

# שיטת לעיבוד תמונה: GrabCut ו-Blending Poisson

אור פחימה

07/07/2024

## מבוא

בעבודה זו עוסק בשני אלגוריתמים חשובים בתחום עיבוד התמונה: GrabCut ו-Blending Poisson. GrabCut משמש לחיתוך וركע מתמונה וمبוסס על שילוב של מודלים סטטיסטיים והציג גרפים. לעומת זאת, אלגוריתם Blending Poisson מאפשר שילוב חלק של אובייקטים מתוך תמונה שונות על ידי פתרון משוואות דיפרנציאליות חלקיות, וכך יוצר תוצאות טבעיות ומיציאותיות יותר.

## GrabCut

מיימתי את האלגוריתם על פי האמור במאמר (עם קצת תוספות):

- 1. אתחול המודלים:** כפי שפורסם במאמר, חישבתי את המרכזים בעזרת KMeans כדי לאפשר אתחול מהיר ומדויק ככל הניתן. בניתי את המודלים עבור הרקע והקדמה, ואתחלתי את המשקלים והשונוויות של כל אחד מהרכבי המודלים.
- 2. עדכון המודלים:** בשלב זה, כפי שצוין במאמר, אין צורך באתחול מחדש של המרכזים, אלא נשתמש במודלים הקיימים על מנת לתחול מרכזים חדשים לבניית מודלים חדשים עם שונוויות אחרות ומשקלים שונים.
- 3. חישוב החתך המינימלי:** עבור שיטה זו השתמשתי באלגוריתם cut min של החבילת `igraph`. בניית הגרף וכל מה הקשור בו מבוצעים בקובץ אחר בשם `utils.c`.  
(א) **הליק לבניית הגרף וחישוב בטא וקשתות מסוג A ו-D:** מבוצעים על פי הנוסחאות המתמטיות המוזכרות במאמר, כאשר על מנת ליעיל את הריצה, קשתות מסוג A נשמרות על מנת לחשב פעם אחת בלבד בכל ריצת האלגוריתם.
- 4. עדכון המסיכה:** מתבצע בעזרת החתך שנמצא בריצה.
- 5. התכניות :** האלגוריתם עוצר את ריצתו עבור כל אחד מהתנאים הבאים :
  - אם האנרגיה בגרף לא השתנה מעל נקודת סף .
  - אם אף אחד מהפיקסלים לא השתנה באיטרציה . אני מבין שהתנאים אמורים להיות שוקלים , אך ראויים לנכון הויאל והאלגוריתם הסתברותי בסופו של דבר ומתכניות שכאשר לא משתנים פיקסלים אין להמשיך את הריצה.

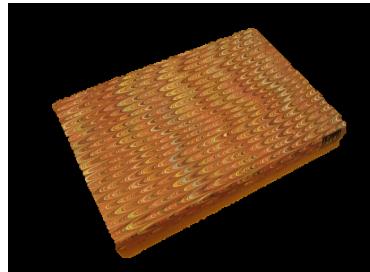
## שינויים ושיפורים שביצעת

□ מצאתי שימושי לשיעיל להעביר את התמונה בפילטר מדיאן, שהוא מבצע טשטוש אך עדין שומר על הקצוטות.

□ בנוסף, לאחר והאלגוריתם מבצע המון חישובים ורץ על הרבה מידע, על מנת ליעיל ולהקטין את זמן הריצה, הקטנתי את התמונה בחצי בתחילת הריצה.

## תוצאות הריצה

להלן התוצאות שהתקבלו מהרצת האלגוריתם על כל התמונות בתיקיה `data` תוך שימוש במלבנים המוכנים:



