**תרגיל בית 5 בעיבוד תמונה – זיהוי מקשים במקלדות שונות בעזרת sift**

**מגיש: טל גורודצקי, 211898739**

**מבוא:**

**פרויקט זה מקבל שני תמונות של מקלדות, אחת של windows ואחת של mac. לאחר עיבוד התמונות השונות, אנו נעזרים בראייה ממוחשבת (sift algorithm) כדי לזהות כמה שיותר התאמות של מקשים במקלדות השונות.**

**תיאור השלבים:**

1. **עיבוד תמונה בסיסי של 2 המקלדות – התמונות של המקלדות הינן צבעוניות, ולכן בניתי את הפונקציה load\_images. פונקציה זו טוענת את התמונות במוד grayscale, ובאמצעות ביצוע threshold התמונות נהיות בינאריות (שחור לבן). הפונקציה מחזירה את התמונות הבינאריות.**
2. **ציור הקונטורים בתמונות – באמצעות הפונקציה draw\_contours, אשר מקבלת כפרמטרים 2 תמונות (המקלדות שלנו לאחר העיבוד) וצובעת אותם אם הם בינאריות. לאחר מכן, משתמשים בפונקציה find\_contours על מנת למצוא את כל הקונטורים בתמונות. מכיוון שהתמונה עשויה להכיל קונטורים רבים שחלקם לא רלוונטים (רוצים למצוא רק את המקשים), בוצע סינון על הקונטורים, ורק הקונטורים ששטחם הוא בין 1000 ל40000 נבחרו להיות קונטורים שמועמדים להיות מקשים במקלדת. לאחר סינון זה, צובעים את הקונטורים במסגרת ירוקה. הפלט של הפונקציה הזו הוא גרף אשר מציג את 2 המקלדות, עם מסגרת ירוקה על כל אחד מהמקשים.**
3. **מציאת המקשים בכל אחת מהמקלדות – באופן דומה לפונקציה שתוארה בסעיף 2, כתבתי פונקציה שנקראת find\_pieces אשר מקבלת 2 תמונות של מקלדות ומוצאת conturs לפי המגבלות שתוארו לעיל. מה ששונה בפונקציה זו זה שהיא מחזירה 4 רשימות – keys1 (כל המקשים שאותרו במקלדת הראשונה), keys2 (כל המקשים שאותרו במקלדת השנייה) ועוד 2 רשימות contours (אחת לכל מקלדת).**
4. **מציאת contours בכל מקש – הרעיון נבע מכך שהתאמת המקשים לא בוצעה באופן מיטבי בין מקש למקש אחר, לאור העובדה שכמעט בכל מקש יש יותר מתו אחד, מה שגורם לsift להוריד את הניקוד למקשים שעשויים להתאים. כדי לפתור בעיה זו, בניתי פונקציה בשם process\_piece\_contours אשר מקבלת תמונה של מקש ופרמטר מספרי. הפונקציה מחשבת בתוך המקש conturs ומסננת את הcontours הרלוונטים ע"פ שטח – חסם תחתון הינו הגודל של המקש לחלק לפרמטר המספרי, חסם עליון הינו הגודל של המקש פחות שגיאה כלשהי. הפונקציה מחזירה רשימה של מילונים עם פרמטרים הרלוונטים לכל קונטור.**
5. **סיווג המקשים ע"פ מספר הקונטורים שנמצאו בכל מקש – מבוסס על ההבנה שבסבירות גבוהה מקש אחד יתאים למקש אחר במקלדת השנייה אם יש לשניהם את אותה כמות קונטורים במקש. לשם כך, בניתי פונקציה בשם classify\_keys\_by\_contours\_count אשר מקבלת מילון של קונטורים בתוך מקש ומחזירה מילון המסדר את האינדקסים של המקשים לפי כמות הקונטורים שאותרו בכל מקש.**
6. **לאחר ביצוע כל הפעולות הללו, בניתי פונקציה match\_between\_keys אשר מקבלת את הרשימות keys1,keys2 ואת רשימות הקונטורים שלהם, כמו כן היא מקבלת את שני המילונים אשר מסווגים כל מקש במקלדת ע"פ כמות הקונטורים שאותרו בו.  
   בפונקציה זו אנו מממשים את האלגוריתם של sift – בעצם עוברים על כל אחד מהמקשים על פי כמות הקונטורים שיש לו, ואז משווים עם המקשים במקלדת השנייה, בהנחה שיש למקש הרצוי את אותו כמות קונטורים. האלגוריתם נעזר בknn, אשר מעניק להתאמה ניקוד מסוים, ואנו מסננים את ההתאמות על פי יחס של 0.7. אנו נבחר את ההתאמה הטובה ביותר של המקש הנוכחי, ונציג אותה למשתמש.**
7. **תוצאות – בפועל אותרו כ-30 מקשים מתוך 76 מקשים שנמצאים במקלדת mac. יש לציין שרוב המקשים שאותרו הם מספרים ואותיות (כמעט כולם) בציון טוב יחסית. ישנם מקשים שהיה ידוע שלא אצליח לאתר, בין היתר בגלל השוני במקלדות (מקש שקיים במקלדת אחת אך לא באחרת, שוני באופן כתיבת המקש [ctrl לעומת control]). כמו כן, מקשים בודדים לא אותרו בגלל ציון נמוך יותר בהתאמה, אבל הם כן היו בטופ 5 התאמות.**
8. **בעת הרצת הקוד, ניתן יהיה לראות את ההתאמות השונות שהקוד הצליח לאתר. לנוחות הבודק, נכתבה פונקציה בשם display\_matches\_with\_keyboard אשר מציגה את ההתאמה במרכז המסך, ואת המקלדות לידה, כאשר המקש המותאם מסומן בירוק ב-2 המקלדות.**