דו"ח סיכום לתרגיל 1 בראייה ממוחשבת
נושאים: keypoint detector, BRIEF descriptor
תאריך: 21.4.20

מגישים:
אופק חררי 305199879
שני ישראלוב 204679153

הערה: עבדנו ב- google Collaboratory notebooks, השתמשנו רק בקבצים שניתנו עם התרגיל.

Part 1 - Keypoint Detector

1.1 load image

1.2 gaussian pyramid

לפני שמממשים את ה- DoG pyramid, צריך לבנות Gaussian pyramid. באמצעות הפונקציה
createGaussianPyramid אנחנו מפעילים באופן הדרגתי low pass gaussian filter על תמונת הקלט.

שאלה: מה הצורה של מטריצת ה-gaussianPyramid?
הצורה של המטריצה היא: $(\mathrm{L}, \text{image shape}) = (6, 139,89)$, כאשר $\mathrm{L} = \mathrm{len}(\mathrm{levels})$

תוצאות הפירמידה:

1.3 the DoG Pyramid

בחלק הזה בנינו DoG pyramid, כל שלב בפירמידה נבנה על ידי חיסור 2 שלבים מה-gaussian pyramid.
ב-createDoGPyramid מקבלים פירמידה בגודל $(\mathrm{L} - 1, \text{image shape}) = (5,139,89)$

תוצאת פירמידת ה- DoG:

1.4 edge suppression

השתמשנו בשיטה להסרת שפות שמבוססת על שיטת principal curvature ratio בסביבה מקומית של נקודה. הפונקציה computePrincipalCurvature מקבלת את ה-DoGPyramid ומחזירה מטריצה באותו גודל, כלומר, $(\mathrm{L} - 1, \text{image shape}) = (5,139,89)$. כל נקודה במטריצה הזאת מכילה את הערך curvature ratio לנקודה המתאימה במטריצת ה- DoGPyramid.

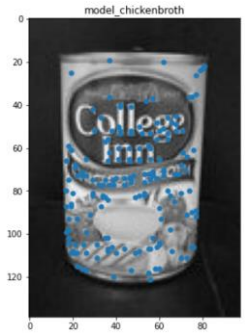
detecting extrema 1.5

כדי לזהות נקודות עניין שהן פינות ו- scale-invariant , ה-DoG detector בוחר נקודות שהן האקסטרימות ב-scale ו-space. אנחנו מתחשבים בשמונת השכנים ב-space ושני שכנים ב-scale (למטה ולמעלה).

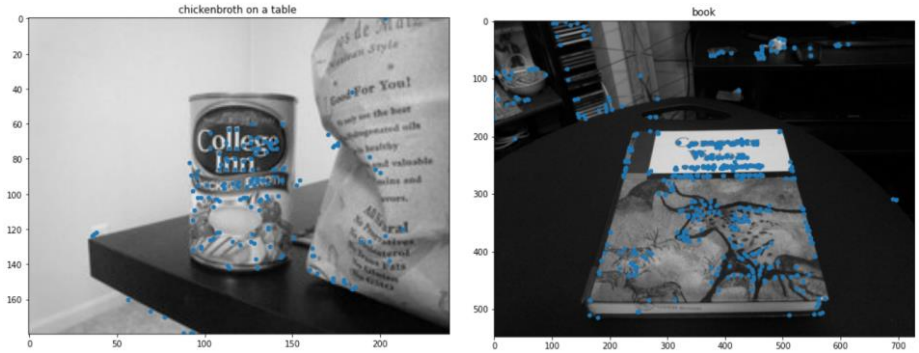
putting it together 1.6

בחלק זה שילבנו הכל כדי לקבל את ה-detector שלנו.

<u>התמונה עם ה-kepoints שזיהינו:</u>



<u>תמונות נוספות עם ה-keypoints:</u>



<u>שאלה: האם אתם מקבלים תוצאות הגיוניות?</u>

קיבלנו תוצאות הגיוניות, DoG רגיש לפינות, שפות ו-blobs ואכן זיהינו גם שפות ו-blobs ולא רק פינות כפי שהיינו רוצים. זאת תוצאה שציפינו לה כי למדנו שזו משימה לא פשוטה.