איור 3: תוצאה עבור תמונה  
book



איור 2: תוצאה עבור תמונה  
banana2



איור 1: תוצאה עבור תמונה  
banana1



איור 6: תוצאה עבור תמונה  
flower



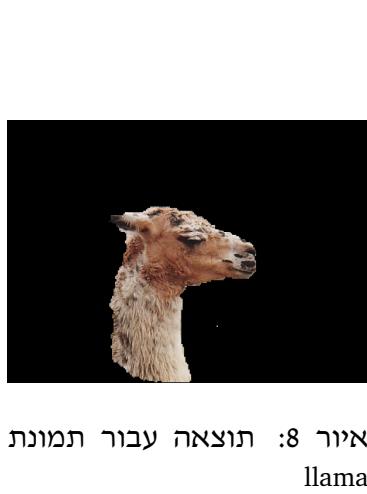
איור 5: תוצאה עבור תמונה  
cross



איור 4: תוצאה עבור תמונה  
bush



איור 9: תוצאה עבור תמונה  
memorial



איור 8: תוצאה עבור תמונה  
llama



איור 7: תוצאה עבור תמונה  
grave



איור 12: תוצאה עברו תמונה  
teddy



אייר 11: תוצאה עברו תמונה  
stone2



אייר 10: תוצאה עברו תמונה  
sheep

## מצדים

Jaccard	Accuracy	name Img
50.36789426161796	74.7705078125	banana1
90.31536192890775	97.56282552083333	banana2
94.18950066151325	97.72688802083334	book
58.54662705021785	88.47703703703704	bush
68.31806703965935	88.15037037037037	cross
97.38406881077039	99.48185185185186	flower
89.31600586527126	98.6237037037037	grave
89.45260347129505	98.13212275972951	llama
87.56269004462347	97.66703703703705	memorial
91.68393609198476	99.53925925925927	sheep
98.1457233927178	99.54361979166667	stone2
96.49977483931715	99.24357704013022	teddy

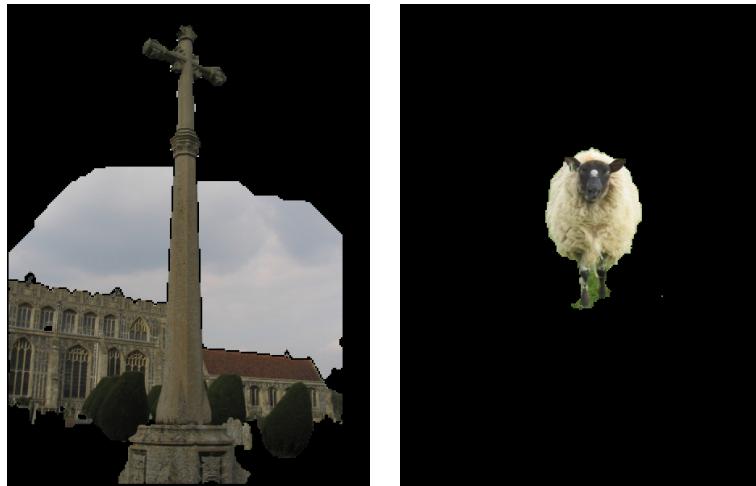
טבלה 1 Results table

## ניתוח הייפר פרמטרים ותוצאה של טשטוש

### השפעת שינוי פרמטר גמא

פרמטר גמא בעצם המכפיל כל קשת מסווג A שלו על עצם על כמה אנחנו משקלים דמיון בין פיקסלים שכנים בריצה. ככל שיגדל האלגוריתם יעדיף להשאיר יותר פיקסלים דומים אחד ליד השני.

עבור גמא = 10



איור 13: תוצאות עبور גמא = 10

ניתן לראות שכאשר אנו מורידים את ערכו אנו מתעדפים פחות דמיון בצבעים מה שהגרים לירידה בביטויים על התמונה של הצלב, למורות שלהפתעתני יש לומר בתמונה של הכבשה מתקבלת תוצאה טיפה יותר טובה.

עbor גמא = 200



איור 14: תוצאות עBOR גמא = 200

על אותו ההגיון כאשר גמא מאד גדול האלגוריתם פחות מעדיף להפריד פיקסלים דומים ורואים את ההשפעתו בתמונה של הלמה, למורות שאני חייב לציין שבעור התמונה של בינה 1 בשום שינוי של פרמטר לא הצליחתי הגיעו לתוצאה טובה כמו זו.

## 0.1 כמות הגאוסיאנים

בדקתי את המודל עבור 2 ו-10 גאוסיאנים. אציין שלא ראיתי שינוי דרמטיים שווה לציון. אינטואטיבית ברור לי שככל שהוא פחות גאוסיאנים אני עלול להגיע לתוצאה של "חוסר התאמת" למול מצב שבו יש יותר מדי גאוסיאנים ועלולים הגיעו למצב של "התאמת יתר".



איור 15: השוואת כמות גאוסיאנים - 2 גאוסיאנים (שמאל) ו-10 גאוסיאנים (ימין)

## 0.2 השפעת טשטוש

כמו שציינתי בפסקת המימוש, אני השתמשתי בפילטר מדיאן שהביא לי תוצאות טובות. האמת שגם פילטר הגאוסיאן שימושי לטשטוש עוזר להגיע לתוצאות טובות בתמונות מסוימות.

### 0.2.1 טשטוש קטן

השתמשתי בקרנל של  $5 \times 5$



איור 16: תוצאה עברו טשטוש קטן - קרナル  $5 \times 5$

ניתן לראות שהשימוש בפילטר זה (במידה נכונה) אכן נותן תוצאה טובה יותר עברו התרמונה של הצלב.

