségi változó λ_i várható értéke felírható a Beer-Lambert törvény alapján:

$$\lambda_i\left(\mu\right) = b_i \exp\left(-\sum_{j\in L_i} l_{ij}\mu_j\right),$$

ahol b_i az i. projekciónál a forrást elhagyó részecskék száma, l_{ij} pedig az i. projekcióban a j. voxelben a foton által befutott út hossza. Az összes projekcióra felírható Y együttes eloszlásfüggvény a $g(y_i|\mu)$ függvények szorzata lesz, hiszen az Y_i-k egymástól független vv-k. Ebből felírhatjuk a log-likelihood függvényt a következő alakban:

$$\ln\left[g(y\mid\mu)\right] = \ln\left[\prod_i g(y_i\mid\mu)\right] = \sum_i \ln\left[g(y_i\mid\mu)\right] =$$

$$\sum_i\left\{-b_i\exp\left(-\sum_{j\in L_i} l_{ij}\mu_j\right) - y_i\sum_{j\in L_i} l_{ij}\mu_j + y_i\ln b_i - \ln y_i!\right\}$$

A konvexitás vizsgálatához venni kell a fenti függvény kétszeres parciális deriváltjait:

$$H_{kl} = \frac{\partial^2}{\partial\mu_k\partial\mu_l}\ln\left[g\left(y,\mu\right)\right] = -\sum_i a_{ik}b_i\exp\left(-\sum_{j\in L_i} l_{ij}\mu_j\right)a_{il}$$

ahol a_{ik} együtthatók az A mátrix elemei. A egy olyan mátrix, aminek sorai az egyes projekciókat indexelik, oszlopai pedig a voxeleket. A a_{ik} eleme azon úthossz, amit az i. projekcióban a k. voxelben befut a részecske, vagyis

$$a_{ik} = \begin{cases} l_{ik} \leftrightarrow k \in L_i \\ 0 \leftrightarrow k \notin L_i \end{cases}.$$

[4] Arról, hogy ez miért megfelelő feltételezés az esetünkben, és milyen feltételekkel, még lesz szó a későbbiekben

Ahhoz, hogy ln(g) konkáv legyen, H-nak negatív definitnek kell lennie, vagyis minden v vektorra: $vHv < 0$ kell, hogy teljesüljön. Ez pontosan akkor áll fenn, ha az A mátrix teljes rangú $n \times m$-es mátrix, ahol $n \geq m$. Tehát legalább annyi projekciót kell végeznünk, ahány voxelből áll a rekonstruálandó skálareloszlás által elfoglalt térfogat.

A következőkben tegyük fel, hogy a konkávság feltétele teljesül (vagyis az ML-EM konvergens), és nézzük meg, hogyan lehet az iterációt előállítani.

1.3 A megvalósított iterációs séma előállítása

Az előzőekben már felírtuk a log-likelihood függvényt $\ln\left[g(y \mid \mu)\right] = \sum_i \left\{ -b_i \exp\left(-\sum_{j \in L_i} l_{ij}\mu_j \right) - y_i \sum_{j \in L_i} l_{ij}\mu_j + y_i \ln b_i - \ln y_i ! \right\}$ alakban. Ugyanígy a teljes adathalmazra (X) is megtehetjük a log-likelihood függvény felírását. Észre kell vennünk ehhez, hogy a j+1. voxelbe jutó részecskék binomiális eloszlást követnek, vagyis $X_{j+1} \sim Binom(n, p)$, ahol a próbák száma n=X_j, valamint a siker valószínűsége $p = \exp\left(-\mu_j l_j\right)$. Tételezzük fel azt is, hogy az X_j, csakúgy, mint a detektort ért beütések száma, Poisson eloszlást követ λ_j paraméterrel ($X(j) \sim Poi(\lambda_j)$). Az egyszerűség kedvéért bevezettük a $\lambda_r = b \exp\left(-\sum_{k=1}^{r-1} \mu_k l_k \right)$ jelölést. Így

$$\ln\left[f(x \mid \mu)\right] =$$

$$\sum_i -b_i + X_1 \ln b_i - \ln X_1 ! + \sum_{j=1}^{m-1} \ln \binom{X_j}{X_{j+1}} + X_{j+1} \ln\left(\exp\left(-l_{ij}\mu_j\right)\right) + \left(X_j - X_{j+1}\right) \ln\left(1 - \exp\left(-l_{ij}\mu_j\right)\right)$$

Láthatjuk, hogy ha ennek a mennyiségnek a várható értékét akarjuk maximalizálni X_m feltétel mellett, akkor még ki kell találnunk az $E(X_j|X_m)$ várható értéket.

Most fontos megemlíteni, hogy mikor is jó az a feltételezés, hogy mind X_j, mind $Y=X_m$ (a detektálható, vagyis az utolsó voxelt elhagyó részecskék száma) Poisson eloszlást követ. Mint láttuk a voxelt elhagyó részecskék a bemenő részecskék számától függő binomiálisak, az exponenciális gyengülés szerinti valószínűséggel. Ez több voxelen való áthaladásra is igaz, a valószínűség változik meg: $p = \exp\left(-\sum_k \mu_k l_k \right)$ A binomiális eloszlás Poisson-

approximációját alkalmazhatjuk erre az esetre, ha a binomiális eloszlás n és p paraméterére: $n \to \infty$ ugyanakkor $p \to 0$, de úgy, hogy $np \to \lambda < \infty$. Ezt úgy teljesítjük, (visszanézve a fenti n és p paraméterekre) hogy sok fotont indítunk a szimulációban, valamint kellően abszorbens és/vagy nagyméretű közeget használunk (ezt nem túl nehéz teljesíteni).

