לק"י

<u>ראייה ממוחשבת, תב1.</u>

<u>חלק 1:</u>

1.2 בהנתן גודל תמונה (H,W,3) ו L=#levels – גודל מטריצת GaussianPyramid הוא (H,W,3,L) , בהנחה שהטשטוש הגאוסיאני לא משנה את מימד התמונה (בדקנו בקוד שזה אכן המצב).

התמונות המתקבלות לאחר טשטוש ב GaussianPyramid :

1.3 בסעיף זה מימשנו DoGPyramid ע"י החסרה של GaussianPyramid צמודים, לפי הביטוי המתמטי הנתון. המוצא הוא ממימד :

```
(len(levels) - 1, shape(im))
```

1.4 בסעיף זה נעזרנו ב cv2.Sobel() ע"מ לחשב נגזרות וכך לחשב את ההסיאן. לאחר חישוב ההסיאן חישבנו את יחסת הקמירות\קעירות בעזרת הנוסחא הנתונה. ב cv2.Sobel() נעזרנו בדגל cv2.CV_64F ע"מ לייצג את הנגזרות ב float , מכיוון שהכנסנו תמונת אפור בטווח [0,1] (ולא uint8!).

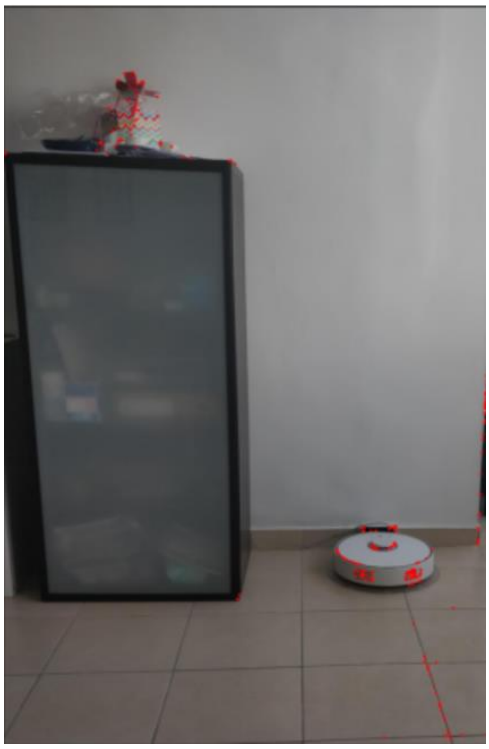
1.5 ע"מ לחלץ את הערכים המתאימים נעזרנו במסיכות בוליאניות, ובמסנן מקסימום\מינימום:

- ע"מ להוציא את המקסימום מעשרת השכנים, נעזרנו בחלון בוליאני בגודל 3X3X3 , כאשר רק 10 האיברים הרלוונטיים הם '1' וזה היה ה footprint לפונקציית ה minimum/maximum_filter של scipy.
- מסיכה שמקבלת True כאשר ה מקסימום\מינימום מהפילטר והערך המקורי זהים (וכך בעצם מוצאים את המיקומים של ערכי ה מקסימום\מינימום).
- מסיכה שמקבלת True כאשר תנאי הקונטרסט מתקיים
- מסיכה שמקבלת True כאשר תנאי ה R מתקיים
- בסוף ייצרנו מסיכה שמקבל True כאשר ארבעת המסיכות מתקבלות True (עם יחס or בין המינימום והמקסימום, ויחס and ביניהן ל-2 תנאי הסף), וכך אנו מוצאים רק פיקסלים שמקיימים את כל התנאים.
- נחזיר את המיקומים של הפיקסלים שהם True (לאחר סידור עמודות x-y).

