פרויקט סוף קורס-עיבוד תמונה

שם: מיכאל בן עמוס

ת.ז: 203862263

בפרויקט זה קיבלנו 4 תמונות של כפות ידיים בצורה קבועה, אני קיבלתי תמונות של החלק החיצוני של כף יד שמאלית.

חלק ראשון:

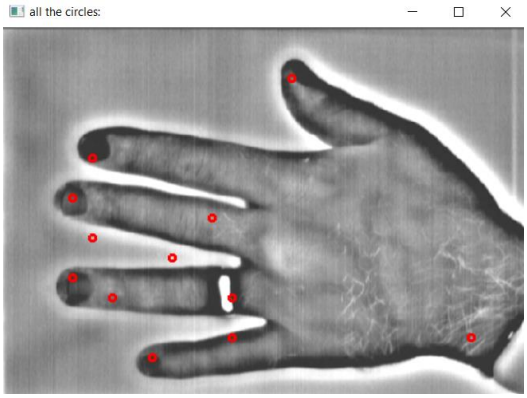
בחלק זה התבקשנו למצוא את קצות האצבעות בכל יד.

אופן פעולת התכנית שלי הוא כזה, רצתי בלולאה על כל התמונות מן התיקייה ובכל תמונה עשיתי את השלבים הבאים:

1. חיתוך התמונה ומציאת מסגרת בעזרת הפונקציות Canny ו- bilateralFilter של הספריה openCV. בגלל החלק הזה נדרש שהתמונות יהיו איכותיות, אחרת הגזירה שלהן לא תביא תוצאה מספיק טובה על מנת שהזיהוי יעבוד.

2. לאחר מציאת המסגרת בצורה הכי טובה שניתן, שלחתי את התוצאות לפונקציה finger_detetction (שורה 34). בפונקציה הזאת אני מנסה לזהות את האצבעות לפי האלגוריתם hough_circle שלמדנו במהלך הסמסטר, אלגוריתם שמזהה מעגלים בתמונות, במטרה לתפוס את חמשת האצבעות.

בגלל שעדיין יש רעשים בתמונות, אם אחפש את 5 המעגלים הטובים ביותר לא אקבל את כל האצבעות ולכן חשבתי לחפש את כל המעגלים ולנפות את המיותרים. בתמונה ניתן לראות את כל המעגלים שהפונקציה זיהתה על אחת הידיים:



כמו שניתן לראות, הפונקציה זיהתה את כל האצבעות אך ישנן נקודות מיותרות בשלבים הבאים אני אסביר איך ניפיתי את הנקודות האלה.

3. על מנת לנפות את הנקודות המיותרות, בניתי מספר פונקציות ניפוי שכל אחת מהן בנפרד כנראה לא תהיה מספיק טובה אבל כולן ביחד נותנות תוצאה טובה, כל הפונקציות עובדות מול התמונה החתוכה אחרי Canny ו-bilateralFilter פירוט על הפונקציות:

   א. Remove_the_rest:
   בפונקציה התבססתי על כך שהנקודות שקרובות לקצות האצבעות נמצאות באזורים בתמונה (מסגרת התמונה) עם יותר "צבע" בגלל שהן קרובות גם לקצה האצבע וגם לחלק העליון והתחתון של האצבע, בניגוד לנקודות שנמצאות במרכז האצבע/ מרכז הכף יד/ מחוץ לכף יד. לכן יצרתי פילטר בגודל 25X25 שפשוט עושה ממוצע של ערכי הפיקסלים בגודל הפילטר, ואז מיינתי את הנקודות שקיבלו את התוצאות הגבוהות ביותר, והשארתי את 12 הנקודות שקיבלו את הערכים הגבוהים ביותר.