Ezek után kanyarodjunk vissza az eredeti célunk felé, ami szerint az $E(X_j \mid X_m)$ feltételes várható értékről szeretnénk információt kapni. X_m binomiális eloszlású, feltéve X_j-t, a korábbi jelölésekkel $X_m \mid X_j \sim Binom\left(X_j, \frac{\lambda_m}{\lambda_j}\right)$. A Bayes-tétel segítségével a fordított feltételt is kiszámíthatjuk, vagyis a kérdéses $P(X_j \mid X_m) = \frac{P(X_m \mid X_j) \cdot P(X_m)}{P(X_j)}$ kiszámítható, hiszen mindegyikről tudjuk már, hogy milyen eloszlású. A részletes levezetés [2]-ben olvasható. Arra a következtetésre jutunk, hogy $X_j - X_m \mid X_m \sim Poi(\lambda_j - \lambda_m)$, amiből egyből következik, hogy $E(X_j - X_m \mid X_m) = \lambda_j - \lambda_m = E(X_j) - E(X_m)$, s ekképp $E(X_j \mid X_m) = X_m + E(X_j) - E(X_m)$, ami pontosan a maximalizálandó várható érték számolásához kell. Most már elkezdhetjük a tényleges maximalizációt, vagyis deriváljuk le az $E(\ln f(X \mid \mu) \mid X_m, \mu)$-t μ szerint, tegyük egyenlővé nullával. ln(f) adott a fenti formulában, a feltételes várható értékeket pedig éppen az előbb írtuk fel. Kiírva:

$$\sum_i \sum_{j \in L_i} \left\{ E(X_{j+1} \mid X_m) \ln\left(\exp(-l_{ij}\mu_j)\right) + \left(E(X_j \mid X_m) - E(X_{j+1} \mid X_m)\right) \ln\left(1 - \exp(-l_{ij}\mu_j)\right) \right\} + R,$$

ahol R tartalmazza az összes olyan paramétert, ami nem függ μ-től. A deriválás után:

$$0 = -\sum_i E(X_{j+1} \mid X_m) l_{ij} + \sum_i \left(E(X_j \mid X_m) - E(X_{j+1} \mid X_m)\right) \frac{l_{ij} \exp(-l_{ij}\mu_j)}{1 - \exp(-l_{ij}\mu_j)}$$

$$= \sum_i -E(X_{j+1} \mid X_m) l_{ij} + \sum_i \left(E(X_j \mid X_m) - E(X_{j+1} \mid X_m)\right) \frac{l_{ij}}{\exp(l_{ij}\mu_j) - 1}$$

Ez a transzcendens egyenlet numerikusan oldható meg, ha $s = l_{ij}\mu_j$ elég kicsi:

$$\frac{1}{e^s - 1} = \frac{1}{s} - \frac{1}{2} + \frac{s}{12} + O(s^3)$$

Az $\frac{1}{e^s - 1} \approx \frac{1}{s} - \frac{1}{2}$ közelítést alkalmazva az iterációs képlet az

$$\mu_i^{n+1} = \frac{\sum_i \left(E\left(X_j \mid X_m\right) - E\left(X_{j+1} \mid X_m\right) \right)}{\frac{1}{2} \sum_i \left(E\left(X_j \mid X_m\right) + E\left(X_{j+1} \mid X_m\right) \right) l_{ij}}$$

alakot ölti.

Behelyettesítve a feltételes várható értékeket, és kicsit szemléletesebb jelölést alkalmazva:

$$\mu[i] = \frac{\sum_{lor \in L_i} \left(X_{lor}[j] - X_{lor}[j+1] \right)}{\sum_{lor \in L_i} \left(Y_{lor} - \hat{Y}_{lor} + \frac{X_{lor}[j] + X_{lor}[j+1]}{2} \right) \cdot L_{lor}[j]} .$$

Ez az iterációs séma azt jelenti, hogy az iterációban soron következő μ[i]-t, vagyis az i. voxel gyengítési együtthatóját úgy számoljuk, hogy minden, az i. voxelen áthaladó, LoR-ra összegezzük a bejövő és kimenő részecskék különbségére adott becslésünket (még az előző μ mellett), majd elosztjuk egy hasonló összeggel (nevezőben), amiben szerepelnek a mérési adatok (Y), a szimulált mérési eredmények az aktuális μ mellett ($\hat{Y}$), valamint az előbbi X-ek, és az adott voxelben befutott átlagos úthossz (L).

2. A dolgozat célja, felépítése

2.1 A szakdolgozat és a TDK dolgozat

A Bsc szakdolgozatomban [4] egy hasonló, iteratív képrekonstrukciós eljárást teszteltem, ami szintén az ML becslésből indult ki, azonban egy másik közelítő módszerrel, a teljes adathalmaz bevezetése nélkül, csupán a mérési eredmények, és a befutott úthosszak becsléséből számította ki az iteráció következő elemét. Az EM algoritmus, és a segítségével előállított formula sokkal pozitívabb kilátásokkal kecsegtet, főleg a konvergencia, és a futási idő szempontjából. Sajnos a kettőt nem lehet precízen összehasonlítani, főleg a fejlesztés jelen fázisában, ahol még nem dolgozhatunk megfelelően nagy felbontásokkal. A TDK munkámban tesztelt algoritmus még csak CPU-n fut, a GPU-ra való implementáció még folyamatban van. Képeket már lehet így is rekonstruálni, persze jóval több idő alatt, mint a szakdolgozatban szereplő módszerrel. A szakdolgozatomból a bevezetőnek az első szakaszát

vettem át, valamint a fantombeolvasás, és az L2 norma definiálásáról szóló részeket. Az elméleti alapok is főképpen újak, és az implementációban sincsen átfedés, csak a tesztelésre alkalmazott módszerek hasonlóak. A TDK munkám célja, hogy bemutassam, az elméleti alapok hogyan alkalmazhatók, bizonyítsam, hogy értelmes a módszer, vagyis konvergál a várt megoldáshoz, és hogy miként sikerült implementálni azt, mint számítógépes szimulációt.

