**SZEGEDI TUDOMÁNYEGYETEM**  
Természettudományi és Informatikai Kar

Képfeldolgozás és Számítógépes Grafika Tanszék

**Diplomamunka**

Automatikus magpontkiválasztás régiónövelés-alapú agytumor-szegmentáláshoz MRI-felvételeken

**Varga-Betlehem Ádám**   
Programtervező informatikus MSc szakos hallgató

Témavezető: Dr. Kardos Péter, adjunktus

Szeged

2025

# Feladatkiírás

Feladatkiírás

# Tartalmi összefoglaló

Tartalmi összefoglaló utólag

# Tartalomjegyzék

[Feladatkiírás 2](#_Toc212810658)

[Tartalmi összefoglaló 3](#_Toc212810659)

[Tartalomjegyzék 4](#_Toc212810660)

[Bevezetés 5](#_Toc212810661)

[Irodalmi áttekintés 6](#_Toc212810662)

[1. Szegmentálás 6](#_Toc212810663)

[2. Az MRI 6](#_Toc212810664)

[A Program megvalósítása 7](#_Toc212810665)

[Megvalósított algoritmusok 8](#_Toc212810666)

[Eredmények 8](#_Toc212810667)

[Irodalomjegyzék 8](#_Toc212810668)

[Nyilatkozat 8](#_Toc212810669)

# Bevezetés

Az agydaganatok diagnosztizálása egy létfontosságú kérdés a megfelelő kezelési stratégia megválasztásához és a beteg túlélési esélyeinek javításához. A leggyakrabban az agy vizsgálatára mágneses rezonancia képalkotást (MRI) alkalmazzák, mivel ez a képkészítési eljárás képes részletes és kontrasztgazdag képet adni a lágyszövetekről, de ennek ellenére egyéb képkészítő eljárást is használhatnak, mint például a röntgen alapú CT-t, ami hozzáférhetőbb és gyorsabb. Az így készült képek értelmezése függ az orvos tapasztalatától, szakértelmétől és időt igényel, így szükség lehet automatikus képfeldolgozó és szegmentáló eljárásokra, amik segítik az orvost munkája során.

Jelen esetben a szegmentálás célja az, hogy megkülönböztessük a tumort az agy többi részétől, ezzel gyorsítva és objektivizálva az orvos munkáját. Napjainkban a legnépszerűbbek a mélytanulásos módszerek, de nem szabad megfeledkezni a klasszikus szegmentáló módszerekről se. Ilyen módszer például a régiónövelés, aminek eredményességét nagyban befolyásolja a kezdőpontok (seed pontok) kiválasztása, ami sokszor manuális feladat. Maga a régiónövelés elve egyszerű, viszont a nem megfelelő seed pontok kiválasztása alul-, felülszegmentálást eredményezhet vagy akár hibás régiókat is. Ezért fontos egy olyan automatikus seed kijelölési eljárás, amivel minimalizálni tudjuk az emberi beavatkozást és növelni az eljárás reprodukálhatóságát. Ebben a diplomamunkában automatikus seed kiválasztó eljárásokat fogunk megnézni és megvalósítani, esetleg javítani.

# Irodalmi áttekintés

## Szegmentálás

A szegmentálás célja, hogy a képeket olyan részekre osszuk fel, amelyek megfelelnek a valós leképezett objektumoknak, területeknek. Általában az előfeldolgozás után következik és megelőzi a jellemző kinyerést, ha a gépi látás moduláris rendszerének modelljében próbáljuk elhelyezni. Egy nehéz feladatról van szó, ami egyáltalán nem egyértelmű. Orvosi képeken a sebésznek tudnia kell mit kell eltávolítania, a radiológusnak tudnia kell melyik részt bombázhatja sugarakkal, de azt is tudnia kell mely részeket nem érheti sugárzás. A szegmentálás bonyolultságából adódóan számos módszert kifejlesztettek, köztük küszöbölés, él-alapú, régió-alapú, illesztésen alapuló szegmentáló módszereket, hogy a BSc-n tárgyalt alap-módszereket említsük. A küszöbölésnek az alapgondolata az, hogy az adott régióra tudunk mondani egy intenzitás értéket, küszöböt, ami felett vagy alatt minden érték az adott szegmenshez tartozik. Sokféle küszöb választási stratégia létezik, talán a legismertebb példa erre az Otsu módszere, ami olyan küszöbértéket keres, amire minimális az osztályon belüli variancia, ezzel megpróbálva elválasztani a hátteret az előtértől. A régió-alapú szegmentálás az objektum által elfoglalt területet határozza meg. Ilyen módszer a régió növelés, ahol első lépésként kiválasztunk egy pontot vagy egy biztosan összetartozó pontokból álló kezdeti régiót. Minden lépésben megvizsgáljuk az aktuális régióval határos pontokat, majd hozzáadjuk azokat a régióhoz, amik kielégítik az adott hasonlósági kritériumot. Az algoritmus megáll, ha már nem tud több pontot hozzáadni a régióhoz. Ebből is látszik, hogy létfontosságú a magpontkiválasztás, ami nagyban feladatfüggő. Ebbe a kategóriába eső más módszer még a szétválasztás és egyesítés, ahol az egyesítés különböző régiókat von össze egy kritérium alapján, míg a szétválasztás esetén a régiók nem elégítik ki a feltételt. Rendszerint a képet, régiót négy egyenlő részre osztjuk szét, amit egy hierarchikus négyes fa adatstruktúrában tárolunk, ahol minden csúcspont a négy feldarabolt részére mutat.

