Image Processing - Exercise 4

Neriya Ben David, neriyabd, 206698581

מבוא

התרגיל עוסק בשילוב בין זוג תמונות כך שהתמונה הראשונה הינה תמונה ברזולוציה נמוכה שמכילה אובייקט מרכזי, ואילו התמונה השנייה מכילה את האובייקט המרכזי מהתמונה הראשונה ברזולוציה גבוהה.
מטרת התרגיל היא לייצר תמונה משולבת כך שהרקע בתמונה המשולבת יהיה הרקע של התמונה הראשונה והאובייקט המרכזי של התמונה החדשה יהיה האובייקט המרכזי ברזולוציה גבוהה שלקוח מהתמונה השנייה.

הכלים המרכזיים בהם השתמשתי:

- שימוש באלגוריתם $SIFT$ לשם יצירת $Descriptors$ לכל תמונה.
- שימוש באלגוריתם KNN למציאת השכנים הקרובים עבור כל $Decriptor$ בין התמונות.
- שימוש ב-$Ransac$ על מנת לסווג בין התאמות לא נכונות להתאמות נכונות
- ביצוע טרנספורמציה על תמונה ו- $blending$ ע״י שימוש במסכה ובקרנל גאוסייני.

אלגוריתם

הערה: אקרא לתמונה בעלת הרזולוציה הנמוכה תמונת יעד ואילו לתמונה עם הרזולוציה הגבוהה תמונת מקור.

אפרט את שלבי האלגוריתם הראשי ולאחר מכן ארחיב על כל תת - אלגוריתם שהשתמשתי בו:

1. קריאת קבצי שתי התמונות בצבע ובאפור.
2. שימוש באלגוריתם $SIFT$ למציאת $keypoints$ ו- $descriptors$ עבור אותם $keypoints$.
3. מציאת התאמות בין $descriptors$ בין תואמים בין שתי התמונות.
4. קביעת רף סינון עבור $descriptors$ תואמים שמצאנו בין שתי התמונות באמצעות מזעור המרחק בין שני ה-$descriptors$ הקרובים ביותר בתמונת היעד עבור כל $descriptor$ נתון בתמונת המקור.
5. ביצוע אלגוריתם $Ransac$ על ה-$descriptors$ הנותרים וסינון $outliers$.
6. הפעלת הטרנספורמציה המתקבלת באמצעות ה-$descriptors$ על תמונת המקור.
7. יצירת מסכה מותאמת על מנת לבצע $blending$ בין תמונת המקור לתמונת היעד.
8. החזרת התמונה המשולבת.

תתי אלגוריתם הממומשים כפונ' כחלק מהאלגוריתם:

- $\boldsymbol{SIFT}$ - מקבל כקלט את התמונה בדרגת אפור ו-$None$ בתור המסכה (האזור שבו הוא צריך לחפש) כלומר שיחפש נקודות עניין בכל התמונה. מחזיר כפלט נקודות עניין ואת ה-$descriptors$ שלהם.
- $\boldsymbol{KNN + Filter\ Descriptors}$ - מקבל כקלט את ה-$descriptors$ של שתי התמונות וכמות שכנים רצויה (2) ומחזיר עבור כל $descriptors$ בתמונת המקור את שני ה-$descriptors$ בתמונת היעד שהכי קרובים אליו ומסנן $descriptors$ בתמונה המקור לפי המרחק בין השכנים שלו שמצאנו.
- $\boldsymbol{Ransac + Homography}$ - מקבל כקלט קואו' של נקודות עניין תואמות בשתי התמונות, ו-$threshold$ המתאר את מקסימום השגיאה שאנו מתירים עבור אבחון נקודות בתור התאמה נכונה. מחזיר כפלט את הטרנספורמציה אותה נבצע עבור כל פיקסל בתמונה.
- $\boldsymbol{warpPerspective}$ - מקבל כקלט את תמונת המקור, מטריצה המייצגת את ההעתקה שנרצה לבצע וגודל הממדים של התמונה לאחר הטרנספורמציה. מחזיר את התמונה לאחר הטרנספורמציה בממדים המבוקשים.