2.2 Az implementáció alapötletei

A bevezetőben ismertetett iterációs sémát többféleképpen is implementálhatjuk. Ahhoz, hogy megismerjük a megvalósítható, és előreláthatóan „jól működő" opciókat, meg kell vizsgálnunk, hogy mik azok a tényezők, amik alapvetően korlátozzák a lehetőségeket. Nyilvánvalóan, az implementációt egy számítógépes szimuláció jelenti, vagyis egy választott programnyelven íródott kódról lehet szó. Az ismertetett statisztikus iteratív rekonstrukció általános körülmények között igen lassú tud lenni. Egy „jól működő" rekonstrukciótól elvárjuk, hogy versenyképes legyen a másfajta (analitikus) módszerekkel szemben, mind futási időben, mind a képminőséget illetően. Ami a programnyelv kiválasztását illeti, legjobb, ha egy olyan nyelvet választunk, ami alacsony szintű, vagyis a gépi kódtól minél kevésbé tér el. A C/C++ nyelvekkel például nemritkán nagyságrenddel jobb futási időt érhetünk el, mint magasabb szintű nyelvek (Java, C#) esetében. További előnye még a kézi memóriakezelés lehetősége, ami, mint a későbbiekben látjuk, nagy könnyítést jelenthet a programozásban. A modern számítástechnika több lehetőséget is kínál az algoritmus felgyorsítására, megoldhatjuk az adott problémát például számítógép-klaszterekkel, vagy a grafikus kártya (GPU[5]) használatával. Ezen módszerek előnyei lényegében a párhuzamosíthatóságból fakadnak, ami azt jelenti, hogy a végrehajtandó feladatot részegységekre bontjuk, amik számítását egy időben meg tudjuk tenni, és így érjük el a futási idő töredékét. A GPU-n való implementációra már vannak bevált módszerek, hiszen a szakdolgozatom alapjául szolgáló algoritmus is már az NTI szerver grafikus kártyáján futott.

[5] Graphics Processing Unit – grafikus feldolgozó egység

2.3 A program felépítése

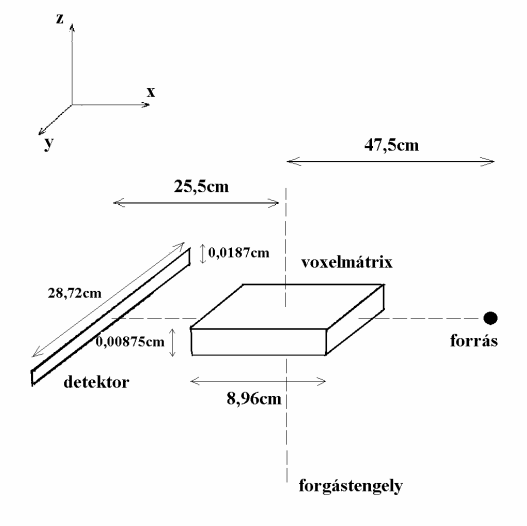
A TDK munkám témáját adó algoritmus felépítését fogom ismertetni általánosságban, főbb elemeit megemlítve. Elsőként megtörténik az eredeti (később rekonstruálandó) kétdimenziós kép bevitele. Az eredeti képen az algoritmus mérési szimulációt végez monokróm nyalábbal, vagyis végrehajt egy (vagy több) ún. előrevetítést aminek eredménye az előző fejezetben Y-nal jelölt beütésszám. A program a fejlesztés jelen fázisában nem szimulál szórást, a fotonnal csak abszorpció történhet. Az előrevetítést Monte-Carlo részecsketranszport módszerrel valósítja meg, a szabad úthosszt a Woodcock-módszerrel mintavételezve. A „mérés" után következik maga a rekonstrukció, melynek lépései:

1. Inicializálás: a fantomot első közelítésben egyenletes eloszlásúnak feltételezzük.
2. A létrehozott gyengítési együttható eloszlás egy ciklusba kerül, ahol is megvalósul a már ismertetett iterációs séma:

$$\mu[i] = \frac{\sum_{lor \in L_i} \left(X_{lor}[j] - X_{lor}[j+1] \right)}{\sum_{lor \in L_i} \left(Y_{lor} - \hat{Y}_{lor} + \frac{X_{lor}[j] + X_{lor}[j+1]}{2} \right) \cdot L_{lor}[j]}$$

3. Egy adott lépésben végrehajtódik ismét az előrevetítés az aktuális eloszláson, aminek eredménye az $\hat{Y}$ beütésszám, a visszavetítés pedig becslést ad X-re, és az L-re

A mérési elrendezés egy cone beam kisállat CT berendezés geometriája volt, erről részletesebb leírást találhatunk [2]-ben. Az ábrán látható méreteket a szakdolgozatomban alkalmaztam, a forrás-voxelrendszer távolság 1cm, a detektor- voxelrendszer távolság 0,5cm volt, valamint kétdimenziós fantomokat teszteltem 10×10-es, 30x30-as és 64x64-es méretekben, a voxelrendszer oldalai x és y irányban 0,6cm voltak és erre merőleges z irányban egy voxelnek a mérete. Ekképpen a voxelmátrix homogén sűrűségű kockákból állt, és ezekből egy z irányban egy voxelnyi vastag téglatestté épült fel. A detektor 50-150 db egységnyi hosszú (0,1cm) pixelből állt. Szemléltetést az elrendezésről az 1. ábra ad (az ábrán a szakdolgozatomban használt méretek vannak megadva).



1. ábra
A tesztelés geometriája

2. 3. Konkrét implementációk

3.1 Kiindulási alap

A munkám során három féle implementációját valósítottam meg az ML-EM algoritmusnak, ezeket először általánosságban ismertetem, majd rámutatok a közöttük lévő különbségekre.

