פרויקט: Medical Images Segmentation

פרויקט גמר קורס עיבוד תמונה וראייה ממוחשבת.

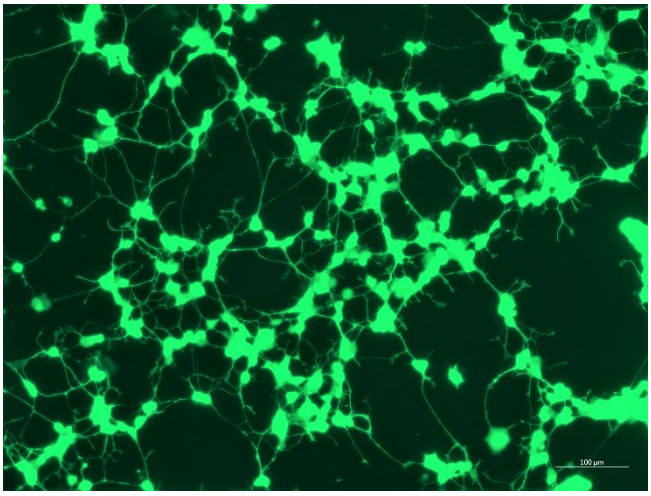
מגישות: עדי ויסברג: 313245268, יעל חוה: 313417420 ונעמי צברי: 207642638.

.**מרצה**: גיל בן ארצי

Introduction:

ניתנו לנו תמונות של דגימות מרקמות המוח ממעבדה שנצבעו בצבע מיוחד המאפשר זיהוי של נוירונים ואקסונים. זאת על מנת לפתח שיטה שתעזור לזיהוי ההתקדמות והשינויים בתאי המוח שתעזור בניתוח שיעור השיפור של ההחלמה ממחלת הסרטן.

כדי לעשות זאת ראשית יש לזהות ולבודד את האקסונים (הקווים הדקים) חישוב האורך והעובי הממוצע שלהם, עם נתונים אלו החוקרים יוכלו ביתר קלות לעשות השוואה בין צילומים מזמנים שונים ולזהות את השינויים.



Approach and Method:

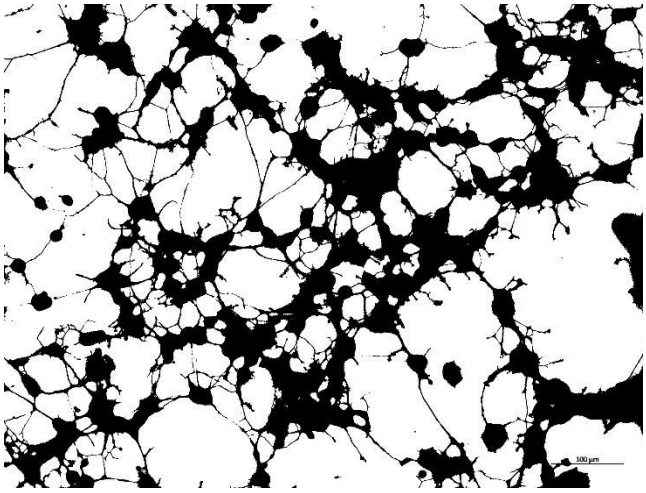
האלגוריתם:

בהינתן התמונה:

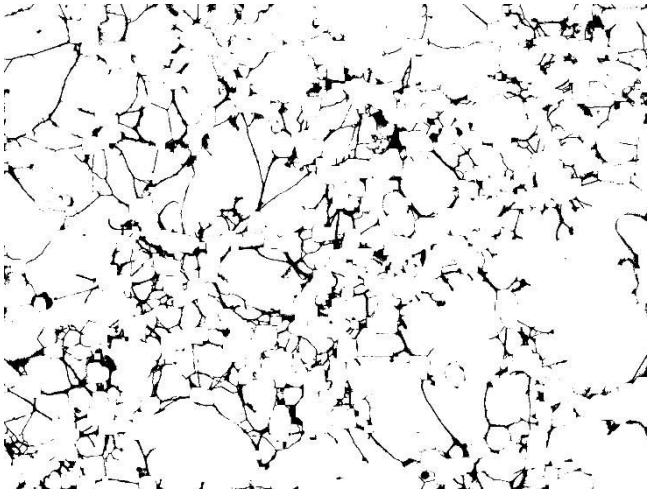
1. הפיכת התמונה ל-2 תמונות בינאריות:

<u>תמונה בינארית mask-</u> קבענו threshold גבוהה יחסית (על פי ההנחה כי נוירונים יותר בהירים מאוקסונים) שלפיה יצרנו תמונה בה בודדנו את הנוירונים בלבד.