פרטי מימוש

אתאר את פרטי המימוש של השלבים שתיארתי:

1. שימוש בפונ' מספריית cv2 על מנת לקרוא קבצי תמונות בצבע ע"י באמצעות פונ' $imread$, המרת התמונות בנוסף לתמונות בדרגות אפור באמצעות פונ' $cvtColor$ ושימוש בפרמטר $COLOR\_BR2GRAY$.
2. יצירת אובייקט באמצעות $cv2.SIFT_create$ושימוש במתודה $detectAndCompute$ של האובייקט עבור דרגות האפור של שתי התמונות על מנת למצוא נקודות עניין ויצירת $descriptors$ עבור נקודות אלו.
3. שימוש בפונ' $knnMatch$ עבור $k = 2$ של אובייקט $cv2.BFMatcher$ על מנת למצוא את שני ה-$descriptors$ מתמונת היעד שהכי קרובים ל-$descriptor$ מתמונת המקור.
4. השארת כל ה- $descriptors$ מתמונת המקור המקיימים $\frac{neighbor_{first}.distance}{neighbor_{second}.distance} < 0.75$.
5. מעבר על כל ה-$descriptors$ מתמונת המקור שנשארו לאחר הסינון והכנסת מיקומי הפיקסלים של נקודות העניין התואמות להם מתמונת המקור ותמונת היעד אל מערכים pos_{kp1}, pos_{kp2}.
   לאחר מכן השתמשתי בפונ' $numpy.reshape$ לשינוי ממדי המערך למערך תלת ממדי בגודל $(number_{keypoints}, 1, 2)$ כך שכל איבר בממד הראשון מייצג נקודת עניין, כל איבר בממד השני מתאר את הפיקסל עבור אותה נקודת עניין והממד השלישי הוא ערכי ה- (x, y) עבור כל פיקסל.

6. שימוש בפונ' $cv2.findHomography$ המקבלת כקלט נקודות העניין תואמות בתמונת המקור ותמונת היעד, מקבלת את הפרמטר $cv2.Ransac$ ו-$ransacReprojThreshold$ (ערך 8 עבור התמונה עם הטירה וערך 0.5 עבור התמונה במדבר), הפונ' מחשבת את ההומוגרפיה מחישוב טרנספורמציה עבור נקודות שהוחשבו כ-$inliers$ באיטרציה בה התקבלו הכמות הכי גדולה של $inliers$, $ransacReprojThreshold$ מבטא את המרחק בין הפיקסלים המקסימלי בהם נקודות העניין ייחשבו בתור $inliers$.
7. שימוש בפונ' $cv2.warpPerspective$ המקבלת כקלט את תמונת המקור, מטריצת הטרנספורמציה וגודל נדרש לאחר טרנספורמציה, על מנת לבצע את הטרנספורמציה על תמונת המקור.
8. יצירת מסכה ע"י הפיכת תמונת הטרנספורמציה לתמונת אפור ע"י $cvt.Color$, שימוש בפונ' $cv2.threshold$ על מנת להפוך את כל הפיקסלים השחורים (המכילים את הערך 0) בתמונה לשחורים ואת כל הפיקסלים המקיימים $image[pixel] > 0$ ללבנים (255).

   לאחר מכן השתמשתי בפונ' $cv2.bitwisenot$ וחילוק ב255 של כל הפיקסלים על מנת להפוך את צבעי השחור והלבן במסכה כך שהאובייקט עליו ביצענו טרנספורמציה יהיה בלבן ואילו הרקע של התמונה יהיה בשחור והפיכת התמונה לתמונה בינארית המיוצגת ע"י 0 ו - 1.

   הגדלת קצוות ערכי הלבן באובייקט המרכזי של התמונה ע"י טשטוש המסכה עם גאוסיין באמצעות $kernel$ בגודל (7, 7) וסטיית תקן של 1, לבסוף שינוי המסכה, למסכה בינארית בה כל ערכי השחור (0) הינם ערכים שלאחר ביצוע הקרנל בעלי ערך 0 וכל הפיקסלים בהם רמת האפור הינם מעל 0 יהפכו ללבנים.
9. שינוי כל האינדקסים בטרנספורמציה שבהם המסכה הינה שחורה לפיקסלים התואמים בתמונת המקור והחזרת התמונה.

היפר פרמטרים, $threshold$ ובחירות נוספות:

בשני האלגוריתמים בחרתי להשתמש ביחס של 0.75 בין השכנים במרחק בין descriptors. בחירה זו נבעה מכך שבשני האלגוריתמים לא נמצאו מספיק נקודות עניין מתחת לרף זה, ואילו ברף זה נמצאו כמות מספקת של נקודות עניין.

עבור תמונת הטירה בחרתי להשתמש בערך 8 עבור threshold באלגוריתם $Ransac$ כיוון שהוא סיווג מספיק נקודות כ-inliers ונתן את התוצאה הכי טובה לאחר ביצוע הטרנספורמציה.

עבור תמונת המדבר בחרתי להשתמש בערך 0.5 עבור threshold באלגוריתם $Ransac$, בחירה זו נבעה מכך שזוהו משמעותית יותר נקודות בצורה נכונה באלגוריתם זה, והיה ניתן למצוא נקודות רבות שמאובחנות כ-inliers עבור גם עבור ערך ה-threshold זה, דבר שהוביל לצמצום שגיאה משמעותית בהתאמת נקודות נכונות.