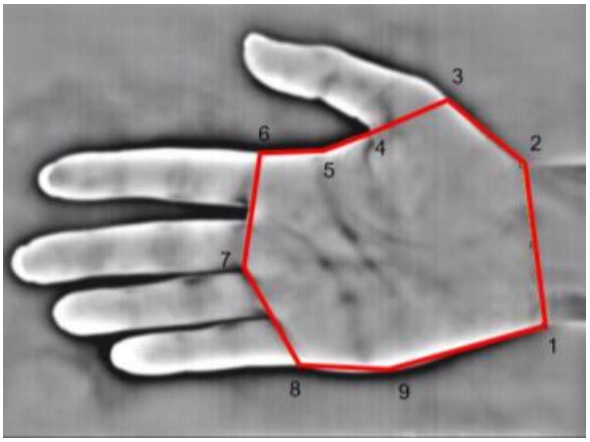
   ב. Remove_the_rest2:
   בפונקציה הזאת עברתי על הנקודות שנשארו ופשוט "טיילתי" על התמונה מהנקודה שמאלה עד שנתקלתי בפיקסל שאינו שחור (שונה מ-0), עבור מספר מסוים של פיקסלים שלא נמצאה נקודה כזאת, זה מבחינתי היה סימן שהנקודה אינה תקינה, כיוון שהנקודה בקצה האצבע מאוד קרובה לקו לבן, אם זזים מספיק שמאלה.
   לאחר מכן הפעלתי את הפונקציה הזאת שוב, עם פרמטרים קצת יותר חלשים של Canny, ואיפסתי חלקים מהתמונה שבהם יש אלמנטים לא רלוונטים לזיהוי האצבעות.

   ג. Remove_the_rest3:
   פה כבר לרוב לא נותרו נקודות מיותרות, פונקציה זאת היא בעצם "וידוא הריגה" עבור תמונות שאינן מספיק איכותיות, או שיש הפרעות לתמונה כגון טבעות/שעונים וכו'.
   מה שעשיתי היה ללכת לנקודה הגבוהה ביותר, שהיא במקרה של התמונות שלי האגודל ולמתוח קווים לכיוון שאר האצבעות, והנקודה הקרובה ביותר לקצה הקו שהוצאתי, היא הנקודה שנשארת.

לאחר כל השלבים הללו ציירתי את 5 הנקודות שנשארו על התמונה המקורית.

חלק שני:

בחלק זה התבקשנו למצוא את מתאר כף היד על פי 9 הנקודות הנ"ל

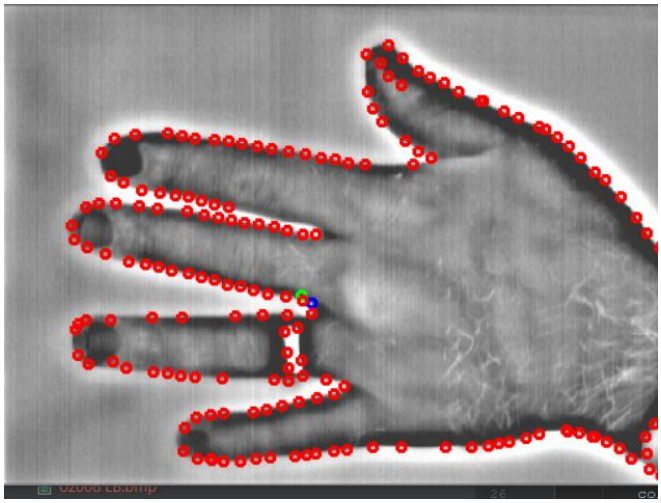


חשוב לציין שאת הפתרון שלי ביססתי על החלק ראשון של הפרויקט ונעזרתי במיקומי האצבעות כדי לאתר את נקודה 7.

הסבר על הפתרון:

הפתרון מתבסס על הפונקציה goodFeaturesToTrack של openCV פונקציה אשר מוצאת נקודות משמעויות בתמונה, שדרוג מסוים לאלגוריתם Harris Corner Detection של הספריה.

דוגמא לתוצאה של הפונקציה, כאשר הנקודות האדומות אלו הנקודות שהפונקציה מצאה, כמו בחלק הקודם אני רוצה להשאיר את 9 הנקודות הרצויות ולנפות את שאר הנקודות.



פונקציות עזר שבניתי:

Distance- מוצאת מרחק בין 2 נקודות

Line- פונקציה אשר מקבלת נקודת התחלה, זווית במעלות ומרחק ומחזירה את הנקודה שנמצאת בסוף הקו, כאשר הזווית ביחס לציר X.

find_closest_point- פונקציה שמקבלת מערך של נקודות (כל הנקודות האדומות) ונקודה נוספת (הנקודה שתתקבל מהפונקציה line) , ומוצאת מתוך קבוצת הנקודות את הנקודה הקרובה ביותר.