המשך ->

1.6 הdetection שקיבלנו:

הבית של תומר (בידוד וזה):



התוצאות שקיבלנו יחסית בסדר אך עדיין מזהות לעתים שפות (שהיינו רוצים להמנע מהן). מכיוון שאלו תמונות טבעיות ולכן קיים להן רעש טבעי, סביר להניח שהחלקה עם סיגמא גדולה יותר \ מספר רב יותר של levels היינו מקבלים תוצאות טובות יותר (כמובן שאם היינו מגדילים את הרמות היינו משלמים בסיבוכיות חישוב).

חלק 2:

2.1 בחרנו את האופציה הראשונה מהמאמר : $x, y \sim Uniform\left(-\frac{S}{2}, \frac{S}{2}\right)$. לאחר מכן לצורך נוחות השימוש נירמלנו את הערכים לגודל החלון הרצוי (כך ש 0,0 נמצא בקצה העליון שמאלי כמקובל ו80 בקצה התחתון לפי סדר שורה) ע"י משחקי אינדקסים.

לבסוף שמרנו כנדרש ב :

`testPattern.mat`

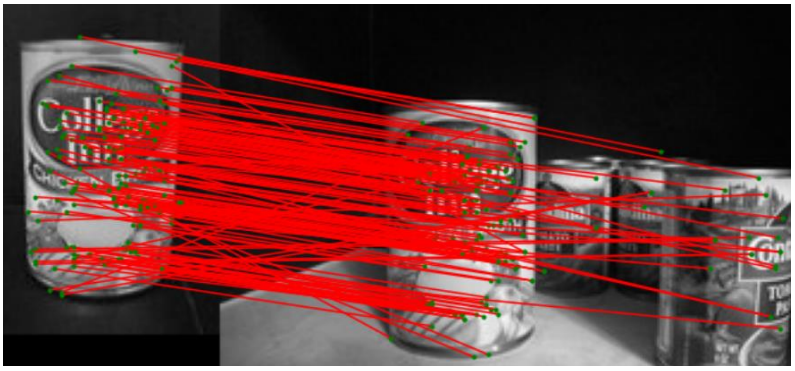
שמצורף בתיקיית הקוד.

2.2 בפונקציה זו, ראשית ניקינו את נקודות העניין שהתקבלו מחלק1 אשר נמצאות בקצה התמונה, כך שלא נכנס חלון שלם סביבן. שנית, חילצנו את ערכי התמונה לפי האינדקסים (המנורמלים לגודל חלון כך ש0,0 בקצה שמאלי עליון ו 80 בקצה התחתון לפי סדר שורה) הוקטורים ששמרנו מסעיף 2.1 . לבסוף שמרנו את ערכי ה דסקריפטור ע"י הנוסחא הנתונה :

$$\rho(p; x, y) := \begin{cases} 1, & \text{if } p(x) < p(y) \\ 0, & \text{otherwise.} \end{cases}$$

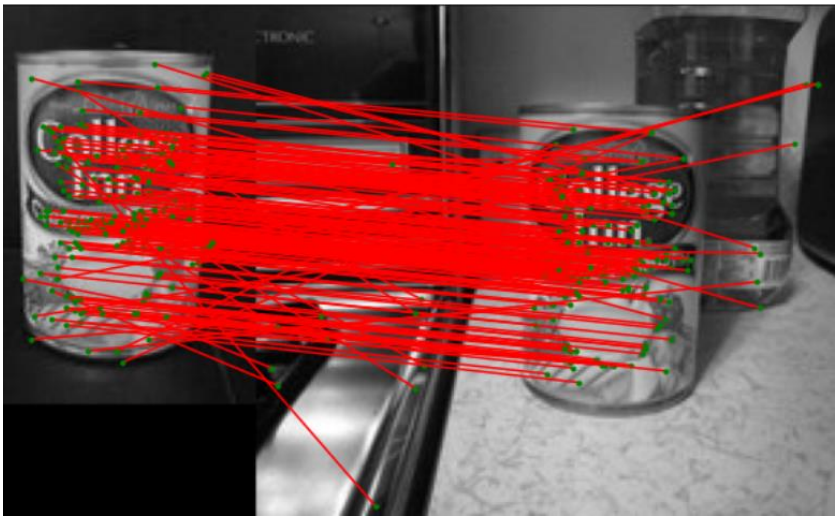
2.3 בפונקציה זו שילבנו את הוצאת הזיהויים מחלק 1 , ותיאורם בעזרת $BRIEF$. נעזרנו בוקטורים X, Y ששמרנו בסעיף הראשון ע"י טעינה של הקובץ המתאים.