שימוש בקרנל (7 x 7) נתן את התוצאות הטובות ביותר להגדלה של המסכה הבינארית כפי שמתואר בפרטי מימוש האלגוריתם.

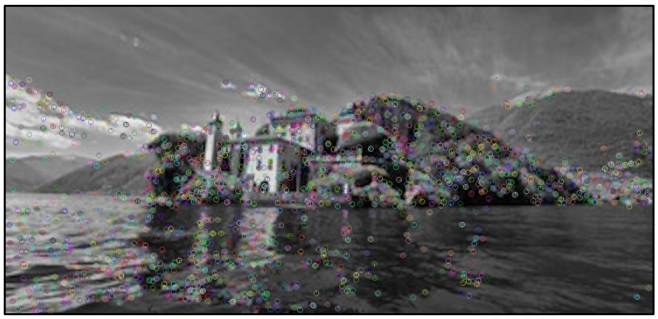
קשיים:

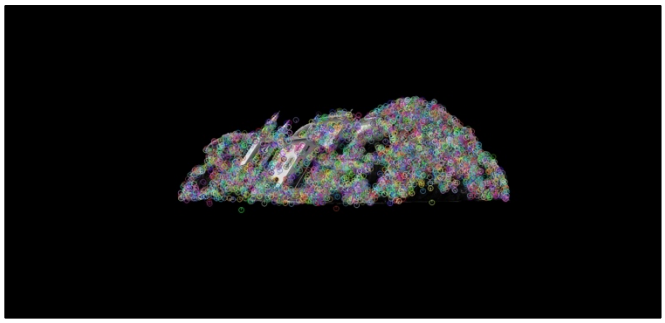
קושי מרכזי בתרגיל היה איך לבצע את ה-$blending$ בין התמונות. ניסיתי להשתמש באלגוריתמים רבים כגון אלגוריתמים שנלמדו בכיתה אך הם אינם נתנו לי תוצאות טובות. לבסוף החשיבה להשתמש במסכה בינארית והחלפת האינדקסים בלבד (תוך הגדלת המסכה להכיל קצת מעבר לאובייקט המרכזי) נתנו תוצאות טובות עד מאוד.

תוצאות וויזואליות

תוצאות וויזואליות עבור תתי אלגוריתמים :

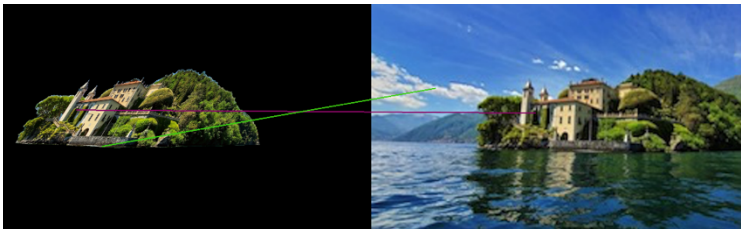
אלגוריתם $SIFT$



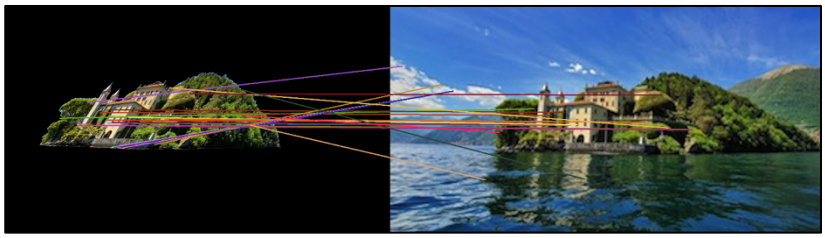


בתמונות אלו ניתן לראות את נקודות העניין שהתקבלו מאלגוריתם SIFT. כפי שניתן לחזות מאלגוריתם זה מתקבלות נקודות עניין רבות שבהמשך נרצה לסנן את "הטובות ביותר" כיוון שאנו זקוקים ל-4 נקודות בלבד כדי לחשב את הטרנספורמציה.

KNN + Filter Descriptors



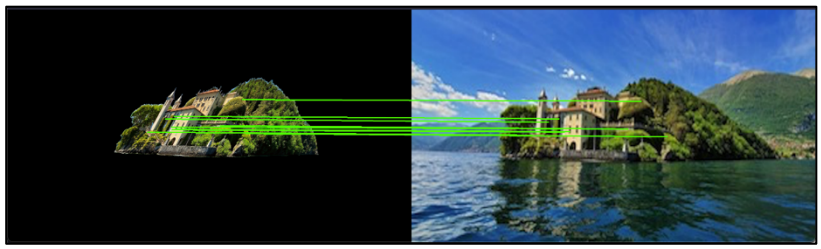
$$\frac{neighbor_{first}}{neighbor_{second}} < 0.6$$