A képrekonstrukciós feladatot most fogalmazzuk meg a számítógépes szimuláció szempontjából. A $\mu[i] = \frac{\sum_{lor \in L_i} \left(X_{lor}[j] - X_{lor}[j+1] \right)}{\sum_{lor \in L_i} \left(Y_{lor} - \hat{Y}_{lor} + \frac{X_{lor}[j] + X_{lor}[j+1]}{2} \right) \cdot L_{lor}[j]}$ közelítő képletben található mennyiségekre kell becslést adnunk, ezeknek számolására kell, hogy alkalmas legyen az algoritmus. Elsőként tekintsünk az Y és $\hat{Y}$ értékeit. Ezen változók tartalmazzák az adott LoR irányában a detektorra beérkező részecskék számát a mérésben, illetve az aktuálisan iterált lineáris gyengítési együttható eloszlás mellett szimulált beütéseket. Ezek becslését az algoritmus ún. előrevetítés részében végezzük el, hiszen ekkor magát a mérést szimuláljuk az $\hat{Y}$ meghatározásához (esetünkben az Y, vagyis magának a mérési

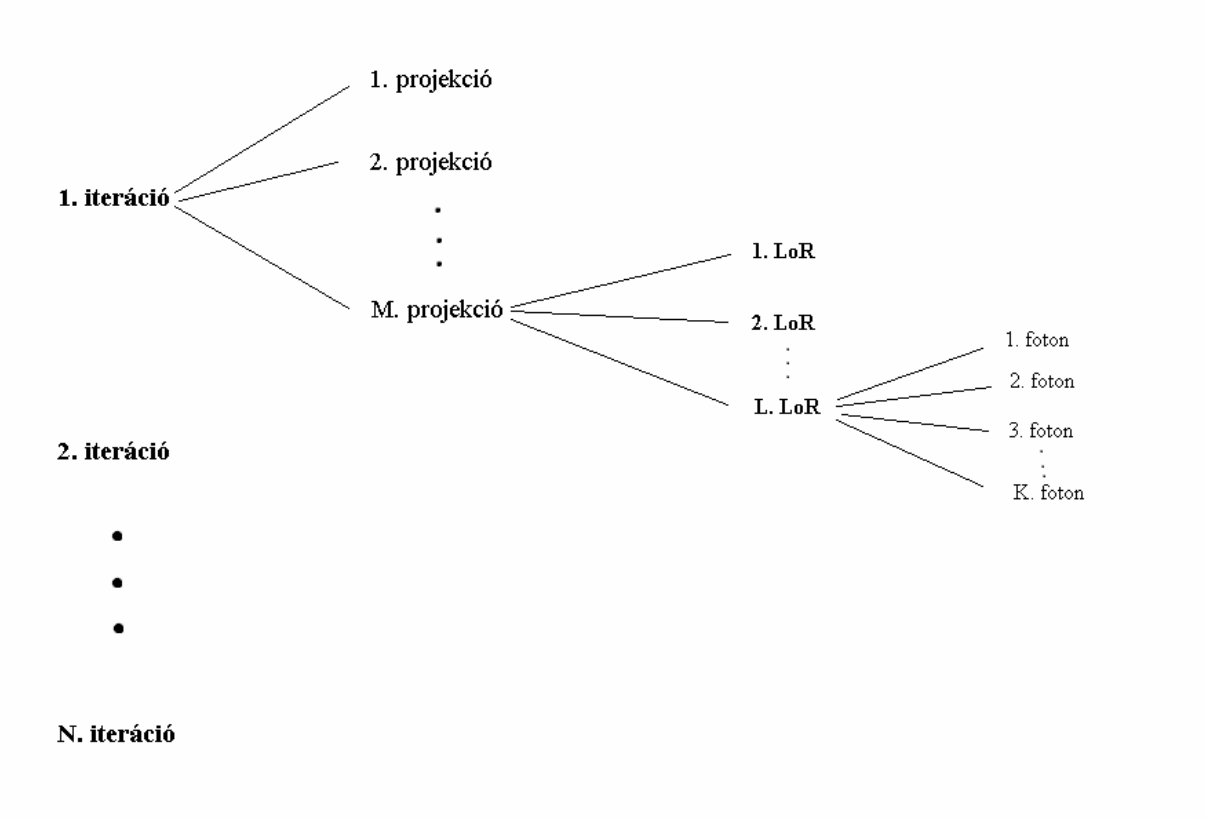
eredménynek a szimulációja is így történik). Az előrevetítés elvégzéséhez Monte-Carlo eljárást alkalmaztam, a szabad úthossz mintavételezését Woodcock-módszerrel végezve el. A programkód ezen része ekvivalens a szakdolgozatomban[4] bemutatott algoritmussal.