1. (א) לאחר מכן ביצענו פעולת erode על מנת להרחיב את היקף הנוירונים כדי שנוכל להשתמש המסכה זו בשלב הבא "למחיקת" הנוירונים כולל קווי המתאר שלהם.



1. (ב) תמונה 2 - נהפוך את התמונה המקורית לבינארית לפי threshold נמוך יותר כדי לקבל גם את האקסונים שהפיקסלים שלהם בהירים יותר.



2. נוריד מתמונה 2 את ה mask - (בעזרת שלב 1-א שבו ביצענו erode קיבלנו את האקסונים כמעט ללא קווי מתאר של הנוירונים). נעבור על כל הפיקסלים שבתמונה: אם בmask הפיקסל הוא שחור זהו נוירון ולכן לא נסמנו בתמונה הנוכחית (= צבע לבן), כך קיבלנו תמונה ללא נוירונים.

3. נשלח את התמונה הבינארית 2 לפונקציה -connected_comp שתמצא לנו את רכיבי הקשירות בתמונה, ז"א תביא לנו "חלונות" שבהם יש אוקסון אחד שלום, מחובר.

4. נשלח את רכיבי הקשירות ל hough line כדי למצוא את משוואות הישר הנחתכות בכל רכיב קשירות (כיוון שלפעמים ישנה קבוצת אוקסונים וכך נקבל "גוש" אוקסונים ולא קווים בודדים וזה יקשה לחישוב אורך והעובי) ואז נעבור על כל רכיב ונשמור במילון את הפיקסלים שעל כל משוואת ישר) נחשב האם פיקסל נמצא בטווח של הישר ולא בהכרח על הקו ממש (בדקנו עבור סטייה של 10 בהפרש בין rho לבין יתר המשוואה).

בשלב זה ניסינו 2 שיטת לפתרון הבעיה- מציאת אורך ועובי:

שיטה 1:

5. עבור כל המערכים של נקודות (שנמצא בטווח של קו) נמצא את משוואת הפולינום (ממעלה 2) של הקו מכיוון שנרצה לדייק, מפני שישר אינו טופס בצורה מלאה את העקום.

6. <u>עבור חישוב אורך האוקסונים</u>: נמצא את הX1 של הנקודה השמאלית למטה ביותר של רכיב הקשירות ואת הx2 עבור הנקודה הימנית למעלה, ונמצא את אורך העקום מx1 עד x2 לפי משוואת למציאת עקום: $\int_a^b \sqrt{1+(f'(x))^2}dx$ לפי x1 ו x2.

נסכום את תוצאות חישוב אורך העקום עבור כל פולינום לסה"כ אורך האקסונים בתמונה.

7. <u>עבור חישוב עובי ממוצע של האוקסונים:</u> כדי למצוא את העובי נעבור על כל פולינום ופולינום שמצאנו ונעשה עליו את הפעולות הבאות:

נקפוץ בלולאה מהפיקסל הכי שמאלי למטה של רכיב הקשירות עד הפיקסל הכי ימני למעלה בקפיצות של 5. עובר כל פיקסל שמצאנו במהלך הלולאה נחשב עבורו את משוואת המשיק ולפיה את משוואת הנורמל שלו,

וכך מצאנו את הישר שחותך את הפיקסל, שהוא למעשה העובי.

נבדוק כמה פיקסלים נמצאים על הנורמל בתמונה:

נייצר חלון בגודל5X5 מסביב לפיקסל נעבור עליו ונבדוק כל פיקסל האם הוא:

1. מואר (לבן) 2.ונמצא על משוואת הנורמל (אחרת נחשב פיקסלים שהם לא העובי) .

נסכום את תוצאות הדגימות ונספור את כמות הדגימות. נעשה ממוצא בין מספר הדגימות והתוצאה

הכוללת. כמות הדגימות/סכום כולל . וכך מצאנו את העובי הממוצע.

שיטה 2:

תחימת כל contour בקופסא מינימלית וחישוב מימדיו.