2.4 תוצאות מעניינות (בכולם עם $ratio = 0.8$ לפי ההנחיות, אלא אם כן פירטנו אחרת) :

- 

כאן ניתן לראות שחלק ניכר מנקודות העניין מתאימות – סביר להניח מכיוון שזווית הצילום יחסית זהה ולכן הסביבה של נקודות העניין דומה, אך יש מעט נקודות עניין עם התאמה לא מספיק טובה – בעיקר בשפות של פחית השימורים (כמצופה).
הסיבה לכך שנקודות השפה עם התאמה לא נכונה נובעת לדעתינו מסביבה יחסית דומה, ומכיוון ש $BRIEF$ עובד על סביבה ההתאמה נופלת במקרה הזה (בהתאם למה שלמדנו).
בעיקר ניתן לראות טעויות בהתאמה של שפות הפחית המקורית לשפות בפחיות שונות – בעיקר מכיוון שהרקע יחסית זהה.

- תמונה נוספת מעניינת

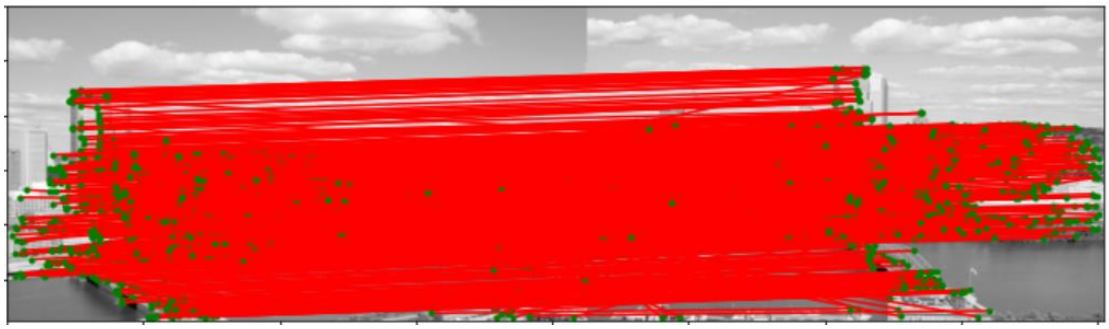


ואילו עם יחס 0.7 :

גם כאן יש נקודות עניין רבות זהות עם מספר קטן של נקודות לא תואמות. גם כאן לדעתינו הסיבה היא סביבה יחסית זהה של נקודות עניין שונות ואופן פעולת *BRIEF* (שעובד על סביבה, ולכן שפות\נקודות עניין עם סביבות זהות יותאמו בתהליך ההתאמה). בנוסף, קיימת התאמה גבוהה בין הנקודות שנמצאות על הפחית (ולהפך עבור נקודות על הרקע), שכן הן לא מושפעות מהרקע שמאחורי הפחית ומקבלות רק את הרקע של הפחית, ולכן הסביבה המושווית ע"י *BRIEF* תהיה יחסית זהה ב-2 המקרים, כתלות בזווית הצילום. כאן זווית הצילום זהה ולכן יש התאמה גבוהה.