Miután a mérési szimuláció eredményei ($\hat{Y}$) a kezünkben vannak, megpróbálunk következtetni a kérdéses μ eloszlásra, vagyis visszavetítést végzünk. A fenti formula szerint a visszavetítésben kézenfekvő becsülni az L értékeit, vagyis az egyes voxelekben befutott úthosszakat. Egyszerű, gyors, és hatékony becslést kapunk a sugárkövetés módszerét alkalmazva, vagyis ha az L-et a voxelmátrixon való végiglépegetéssel becsüljük. Pontosabban választunk egy lépésközt, valamint megadjuk a LoR irányát, majd a LoR és a voxelmátrix két döféspontja (bemenet és kimenet) között ezzel a lépésközzel végigléptetjük a képzeletbeli részecskénket. Eközben minden voxelre regisztráljuk a bennük elkövetett lépések számát, ebből számoljuk az úthosszakat.
Nem triviális az X mennyiségek becslése, hiszen több opció is kínálkozik kiszámításukra. Az X[j] és az X[j+1] mennyiségek reprezentálják a LoR által érintett j.voxelbe és a soron következő voxelbe bemenő fotonok számát. Tárgyalhatjuk becslésüket együtt a mérési szimulációval, hiszen a fotonok iránysorsolása és voxelrendszeren történő áthaladása egyébként is itt kerül szimulációra, így az előrevetítésbe könnyen beleiktatható az X-ek becslése is. Más szempontból viszont egy értelmesebb eljárást kapunk, ha mégsem az előrevetítésben számoljuk ki ezeket a mennyiségeket, hanem a visszavetítésben, az L-ekkel együtt. Ennek a szemléletnek alapja, hogy nem veszítünk információt a fizikai jelenségekről azzal, hogy az X-eket nem Monte-Carlo módszerrel becsüljük, hanem analitikusan számoljuk ki. Összességében mindkét módszert megvizsgáltam a TDK munkám során, implementáltam egy olyan algoritmust, ami az előrevetítésben becsli az X mennyiséget, és további kettőt, amik pedig a visszavetítésben számolják értékét. Ezeket az eljárásokat fogom a továbbiakban részletesen ismertetni. Az egyes verziókat v1, v2, v3-mal fogom jelölni, és így fogok hivatkozni rájuk.

3.2 A v1 algoritmus

A v1 algoritmusában az iterációs ciklusba (minden ciklus eredménye egy újabb közelítés a μ eloszlásra) beágyazva található egy ciklus a projekciókra (különböző szögekből felvett

vetületekre) valamint egy másik, az adott forrás-detektor elfordulási szög mellett szimulálandó LoR-ok számára. Tehát egy projekción belül több LoR mentén gyűjtjük össze az információt, egy LoR kijelölésével pedig végrehajtjuk az előrevetítést és a visszavetítést is. Az előrevetítésben az X értékek, azaz az egyes voxelekbe belépő részecskék számlálása egyszerű elven működik: egy adott foton esetén, amíg el nem érjük az abszorpció helyét, addig az útba eső voxelekhez rendelt X értéket eggyel növelni kell, abszorpció után pedig nem. Az útba eső voxeleket nem a hagyományos sugárkövetéssel határoztam meg, hanem úgy, hogy kiszámoltam a szóban forgó LoR és a voxeleket határoló síkok metszéspontját. Ezt egy egyenes gömbi koordinátarendszerben felírt egyenletéből (a LoR azimut és polárszöge, valamint a forrás pozíció paraméterezi), és a voxelrendszert meghatározó xy, xz , és yz irányú síkok egyenletéből számoltam egyszerű behelyettesítésekkel. Ha ezek a metszéspontok megvannak, pontosan tudni lehet, hogy melyik voxelből melyikbe ment át a foton, így X becsülhető. Az L úthosszak becslésénél általában a voxelméret egytized részével való léptetést végeztem el. Ez elegendő felosztás, ezt könnyen beláthatjuk, ha újra megtekintjük az iterációs sémát (X és Y statisztikus ingadozása sokkal nagyobb hibát okoz, mint az, hogy L úthossz a voxelméret tizedével hosszabb vagy rövidebb-e).



2. ábra Az előrevetítés struktúrája v1-ben

Előnyök

- Kevés részecske indításával is jó minőségű kép
- Szimuláció tesztelésére alkalmas
- Könnyen áttekinthető, jól programozható

Hátrányok

- Lassú (sok egyenes-sík metszéspontot számol ki)
- Nehezen egyeztethető össze valódi méréssel (struktúra miatt)
- Egy LoR-hoz több foton kell

Összesítés: A szimuláció tesztelésére, és a programkód debuggolására jó, azonban nehezen optimalizálható, módosítható, és a valódi mérési eredményekkel nehezen egyeztethető, várhatóan igen lassú, viszont gyorsan konvergál (ez ellentmondásosnak tűnhet, de itt arról van szó, hogy egyetlen iteráció elvégzése sok időt vesz igénybe, viszont kis rendszerméreteknél hatékonyan konvergál)

3.3 A v2 algoritmus

A v2 implementáció alapötlete, hasonló az előzőhöz, vagyis ha már alkalmazunk egy sugárkövetési algoritmust a visszavetítésben, használjuk ezt arra is, hogy egyúttal megbecsüljük X fotonszámokat is. A különbség tehát az, hogy most nem az Y beütésszámokkal együtt adunk becslést a voxeleket érő részecskeszámra, hanem a visszavetítésben, az úthosszak számolásánál. Ez azért jó gondolat, mert a sugárkövetés során regisztráljuk az aktuális pozíciónkat, vagyis mindig tudjuk, hogy melyik voxelt éri az adott LoR. Nyilvánvaló, hogy ennek megvalósításához az előre-és visszavetítést nem végezhetjük el egymástól függetlenül, hiszen a cél az, hogy pontosan azon a LoR-on kövessük végig a foton útját, amin az előrevetítésben ment. A sugárkövetés során miközben léptetjük a pozíciót és számláljuk az egyes voxelekben eltöltött lépések számát, meg is növeljük a voxelhez tartozó X értékét eggyel, ha még az abszorpció pozícióját nem értük el. Az eljárás további részei ekvivalensek v1-gyel.