<u>שאלה: איך ניתן לשפר את התוצאות?</u>

אנחנו השתמשנו לאורך השאלה בפרמטרים הללו: $\theta_c = 0.03,\ \theta_r = 12$. משינויים אפילו מינוריים בערכים הללו מקבלים תוצאות אחרות. באמצעות ה-principal curvature ניסינו להוריד את השפות על ידי הסרת נקודות שמקיימות $R > \theta_r$. ככל שהיינו מורידים את θ_r היינו חסינים יותר מפני נקודות edge. ככל שהיינו מגדילים את θ_c היינו מורידים עוד נקודות אקסטרימום מקומיות. אפשר לבצע אופטימיזציה לבחירת הפרמטרים הללו שיניבו לנו את התוצאות הכי טובות לתמונה מסויימת.

חלק 2 – BRIEF Descriptor

2.1 creating a set of BRIEF tests

בפונקציה makeTestPattern יצרנו זוגות של test pairs. ממאמר [3] קראנו על השיטות השונות ובחרנו לדגום את (x_i, y_i) בצורה רנדומאלית. התוצאות של הפונקציה הזאת עבור patchWidth=9 ו- nbits=256 מופיעות בקובץ testPattern.mat המצורפת בתיקיית ה-code.

2.2 compute the BRIEF Descriptor

בחלק זה חישבנו את ה- BRIEF descriptor לנקודות הרלוונטיות.

2.3 putting it all together

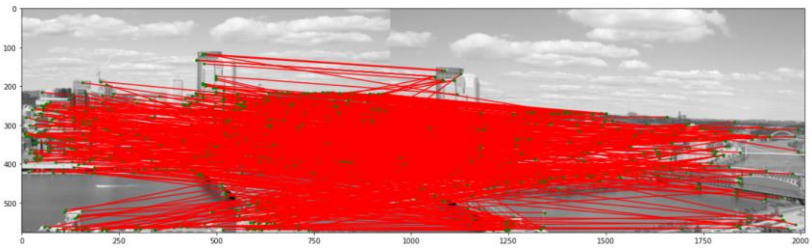
הפונקציה הזאת מבצעת את כל השלבים ההכרחיים למיצוי ה-descriptors.

2.4 Check Point: Descriptor Matching

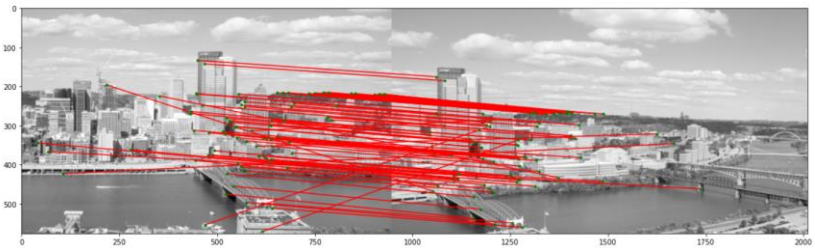
כתבנו testMatch כדי לחשב את ה-feature matches בין שתי תמונות.

השוואות:

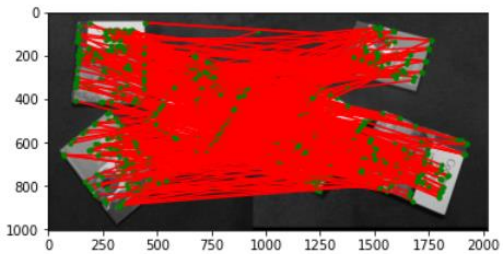
(1) **תמונות ה-incline , נקודות מבט שונות, תוצאה: matches=793**



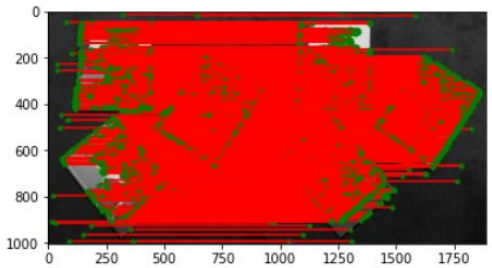
הערה: כשהורדנו ל- ratio=0.4 מ-ratio=0.8 קיבלנו 95 התאמות