המשך ←

- עבור $incline$:

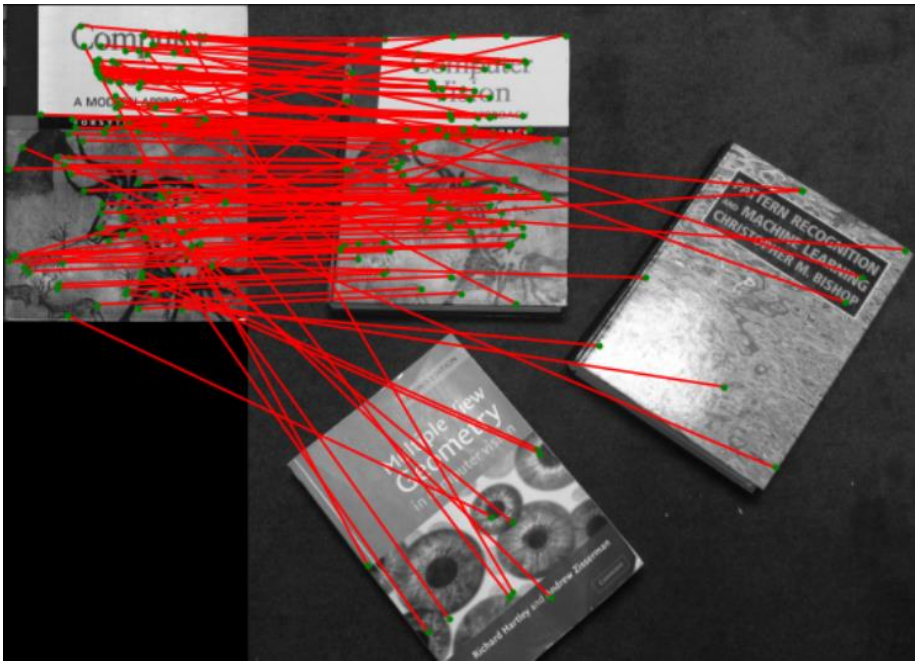


קשה לזהות עם יחס זה, לכן נבדוק עם יחס 0.3 :

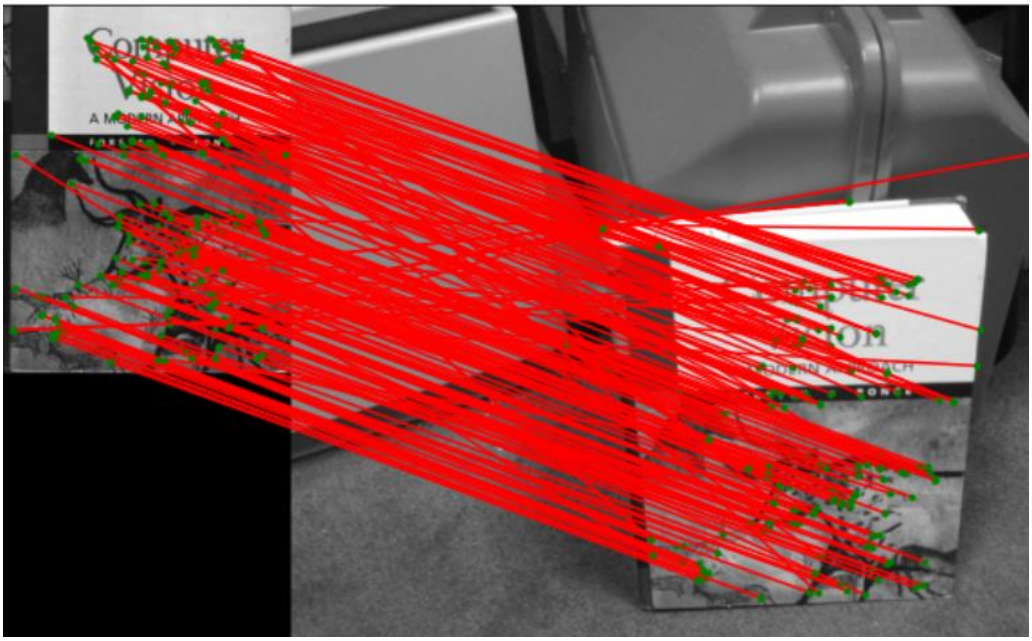
ניתן לראות התאמה יחסית טובה. בקווים שניתן לעקוב אחריהם – ניתן לראות שהפיצ'רים זהים ברוב המקרים. מכיוון שכאן יש סיבוב אופקי של התמונה – הסביבה של כל נקודת עניין נשארת זהה ולכן ההתאמה טובה מאד.