## Az MRI

A mágneses rezonancia képalkotás (MRI), azon alapul, hogy egy atommag spinjének erős mágneses térben két állapota van. Párhuzamos, ami az alacsony energia szintű állapot, és az antipárhuzamos, ami a magas, és megfelelő hullámhosszúságú rádióhullámokkal az alacsonyból a magasabba vihető, ez a gerjesztési folyamat. Utána elindul a relaxációs folyamat, amiben a spinek visszatérnek nyugalmi, azaz az alacsony energiaszintű állapotukba, miközben rádiósugárzást bocsátanak ki. Ezt a folyamatot többször ismételjük TR időnként a jel-zaj viszony javítása érdekében, végül ennek a sugárzásnak az erősségének és időbeni változásának mérésével kapjuk a képet. A T1 a longitudinális, míg a T2 a transzverzális relaxációs idő. Mindkettő a térrész fizikai és kémiai tulajdonságától függ, és a T2 jóval rövidebb ideg tart, mint T1. T1 súlyozott képet kapunk akkor, ha az ismétlési idő (TR) rövid, mivel nem telik el elég idő, hogy bizonyos szövetek longitudinális magnetizációja (T1) visszaálljon, és a 90o-os pulzus és maximális jel között eltelt idő (TE) is kicsi vagy nincs, így T2 különbségek nem érvényesülnek. T2 súlyozott képet kapunk, ha az ismétlési idő hosszú, hiszen így T1 relaxációban nincs jelentős különbség, és ha TE hosszú, akkor a T2 relaxációban lényegesebb különbségek keletkeznek. A FLAIR (Free Liquid Attenuated Inversion Recovery) kép inverziós szekvenciával készült T2 kép, ahol a folyadék nem ad jelet, mert az első impulzus után addig várunk, amíg a folyadék mágnesezettsége nulla nem lesz és utána adjuk ki a következő impulzust. Mi ilyen FLAIR képeken, annak is a transzverzális szeletein fogunk dolgozni.

# A Program megvalósítása

Az alkalmazás Python nyelven íródott. A felhasználói felület Pysied6 és matplotlib könyvtárral lett megvalósítva. A felületen töltjük be a képeket, választjuk ki a futtatni kívánt algoritmust, és állítjuk be a paramétereket. A számításokhoz numpy, opencv és egyéb könyvtárakat használtam fel. A program egy MR képet fog szegmentálni régió növelési algoritmussal, így lehetséges felbontani a feldolgozást diszkrét, moduláris lépésekre. Ennek a megvalósítása a feldolgozási lánc vagy csővezeték (pipeline) elv szerint történik, vagyis egymás után végrehajtott lépések sorozatával. Először az előfeldolgozási lépések futnak le, majd a magpont-kiválasztás, végül pedig a régiónövesztés. Minden lépés ugyanazon a *PipelineContext* objektumon keresztük kommunikál egymással, amiben tároljuk az eredeti képet, a különböző lépések eredményét és a felhasználó által beállított paramétereket. Mivel ezek a lépések különböző megvalósításokat követhetnek, így a megvalósításukra a stratégia mintát (strategy pattern), használtam fel. Azaz minden lépés egy azonos interfészt valósít meg, így tetszőlegesen cserélhetők, anélkül, hogy a program bármelyik másik részéhez hozzá kelljen nyúlni. Az előfeldolgozási lépések a *process(ctx)*, a magpont-kiválasztás a *select(ctx)*, míg a régiónövelés a *run(ctx)* metódust hívja, így a lépések nem egy interfészből származnak, így jól elkülönülve egymástól. A különböző algoritmusokat a gyártó minta (factory pattern) segítségével hozzuk létre. A *PipelineFactory* osztály minden módszerhez, egy előre definiált pipeline-t épít fel, amit statikus függvényként lehet meghívni, így a felhasználói felületen elég kiválasztani melyik módszert szeretnénk futtatni. Minden pipeline deklarálhatja, milyen vezérlőelemekre van szüksége ahhoz, hogy a lépések paramétereit a felhasználó beállíthassa. Ezen felsorolt elemek egy *ControlPanel* ősosztályból származnak, és csúszkákat, számbeíró mezőket valósítanak meg. Ezen paneleket a *ControlWindow* osztály kezeli és jeleníti meg a felhasználó számára, amikor megváltoztatja az adott pipeline-t. Ez a megközelítés a kompozit (composite) és építő (builder) mintákat ötvözi.

# Megvalósított algoritmusok

A Cikkekről kéne írni, meg az én megvalósításomról, majd módosításokról

# Eredmények

eredmények

# Irodalomjegyzék

# Nyilatkozat