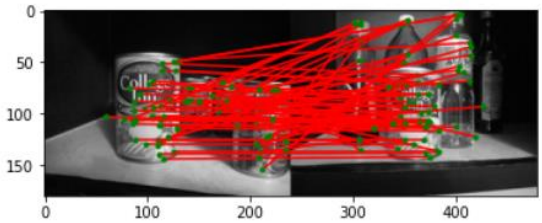
(2) **אותה תמונה מסובבת, תוצאה: matches=561**



(3) **אותה תמונה בדיוק, תוצאה: matches=2885**



(4) **<u>אותו אובייקט במיקומים שונים עם רקעים שונים וגודל שונה, תוצאה: matches=75</u>**



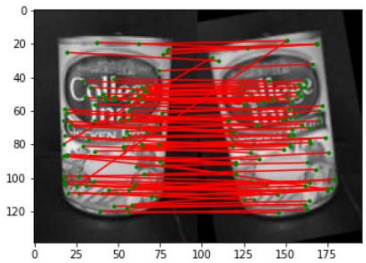
<u>שאלה: אילו מקרים הגיבו טוב יותר ואילו פחות?</u>

ניתן לראות שהמקרה השלישי, אותה תמונה בדיוק, הגיב בצורה הכי חזקה, יש מספר גדול מאוד של התאמות לעומת השאר. זה הגיוני שכן זו אותה תמונה שעוברת את אותו תהליך זיהוי נקודות עניין. התמונה הראשונה והתמונה המסובבת, המקרה השני, הגיבו פחות טוב. התמונה הראשונה הגיבה פחות טוב כי אמנם זו אותה סצנה בשתי התמונות אך מכיוון שיש בכל אחת מהן פריטים שאין בשנייה ציפינו לפחות התאמות. ניתן לראות את ההתאמה בצורה יפה בפינות במגדל הגבוה ביותר. לגבי התמונה השנייה נדון על השפעת הסיבוב בסעיף הבא. אך התמונה שהביאה את מספר ה-matches הנמוך ביותר, זו התמונה במקרה הרביעי שהיא בעצם אותו אובייקט בשתי סצנות שונות לחלוטין גם מבחינת גודל האובייקט, תאורה ורקע שונים.

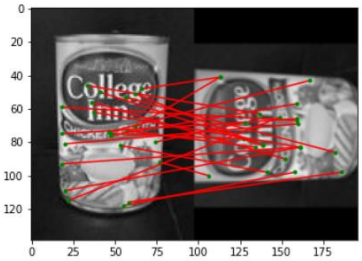
BRIEF and rotations (Bonus) 2.5

<u>דוגמאות</u>

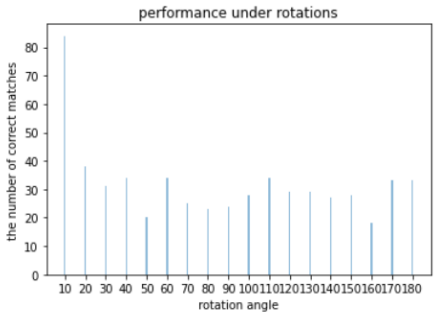
זווית סיבוב 10 מעלות
matches=90



זווית סיבוב 90 מעלות
matches=23



<u>Bar graph:</u>



<u>שאלה: הסבר למה לדעתך ההתנהגות היא כפי שיצא לנו בגרף?</u>

נשים לב שעבור סיבוב של 10 מעלות אנחנו במצב טוב במספר ההתאמות, ולאחר מכן ההבדלים בין הזוויות אינם גדולים. לדעתנו הנקודות דגימה שלנו בתחילת part 2 לא משתנות בין התמונה המסובבת והתמונה הרגילה ולכן יש פחות נקודות התאמה, אם נוכל לדאוג לכך שגם הנקודות דגימה יעברו את ה-rotation אז נוכל לקבל התאמה טובה יותר. מחיפוש באינטרנט ראינו שאכן יש שיטה כזאת הנקראת Oriented Fast and Rotated BRIEF (ORB) descriptor שמבצעת למידה של בחירת הדגימות כדי להוריד את הקורולציה בינהם וכך היא אינווריאנטית לסיבובים.