1. מצאנו עבור רכיבי הקשירות את הקונטורים התואמים עבורם (גבולות הרכיבים) באמצעות findContours .
2. עברנו על כל קונטור שנמצא במידה והשטח שלו גדול מפיקסל 1 ומצאנו:
   - נקודות המתארות גבולות של "קופסא" מינימלית המקיפה כל רכיב, אשר יכולה להיות מסובבת לכל הכיוונים.
   - סידרנו את הנקודות כך שיופיעו בפינה הימנית-עליונה, ימנית-תחתונה, שמאלית-עליונה ושמאלית תחתונה כדי שנוכל לשרטט את הקווים התוחמים את רכיב הקשירות ויוצרים את ה"קופסא".
   - חישבנו את נקודת האמצע בין הנקודה השמאלית-עליונה לנקודה הימנית עליונה וכן את נקודת האמצע עבור הנקודה השמאלית-תחתונה לימנית-תחתונה.
   - לאחר מכן חישבנו את המרחק האוקלידי בין 2 נקודות האמצע שמצאנו על מנת למצוא את מימדי כל רכיב.
3. לאחר שמצאנו את המימדים עבור כל רכיב - ביצענו ממוצע כללי של האורך והרוחב של האוקסונים והמרנו בחזרה לפיקסלים מיחידת מידה Inch.

הקשיים שנתקלנו בהם:

במגבלת הזמן והתקופה לא ממשנו מספר חלקים ונתקלנו בשגיאות שעדיין לא נפתרו, והם:
לא הגענו לתוצאה מספיק רצויה ב hough line ולכן החלטנו שלא לכלול אותו בקוד כרגע ולהניח את ההנחה: עבור כל רכיב קשירות נקבל רק אוקסון אחד ולא מספר אוקסונים, ז"א עבור כל רכיב קשירות קיבלנו רק עקומה אחת,כמובן שהנחה זו אינה מדויקת ולכן הביא לחישובי אורך אינם מדויקים אך בעיה זו ניתן לפתור ואינו פוגע באלגוריתם.
<u>בחישוב העובי-</u> נתקלנו בכמה מצבי קצה כדוגמת- קורדינטה שלישית או יציאה מהמערך תוצאות ששינו את התוצאה הרצויה אך הדבר פתיר ואינו פוגע ברעיון האלגוריתם המפורט.

method:

<u>openImage</u>: העלאת תמונה לקוד.

<u>toBinary</u>: המרת התמונה מ rgb לבינארי, תוך התייחסות לthreshold המתאים.

<u>deleteNeuron</u>: מחיקת הניורון לפי הmask ,רץ על 2 התמונות יחד, כאשר בmask המספר 255

(ז"א חלק מנוירון) אז נמחק אותו – נהפוך ל0 (שחור).

findDots: מוצא את הנקודות הנמצאות על הקווים שמצא ה hough line (שמוצא קווים בתמונה).

מייצר dictionary עבור כל הקווים, עובר קו קו ומוצא לפי משוואת הישר (_____) האם הפיקסל נמצא על הנקודה או קרוב אליה בסטייה של _____ ושומר כל פיקסל ב key של הישר הנ"ל.

method1: שיטה 1 (לפי המפורט למעלה)

polynom: עבור רכיב קשירות תמצא את הפולינום המתאים לו ממעלה 2.

findLengthofOxons: מפורט בשיטה 1 שלב 6.

integral_calc: חישוב האינטגרל עבור אורך עקום לפי הפירוט בשיטה 1 שלב 6.

findThickOfEons: מציאת העובי כמו שמפורט בשיטה 1 שלב 7.

findTangent: מציאת משוואת המשיק עבור פולינום כמו שמפורט בשיטה 1 שלב 7.

findRationa: מציאת משוואת הנורמל עבור הפולינום כמו שמפורט בשיטה 1 שלב 7.

findThickAndLength: קריאה לשני הפונקציות findThickOfEons ו findLengthofOxons.

הטווחים/thresould/ סטיות מומלצות:

נעשו testing שונים למציאת הthreshold האפקטיבים ביותר.

הגענו למסקנה שה threshold המומלצים הם:

Mask- 150

image binary - 45

ושיטות שונות לזיהוי העקומים ודרכים שונות לזיהוי העובי והאורך.

מהם הגבולות -טווח ההצלחה.

איך זה עובד ומדוע.

Results:

קשיים שנתקלנו בהם:

קושי 1 בפרוייקט זה היה במציאת הthresould הרלוונטים לכל תמונה שנקבל ובנוסף בידוד האקסונים מהגרעין הנוירון כך שנקבל תמונה נקייה מנוירונים אך גם לא נאבד מידע נחוץ.