Előnyök

- Relatíve gyors (gyorsabb, mint v1)
- Könnyen konvergál
- Kevés indított foton is elég a jó minőségű képhez

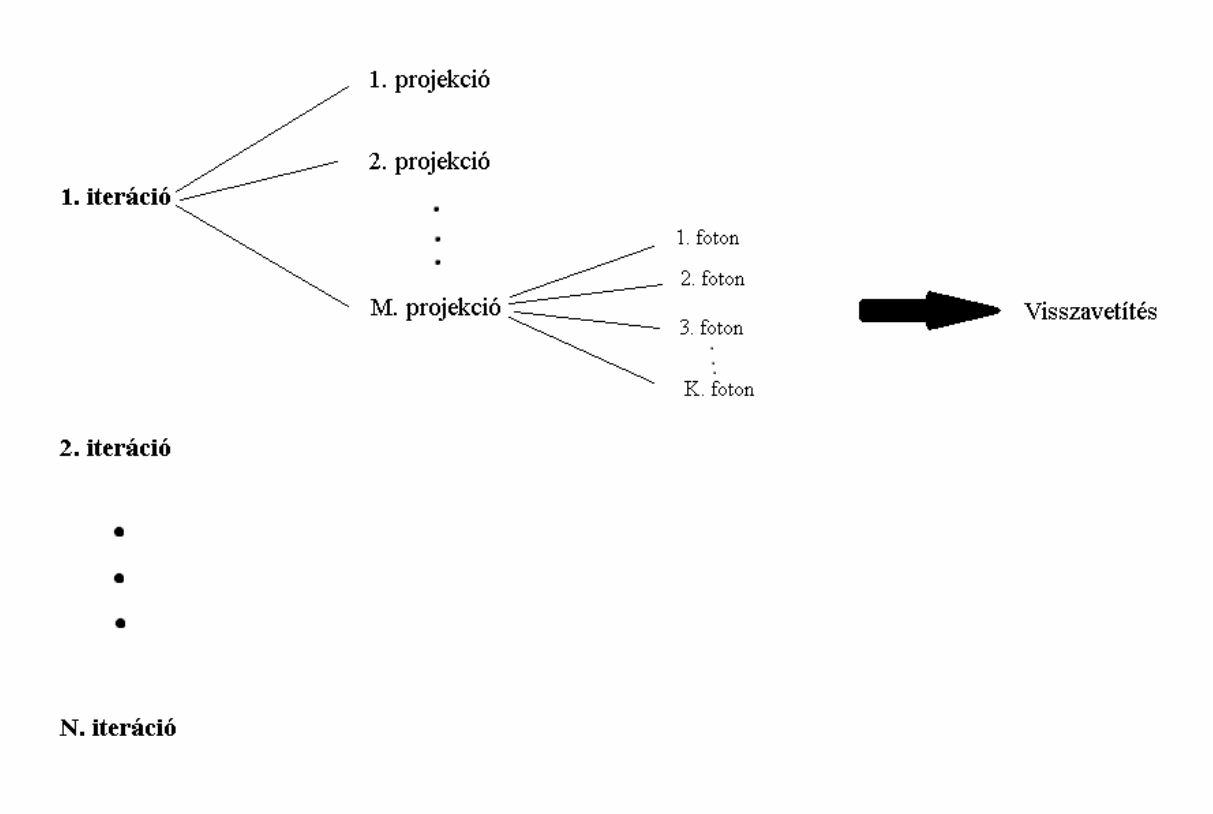
Hátrányok

- Nehezen fejleszthető a továbbiakban, nem végezhető függetlenül az előrevetítés és a visszavetítés
- Egy LoR-hoz több foton kell

Összességében gyors konvergenciát mutat, de csak ha a valódi mérést is szimuláljuk, nyilvánvaló, hogy egy reális mérésben nem csak meghatározott utakon haladnak végig a részecskék. Tesztet könnyen lehet v2-vel végrehajtani, így ezt használom a továbbiakban.

3.4 A v3 algoritmus

A harmadik módszer kiküszöbölni igyekszik a másik kettő hátrányát, ugyanis ebben az algoritmusban teljesen szétválasztottam az előre-és visszavetítést, és törekedtem a gyors és egyszerű megoldásokra, anélkül, hogy a fizikai tartalomból bármit is vesztett volna az eljárás. Mint már említettem, az egye voxelekbe jutó fotonok száma analitikusan is meghatározható, a Beer-Lambert törvény segítségével. Az L úthossz-számításokra eddig is alkalmazott sugárkövetést a v2-vel ellentétben függetlenítettem az előrevetítéstől (abszorpció helyétől), és az X-eket az exponenciális gyengítéssel számoltam, de természetesen ugyanúgy az L becslésekkel párhuzamosan. Tehát a v3 verzióban a sugárkövetés kettős feladatot lát el (L és X) úgy, hogy az előrevetítésben kijelölt LoR-ok, Monte-Carlo lépések irrelevánsak, ezek ismerete nélkül is visszavetíthető a mérési eredmény. A visszavetítésben önkényesen kijelölt LoR-ok mentén végezzük el az úthosszak (L) és részecskeszámok (X) becslését. Ezek az egyenesek a forrást az egyes detektorpixelek közepével összekötő szakaszok által kijelölt görbék.



3. ábra A v3 előrevetítések szerkezete

A visszavetítés működésének szemléltetéséhez nézzük meg, hogyan számol a program például a középső detektorpixelt a forrással összekötő LoR mentén. Először meghatározzuk azt, hogy hol találkozik először a foton a voxelrendszerrel, vagyis hol van a belépési pont. A szimulációban a képzeletbeli minta egy téglatestben van elhelyezve, ami voxelekre van osztva. Ennek a téglatestnek a lapjaira számítja ki az algoritmus az egyenessel való metszéspontokat, majd a forráshoz legközelebbi döféspontot adja meg, mint a belépés helye. Ezután egy, a felhasználó által választott, lépésközzel elkezdi tolni a LoR irányába a részecske pozícióját. Eközben ellenőrzi, hogy még ugyanabban a voxelben vagyunk-e, mint az előző lépésben, és ha nem, akkor L értékének a lépések számának és a lépésköznek a szorzatát adja, X pedig a következőképp adódik: $X[j+1] = X[j]\exp\{-\mu_{iter}[i]\cdot L[j]\}$. Itt i és j ugyanazon voxelt indexeli, de más sorrendben indexelt vektorok elemeire utal.[6]

Előnyök:

- Realisztikus modellezés
- Könnyen fejleszthető
- Rekonstrukció szempontjából több választható paraméterrel rendelkezünk

[6] Például L[j] abban a voxelben befutott úthosszat jelenti, amit a LoR j.-ként érint, μ[i] pedig ugyanennek a voxelnek a lineáris gyengítési együtthatója, de nem a LoR mentén fut az i index, hanem a voxelmátrix elemein.