- עבור ספר ה CV , נעבוד עם יחס 0.5 (כי כך ניתן לראות יחסית טוב את התוצאות) :

כאן ניתן לראות בבירור שככל שהאובייקט מקבל עיוות גדול יותר מבחינת הזווית – כך התאמת הנקודות גרועה יותר. למשל: התאמת נקודות העניין שבנמצאים בכותרת של הספר ממש לא טובה, מכיוון שהכותרת בתמונה השמאלית קיבלה עיוות גדול יחסית מכיוון שיחסית רחוקה מנקודת הצילום, ואילו התאמת הנקודות בחלק התחתון של הספר מתבצעת בצורה טובה יותר. תוצאה זו הגיונית, מכיוון ש $BRIEF$ עובד על התאמה של חלון סימטרי (לפחות במקרה שלנו), ולכן עיוות לא סימטרי של התמונה לא יעבוד בצורה מספקת. למשל עם נסתכל על הנקודה שיוצאת מהאות m בצד השמאלי, סביר להניח שיש לה רצף יחסית גדול של פיקסלים שנמצאים אנכית מעליה (באותה עמודה) עם אותו ערך, ואילו בחלק הימני מספר הפיקסלים שנמצאים מעל נקודת העניין בעלי אותו ערך יהיה קטן במקרה הזה בגלל עיוות זווית הצילום. מכיוון ש $BRIEF$ בודק מרחק $hamming$, במקרה הזה המרחק ישפיע.



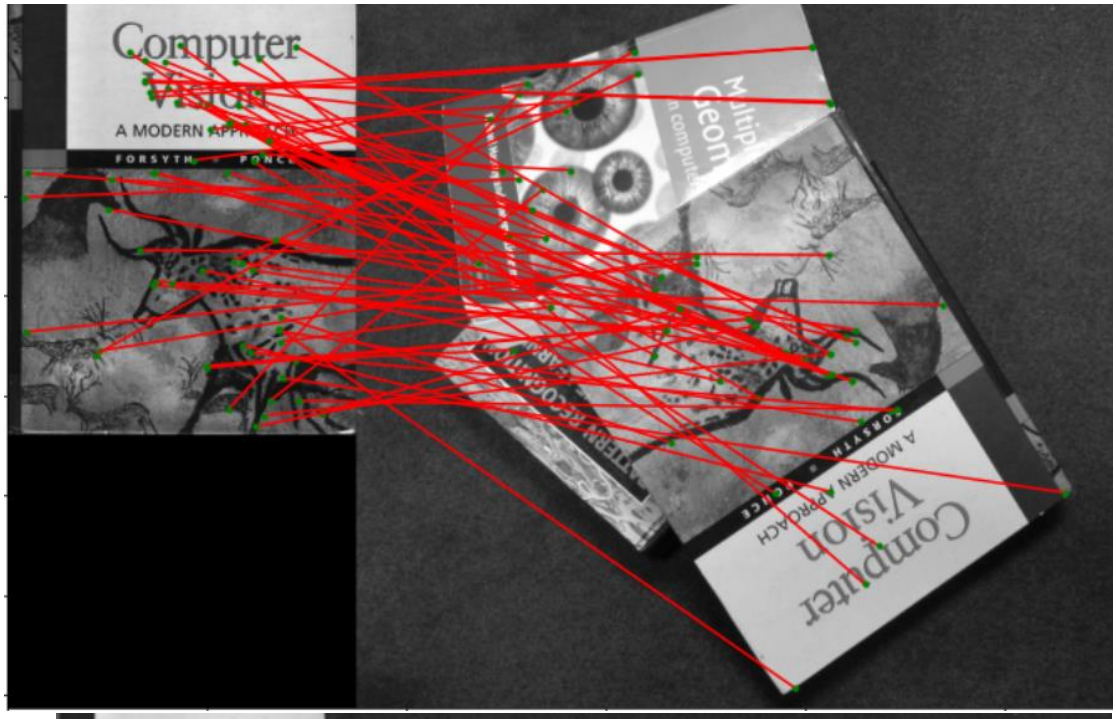
כאן זווית הצילום יחסית זהה, אם כי הספר מימין מקבל סיבוב קטן. לכן התוצאה הכללית היא טובה, גם באיזור כותרת הספר, עם נקודות אנומליה בודדות. לדעתנו תוצאה זו מניחה את הדעת, ומסתדרת עם ההיגיון של אופן העבודה של $BRIEF$, שכן העיוות הלא סימטרי מהמקור לא גדול מדי.