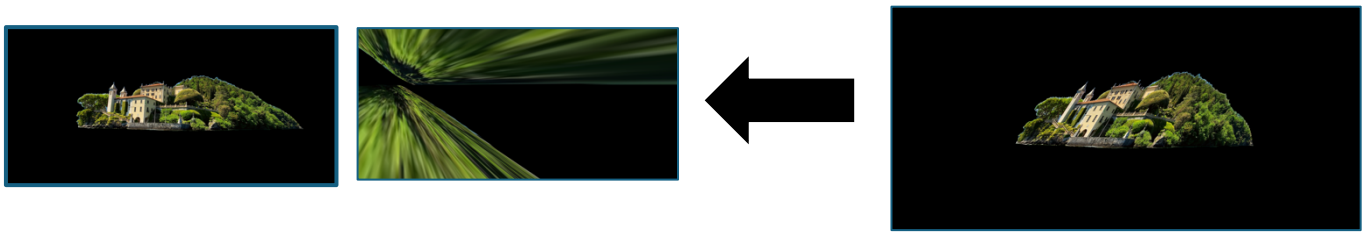
$$\frac{neighbor_{first}}{neighbor_{second}} < 0.75$$

עבור שתי תמונות אלו ניתן לראות את ההתאמה בין נקודות העניין שנשארו לאחר סינון באמצעות מרחק בין שני השכנים הכי קרובים. ניתן לראות בתמונה מימין כי בחירת רף נמוכה מדי של יחס של 0.6 משאירה רק שתי נקודות עניין ורק אחת שאובחנה בצורה נכונה. לעומת זאת, בתמונה השנייה עם סף של 0.75 ניתן לראות כי ישנם הרבה נקודות עניין שעוברות סף זה ומהתבוננות לעומק ניתן לראות שרוב ההתאמות הינן נכונות אף כי יש לא מעט שאינן כאלו.

$Ransac + Homograaphy$



במבט מעמיק ניתן לראות שאחרי הפעלת Ransac כל ההתאמות שנשארו הינן התאמות נכונות בין נקודות העניין בשתי התמונות, מה שמעיד גם על בחירת פרמטרים טובה ב-threshold עבור אלגוריתם ransac.



בהעתקה מימין השתמשתי ב-threshold של 4 פיקסלים באלגוריתם Ransac, דבר שגרם לכך שהיו פחות מ-4 נקודות עניין שהותאמו נכון ולכן הטרנספורמציה כשלה, לעומת המקרה מימין המייצג את ההעתקה שהתקבלה מנקודות העניין שאובחנו בתמונה שהוצגה בדף קודם מימין. (threshold של 8 פיקסלים).

תוצאות האלגוריתם:

בשתי התמונות ניתן לראות כי הטרנספורמציה וה-blending שביצעתי נתנו תוצאות טובות.

חסרונות בתמונה הראשונה: ישנו צל שחור סביב קצוות השיער, נובע מכך שהמסכה באלגוריתם הכילה

חסרונות בתמונה השנייה: ישנו טשטוש בחלק העליון של הטירה וכי הגבעה בצד ימין למטה קצת חורגת מהיכן שהיא אמורה להיות.

מסקנות

בתרגיל זה השתמשתי במספר אלגוריתמים שנלמדו בכיתה על מנת לשלב בין תמונה ברזולוציה גבוהה המכילה את האובייקט המרכזי לתמונה בעלת רקע עם האובייקט המרכזי אך ברזולוציה נמוכה. השתמשתי באלגוריתמים רבים שראינו בכיתה כגון $SIFT, Ransac$ על מנת למצוא טרנספורמציה שתמקם את התמונה בעלת הרזולוציה הגבוהה באופן המתאים לתמונה בעלת הרזולוציה הנמוכה.

תובנה שעלתה לי מהתרגיל היא שישנם מקרים שבו האלגוריתם $pyramid\ blending$ אינו נותן תוצאות טובות לשילוב בין תמונות. ההנחה שלי היא שזה נובע מכך שהתמונות שאיחדנו אינן שילוב בין שתי תמונות שונות אלא הדבקה של אותה התמונה רק ברזולוציה גבוהה יותר. לעומת זאת, שימוש ב-$blending$ נאיבי לפי מסכה בינארית שקצת יותר גדולה מגודל האובייקט אותו רצינו להדביק נותן תוצאות טובות יותר.

בנוסף, בתרגיל למדתי להכיר לעומק את הספרייה $cv2$ ופונקציונליות רבה שלה כגון אובייקטים במחלקה ושימוש באלגוריתמים מורכבים כגון $SIFT$.

לסיכום כפי שניתן לראות, האלגוריתם שמימשתי נתן תוצאות מרשימות העונות על הנדרש בתרגיל.