Hátrányok:

- Sok részecske szimulációja szükséges
- Lassan konvergál

4. Tesztelés

Az algoritmus tesztelését a v2 verzióval hajtottam végre. Ennek egyszerű oka volt: ez volt a leggyorsabban konvergáló, és a legszebb képet adó rekonstrukció. Meg kell említenem, hogy természetesen a legjobb a v3 módszer lenne, azonban ezzel még a TDK leadási határidején belül nem tudtam képet alkotni. Kis méreteknél a v1 és v2 verziók alkalmasak az iteráció kvalitatív jellemzésére. Azt tűztem ki célul, hogy ezek segítségével megállapítom, hogy az elméleti háttér alkalmazható-e a gyakorlatban (hatékony-e az általam implementált ML-EM módszer), és mennyiben jobb a szakdolgozatomban bemutatott közelítéstől. A CPU-n való futás lassúsága miatt teszteseteket csak kétdimenziós néhányszor tíz voxelből álló rendszerre készítettem el.

4.1 Fantom beolvasás

A gyors és hatékony teszt-végrehajtás érdekében hasonlóan külső inputként érdemes a fantomokat is megadni. Ezen célból is írtam egy beolvasó kódot, ami egy mátrixból olvas be számokat, és megfelelteti ezeknek a voxelek gyengítési együtthatóját. A beolvasott érték az általam generált fantomok esetében egy 256 árnyalatot tartalmazó szürkesála volt, amit megfelelő értékekre normálva dolgozott fel a beolvasó algoritmus. A rekonstruált kép a program kimeneti fájljából készíthető. A szakdolgozatomban az Origin szoftver segítségével készült ábrákat láthatunk, a kimenő fájlt mátrixszá alakítottam, majd szürkesálán ábrázoltam. A képeken a nagyobb sűrűségű pixelek a világosabbak, míg a kisebb sűrűségűek sötétek.

4.2 Az L2 norma

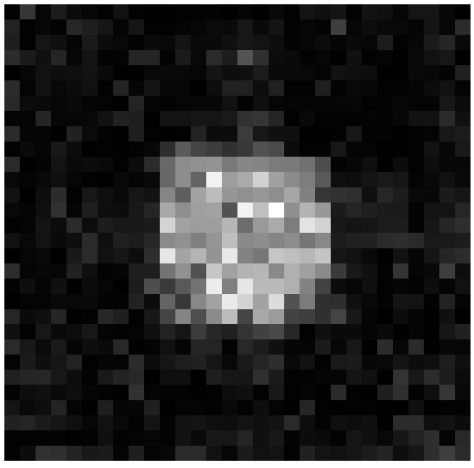
Alapvetően az a célunk, hogy eldöntsük, mennyire hasonlít a rekonstruált kép az eredeti fantom gyengítési együttható eloszlására. Ennek vizsgálatára több módszer is létezik, az egyik legegyszerűbb ilyen az L2 norma kiszámítása. Az L2 normát a két kép, vagy képrészlet, között az alábbi formában definiáljuk:

$$L2 = \frac{\sum_{i=1}^{N}\left(M_i^{eredeti} - M_i^{rekon}\right)^2}{\sum_{i=1}^{N}\left(M_i^{eredeti}\right)^2},$$

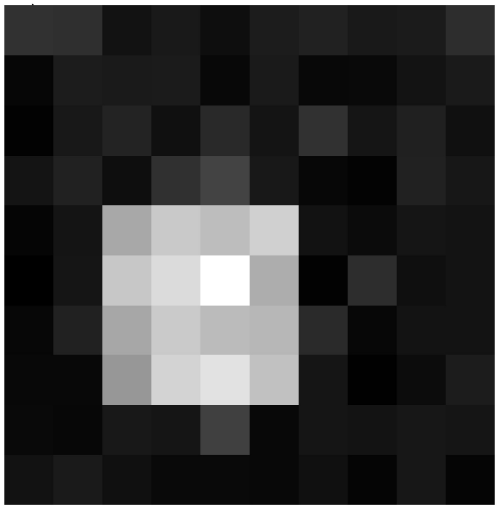
ahol M_i a kép(részlet) i. pixeljéhez tartozó érték (jelen esetben lineáris gyengítési együttható). Minél jobban hasonlít az eredeti eloszlásra a rekonstruált kép, annál jobb a képminőség, és annál kisebb az L2 norma.

4.3 Tesztesetek

Az első két fantom rekonstrukcióját csak érdekességképp mutatom meg, ezek voltak a legelsők, amiket a program rekonstruálni tudott, a 4. ábrán látható 10x10-es, és az 5. ábrán látható 30x30-as fantom. Mind a kettőn egy-egy négyzetnek kell látszódnia, a bal oldalin centrális pozícióban, a jobb oldalin pedig a sarok felé. Látható, hogy a rekonstrukció konvergál mindkét esetben.