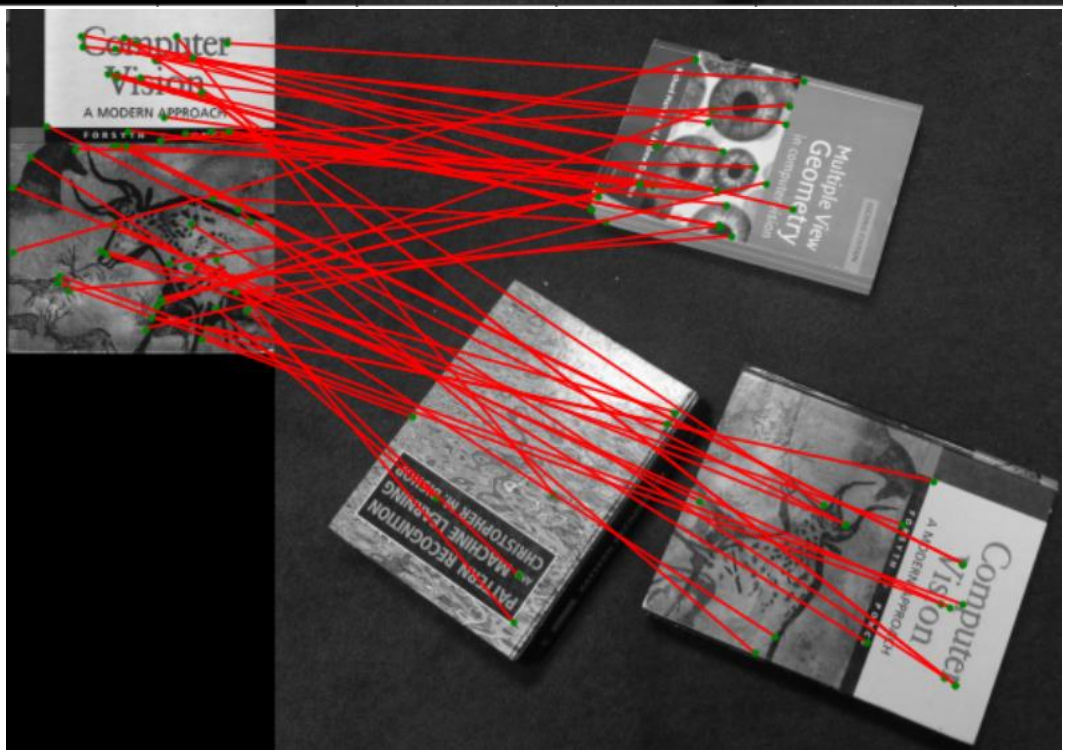


כאן זווית הצילום יחסית זהה, ולכן ניתן לראות התאמה גבוהה בין הנקודות עם מעט טעויות. התוצאה הגיונית , מכיוון שזווית הצילום יחסית זהה ולכן רוב נקודות העניין שנמצאות בתוך הכריכה, מקבלות סביבה זהה (רקע של הכריכה), ולכן יש התאמה גבוהה.