### 0.2.2 טשטוש גדול

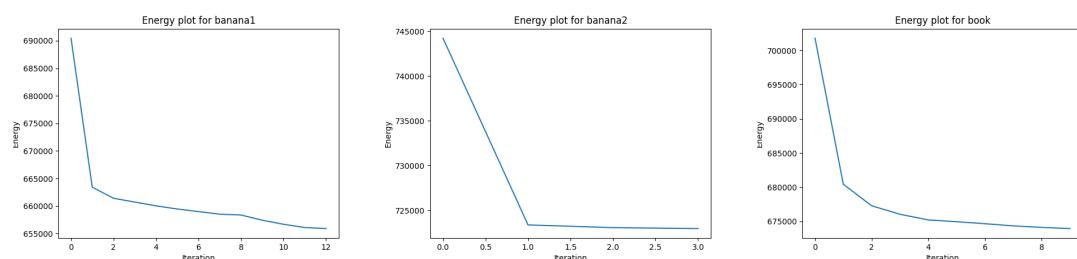
השתמש בקורס של  $21 \times 21$  אינטואיטיבית ברור שטשטוש במידה לא נכונה עלול לטשטש עבורנו את הנקודות של האובייקט ובכך ליצור סגמנטציה שהיא פחות טובה.



אייר 17: תוצאה עבר טשטוש גדול - קernel  $21 \times 21$

### 0.3 זמני ריצה

לאחר הקטנת התמונה בחצי, זמני הריצה הממוצעים הם 8 - 30 שניות, בהתאם לפיזור הצלבים בתמונה, גודל המסיכה, כמוות הפיקסלים. האלגוריתם מתכנס לרוב לאחר 5 - 12 איטרציות. כמו כן, ניתן לראות את התנהלות האנרגיה עבר ריצות נבחרות:



אייר 18: התנהלות האנרגיה עבר ריצות נבחרות

**0.4 מגבלות המימוש וקשיים**

כפי שניתן לראות מהתוצאות בריצות, כאשר הצלבים של האובייקט אותו רוצים לחתוך דומה מאוד לצבעי הרקע שלו, לאלגוריתם קשה מאוד לסמן את האובייקט (עד כדי חוסר הצלחה ברוב המקרים). ניתן לראות זאת היטב בתמונות של בינהו והעצי:



איור 19: תוצאות ריצה עברו בינהו והעצי

בנוסף, גילוי נאות: הקטנתה את המסיכה המוכנה של התמונה של הצלב. זה עוד אחת מمبرשות הריצה כאשר אין מספיק פיקסל רקע לאלגוריתם להתבסס עליהם, הוא לא מצליח בכלל לזהות את האובייקט. ניתן לראות את התוצאה לפי שינוי ערכי המלבן:



איור 20: תוצאות ריצה עברו הצלב עם מסיכה מוקטנת

## 1 מימוש Blending Poisson

תחילה, האלגוריתם ממיר את תמונות המקור והיעד לפורמט `float32` כדי למנוע גלישת ערכיים במהלך החישוב. לאחר מכן, הוא מזהה את האינדקסים של הפיקסלים בmseca ומחשב את אופרטור הלפלסיאן. אופרטור הלפלסיאן משמש לחישוב ההבדלים בין פיקסלים שכנים בתמונה.

האלגוריתם יוצר מטריצת דיליה A ומטריצת  $\delta$  שמייצגות את מערכת המשוואות הלינאריות שיש לפתור. מטריצת A כוללת את המשקלים של קשותות מסוג N וחיבור הפיקסלים בmseca לפיקסלים הסובבים בתמונה המקורית, בעוד מטריצת  $\delta$  כוללת את התרומות של הערכים מתמונה היעד ומהתמונה המקורית.

לאחר מכן, האלגוריתם פותר את מערכת המשוואות עבור כל ערוץ צבע בנפרד וմשלב את התוצאה באזורי mseca בתמונה היעד. התוצאה הסופית נשמרת כתמונה `uint8` לאחר עדכון הערכים לטווח התקני  $[0, 255]$ .

### השפעת mseca לא הדקה

בעמוד הבא מוצגות תוצאות הריצה עבור כל המסיכות שהתקבלו. ניתן לראות שגם המסיכה שהתקבלה לא הייתה מדוקית, כמו בתמונות של בננה וועצץ, האלגוריתם מצליח להתמודד עם הקושי ולהניב תוצאות טובות.

## 2 תוצאות הריצה



Book



2 Banana



1 Banana



Flower



Cross



Bush



Memorial



Llama



Grave



Teddy



Stone



Sheep

אייר 21: תוצאות Blending Poisson עברו התמונות השונות.