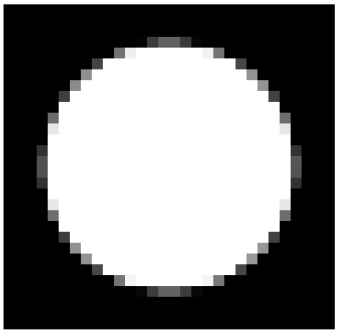


4. ábra 30x30 voxeles fantom

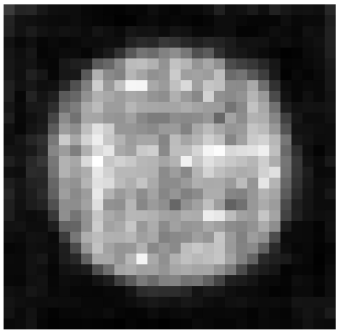


5. ábra 10x10-es fantom

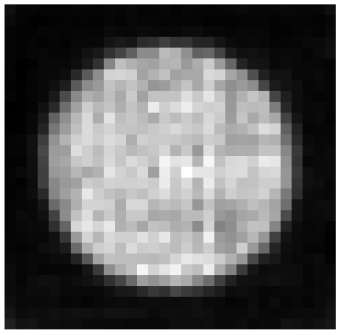
Azt vizsgáltam a kör alakú fantomon elvégzett esetekben, hogy meddig érdemes folytatni az iterációt, vagyis mikor lehet leállítani. Ehhez néztem az L2 norma értékének alakulását, és ez alapján a 20. iterációnál állítottam meg a program futását, mert onnantól romlani kezdett a kép minősége. A 7. és a 8. ábrát összevetve látható a rekonstrukció javulása az iteráció számának növelésével. A kör alakú fantom 30x30-sa méretű volt. A háttér gyengítési együtthatója nulla volt.



6. ábra Eredeti

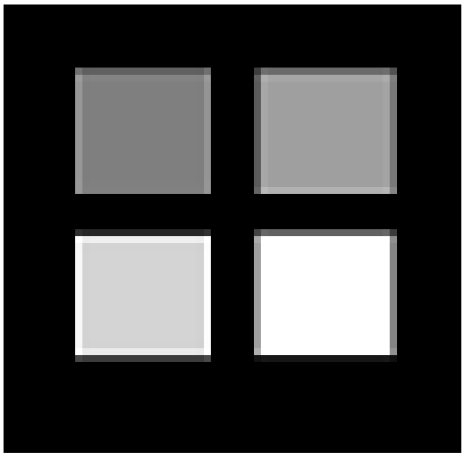


7. ábra 10 iteráció után

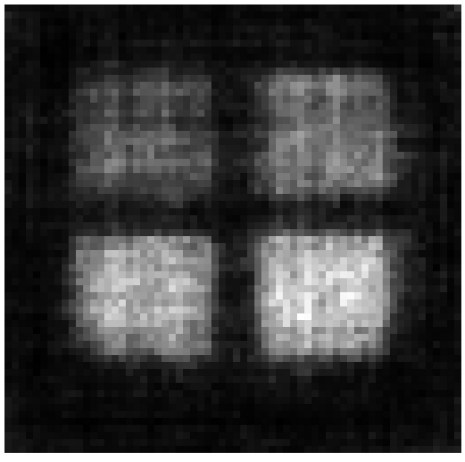


8. ábra 20 iteráció után

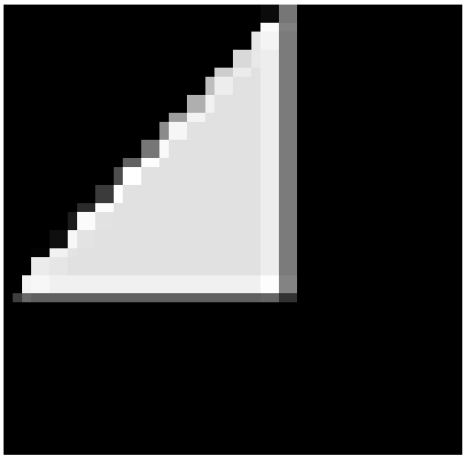
Megnéztem különböző gyengítési együtthatójú homogén részletek rekonstrukcióját is, ez látható a 9. ábrán. Ez már egy 64x64-es fantom volt, amin a legvilágosabb négyzet a négyszeres gyengítési együtthatójú a legsötétebbhez képest, a többi a kettő között van lineárisan.

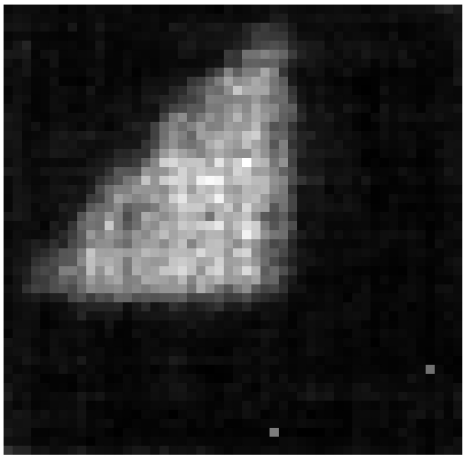


9. ábra Eredeti 64x64-es kép



10. ábra Rekonstruált 64x64-s kép





A fenti ábrák 30x30-as felbontásban készültek, a háromszög szépen felismerhető a rekonstruált képen is, pedig nem túl jó a felbontás. Azt várom, hogy nagyobb felbontású képeken sokkal jobban fog működni az algoritmus (kisebb úthosszak miatt jobb a közelítés), és a v3 algoritmus gyorsan fog majd futni grafikus kártyán.

5. Irodalomjegyzék

[1] Kiss István : "*CT berendezés képrekonstrukciós algoritmusának implementálása grafikus kártyán*", Diplomamunka

[2] K. Lange, R. Carson : "*EM Reconstruction Algorithm for Emission and Transmission Tomography*", **Jornal of Computer Assisted Tomography**, 8(2):306-316

[3] Jakab G. : "*Hibrid Monte Carlo CT szimuláció a GPU-n*", KÉPAF 2013- a Képfeldolgozók és Alakfelismerők Társaságának 9. országos konferenciája

[4] Molnár Balázs: "*Iteratív képrekonstrukciós eljárás gamma tomográfiához: tesztelés és verifikáció*", Szakdolgozat