המשך ←

עבור 2 התמונות הבאות יש אותו הסבר :



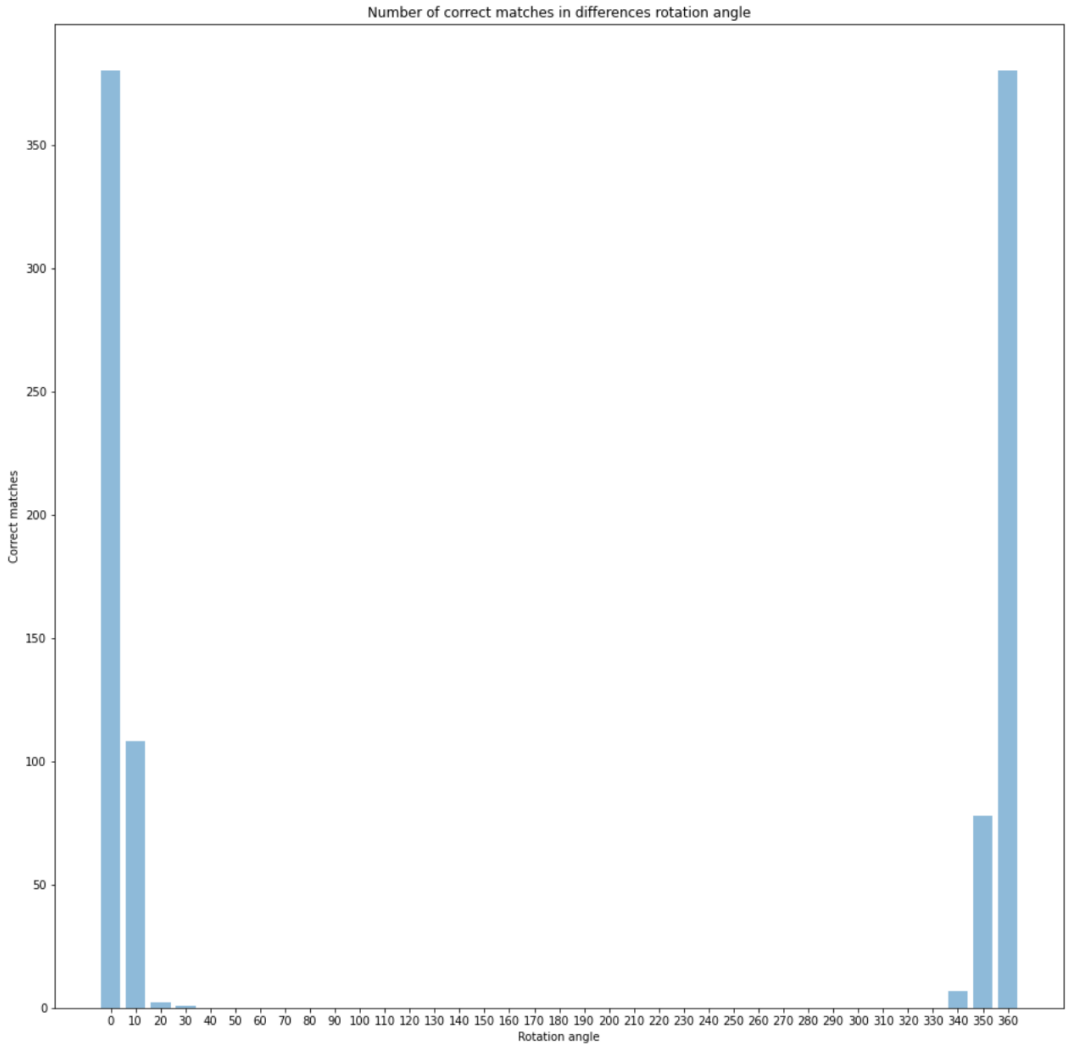


כצפוי, כאן התוצאה היא חד משמעית לא טובה (אין התאמה כלל) שכן העיוות מהמקור לא סימטרי לחלוטין (סיבוב). כאן כל נקודת עניין תקבל סביבה שונה לגמרי בגלל הסיבוב (בגלל הסיבות שהסברנו בדוגמאות הקודמות) ולכן מרחק ה *hamming* יהיה גדול – מה שיגרום התאמה נמוכה. בנוסף ניתן לראות התאמה מוטעית בין נקודות שונות בעלי רקעים דומים.