ובכלל רוב התהליך העבודה התמקד במציאת הטווח והסטיות, ניסיון להוריד רעשים.

קושי 2 hough line הינו אלגוריתם מורכב המצריך הבנה מעמיקה בנושא, כיון שנושא זה הינו נושא חדש לנו ובטח שנצרכנו להבנה מעמיקה ביותר שהצריכה ממנו שעות מרובות אך במגבלת הזמן לא הספקנו לקבל תוצאה טובה מספיק.

קושי 3 דרכים יצירתיות למציאת עקומים וישרים שלא בהכרח מזוהים לאורך כל הדרך, כדי לפתור בעיה זו חקנו וחשבנו על דרכים שונות ומגוונות. בסופו של דבר מצאנו פתרונות בעיקר בחישובים של פולינומים ומרחקים אוקלידים. (חשוב לנו לציין שרעיונות אלו פיתחנו בעצמנו ולא נעזרנו בגורם חיצוני).

בנוסף למדנו לממש ולהשתמש במושגים החדשים לנו מעולם הראייה ממוחשבת אשר עד לפני הפרויקט היו בגדר תאוריה מבחינתנו.

ניסינו את הקוד עם כל התמונות שנתנו לנו אך המרצה המליץ להתמקד בתמונה בשם "ima1.tif".

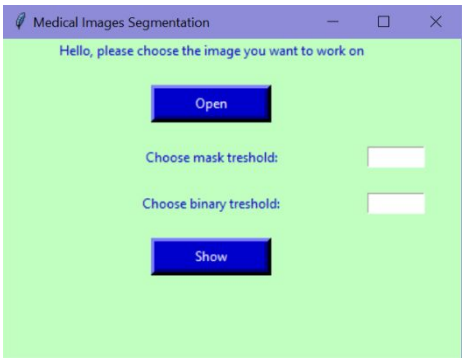
ולכן התמקדנו למצוא את הפתרון המירבי ביותר עבור תמונה זו.

התוצאה הסופית שהגענו אליה:

בסופו של דבר הגענו לתוצאה המתבקשת:

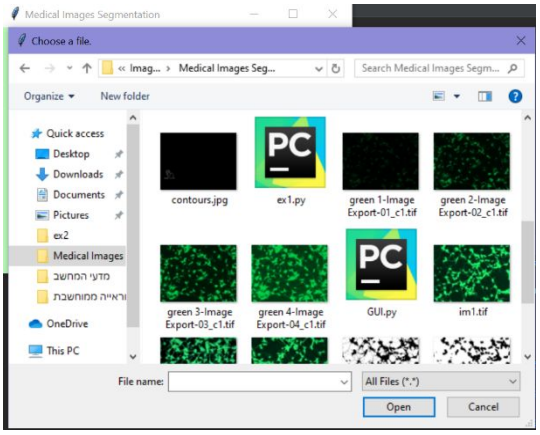
עבור תמונה נתונה מצאנו את האורך והעובי הממוצע המשוער.

בנינו ממשק גרפי פשוט ונוח לשימוש כדי להקל על השימוש והבנת המערכת.

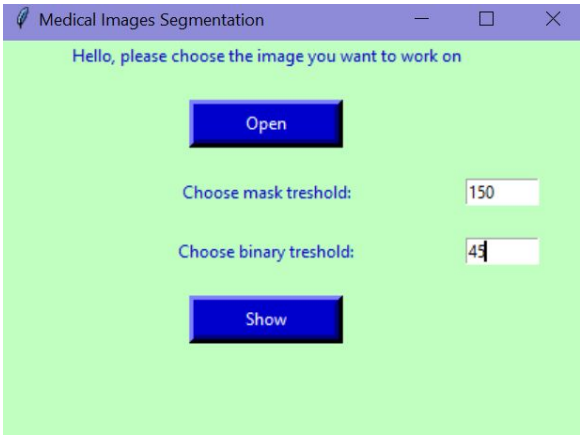


זהו הממשק הגרפי של התוכנה, נפתח אותו כאשר נלחץ על הpaly של המחלקה GUI.

נלחץ על open כדי לבחור תמונה.



נבחר את התמונה הרצויה, מומלץ לבחור: "im1.tif"



נבחר את ה thresuld הרצוי ליצירת 2 התמונות הבינאריות,

מומלץ:

mask – 150

binary thresuld – 45.

ונלחץ show לקבלת התוצאה.