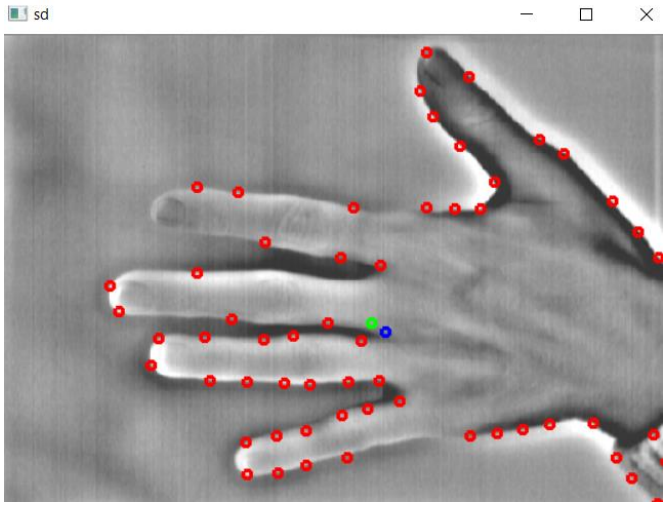
מציאת הנקודות בעזרת הפונקציות הנ"ל:

נקודה 7:

זה היה השלב המאתגר ביותר, וזאת הנקודה הכי קריטית עבור הפתרון שלי, ברגע שהיא תמצא בצורה מדויקת שאר הנקודות ימצאו גם הן.

על מנת למצוא אותה השתמשתי בנקודות מהחלק הראשון, לקחת את הנקודה של קצה האצבע הרביעית (קמיצה), ומתחתי ממנה קו בזווית של בערך 10 מעלות ובאורך של בערך 160 פיקסלים. ובחרתי את הנקודה הקרובה ביותר לקצה הקו.

כאן השתמשתי בפונקציה goodFeaturesToTrack עם פרמטרים שונים על מנת לקבל פחות נקודות, ניתן לראות בתמונה את כל הנקודות שהפונקציה מצאה (באדום), ואת הנקודה בקצה הקו הישר (בירוק), והנקודה האדומה הקרובה אליה ביותר שנבחרה (בכחול)



הוספתי כמה מצבים שמצביעים על כך שהפתרון של החלק הראשון לא היה מדויק ובמקרים נדירים אלה, פשוט בחרתי את הנקודה הכי קרובה למרכז התחתון של התמונה, זה כמובן לא יהיה מדויק מאוד, אבל יתן פתרון סביר בהתאם לנסיבות.

מציאת שאר הנקודות:

ברגע שמצאתי את נקודה 7 בהצלחה, למצוא את שאר הנקודות זה יותר פשוט, כל הידיים בעולם בנויות באותה צורה וברגע שהפונקציה goodFeaturesToTrack מצאה את נקודות העניין רק נשאר לאתר אותן.

בעזרת הפונקציות שבניתי, מתחתי קווים מנקודה לנקודה בזוויות המתאימות על מנת למצוא את הנקודות הקרובות ביותר שהחזירה הפונקציה goodFeaturesToTrack.

לצורך העניין כדי למצוא את נקודה 6 כל מה שהיית צריך זה למתוח קו בזווית של כ-70 מעלות ובאורך של כ-100 מהנקודה 7 ולבחור את הנקודה הקרובה ביותר אליו.

במקרים שהנקודה הקרובה ביותר (משתנה לכל נקודה), הייתה במרחק גדול מדי מן הנקודה המוערכת בסוף הקו, לקחתי את הנקודה שהייתה בסוף הקו. במקרה קיצון כאשר הפונקציה goodFeaturesToTrack לא מצאה נקודות כראוי, אז הדרך הזאת יכולה לתת תוצאה רעה, ומצאתי שהאילוץ הזה משפר את דיוק הפתרון באופן כללי.

דוגמא לפתרון סופי:

פרטים טכניים:

כל הנקודות של כל תמונה מ-2 החלקים נכתבות לקובץ אקסל נפרד, בשם:

part1_image#index.xlsx

part2_image#index.xlsx

כאשר index# זהו מספר התמונה המתחיל מ-0 עד מספר התמונות שיש בתיקייה.

כמו שאמרתי, החלק השני מתבסס על החלק הראשון, וכדי לחסוך בזמן ריצה, את הנתונים של החלק השני שאבתי מהקובץ אקסל של החלק הראשון,

לכן כדי שהתכנית של החלק השני תעבוד כנדרש, חייב להריץ את קודם את החלק הראשון, על מנת שקבציי האקסל יוצרו כנדרש, והתכנית תוכל לקרוא את הנקודות.

דבר נוסף:

עקב בעיות טכניות, לא הצלחתי לקרוא את התמונות מהתיקייה images לכן התמונות צריכות להיות בתיקייה הראשית של הפרויקט.