המשך ←

בונוס:

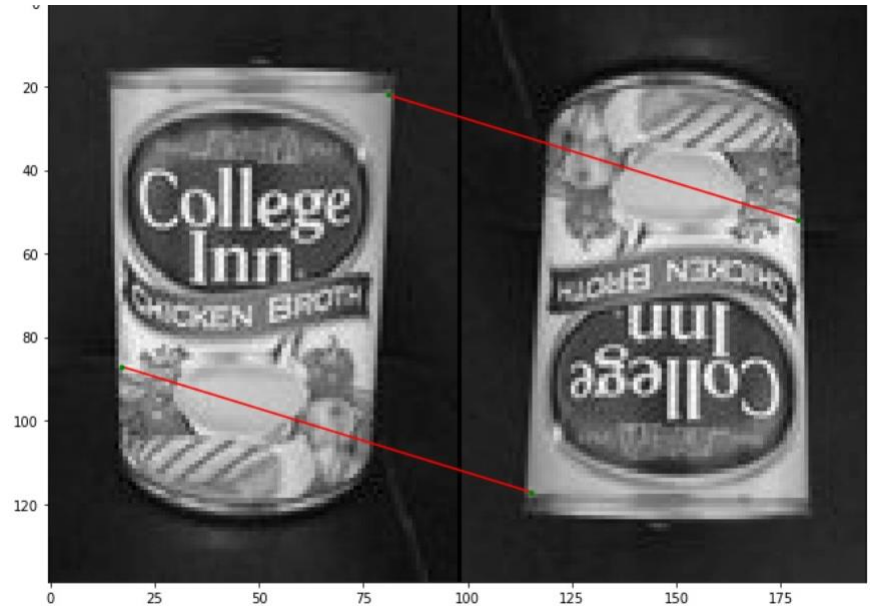
כדי לגלות אם הזיהוי שלנו נכון ביצענו עבור זוג נקודות שהאלגוריתם מצא כהתאמה את השלבים הבאים:
יצרנו תמונה עם אפסים וערך יחיד בעל הערך "1" בנקודה החשודה מתמונת המקור.
ביצענו סיבוב על התמונה.
בדקנו עם בסביבת הנקודה החשודה כהתאמה מתמונת הסיבוב יש עכשיו ערכים שונים מאפס.
אם נמצאו ערכים כאלה אז קבענו כי ההתאמה נכונה.



ניתן לראות ירידה חדה במספר ההתאמות יחד עם הגדלת הזווית. כבר אחרי סיבוב ב20 מעלות אנו יורדים למספר התאמות בודדות.
כמו שהסברנו בסעיפים הקודמים, תוצאה זו הגיונית, מכיוון ש $BRIEF$ עובד על התאמה של חלון סימטרי (לפחות במקרה שלנו), ולכן עיוות לא סימטרי של התמונה (סיבוב הוא עיוות לא סימטרי כי הוא משנה הפיקסלים בחלון ואת מיקומם) לא יעבוד בצורה מספקת.

המשך ←

דוגמא להתאמה שגוייה שממכישה זאת:



0

אלו שתי התאמות שהאלגוריתם מצא שקבענו כשגויות. ניתן לראות שבתמונת המקור בפינה הימנית הוא מזהה שלושה גוונים, שחור מימין, אפור כהה מלמעלה ואפור בהיר מלמטה. ניתן לראות בתמונה המסובבת שהוא מוצא זאת במקום אחר. אפשר גם להבחין שהאלגוריתם מצא את אותם נקודות גם כשהן מסובבות.