המחלקה להנדסת
חשמל ומחשבים
אוניברסיטת
בן-גוריון בנגב

דו"ח מסכם פרויקט בקורס מבוא לעיבוד ספרתי של תמונות 361.1.4751

שם הפרויקט :

Lego Land

מגישים:

1. זיו חדד
2. עוז חג'בי
3. ערן דוטיש
4. תומאס מנדלסון

תאריך הגשה:

יום חמישי 09 מאי 2024

תוכן עניינים

הקדמה 3
תקציר הפרויקט 3
מטרת הפרויקט 3
תיאור המערכת 4
שלב התכנון והעבודה על הפרויקט 5
מיקום המצלמות וייעודן 5
אתגרי הפרויקט 5
אילוצי הפרויקט 6
הנחות הפרויקט 6
תיאור אלגוריתם - זיהוי חתיכות הלגו 7
הסבר על האלגוריתם 7
קלט, פלט ופרמטרים 8
תרשים זרימה 9
תוצאות ביניים 9
תיאור אלגוריתם – זיהוי ההרכבה 11
הסבר על האלגוריתם 11
קלט, פלט ופרמטרים 12
תרשים זרימה 13
תוצאות ביניים 13
הסבר על פונקציות משמעותיות 15
אנליזה של אחד החלקים 17
סיכום 21
ביבליוגרפיה 22
סרטון 22
קישור לסרטון 22

הקדמה

תקציר הפרויקט

הפרויקט מדמה מערכת המלווה את המשתמש במשחק הלגו על משטח מוגדר, המערכת בנויה משני שלבים.

בשלב הראשון, המשתמש מפזר חתיכות לגו על המשטח, המערכת מזהה את החלקים ומקטלגת אותם עפ"י צבע וצורה.

למשתמש מוצגת רשימת החלקים המתעדכנת בזמן אמת מהמצלמה.

בשלב השני, המשתמש מקבל רשימה של דוגמאות הניתנות להרכבה מהחלקים שברשימה.

המשתמש בוחר את אחת הדוגמאות ומוצגת לו תמונה כיצד לבנות, באישור המשתמש מופעל שעון עצר למדידת זמן הבנייה של הדוגמא אשר נעצר ברגע בו המערכת זיהתה כי המשתמש בנה את הדוגמא בצורה נכונה.

מטרת הפרויקט

בפרויקט זה השתמשנו בכלים הנלמדים לקורס ליצירת משחק אינטראקטיבי המבוסס על עיבוד תמונה בזמן אמת.

הפרויקט מאפשר למשתמש יחיד לבצע הרכבות שונות מחלקי הלגו שונים אשר מחזיק.

שימוש לדוגמא הוא מתן האפשרות לילד לשחק בלגו למול אתגרי הרכבה משתנים, מחקרים מראים שמשחק בלגו משפר את ההתפתחות הקוגניטיבית של ילדים בגילאי 4-5 [2][1].

תיאור המערכת

המערכת מורכבת מממשק משתמש גרפי שבאמצעותו המשתמש מנווט בין השלבים השונים.

למערכת מחוברת 2 מצלמות שמשמות כאמצעי הקלט של הבנייה מול המשתמש.

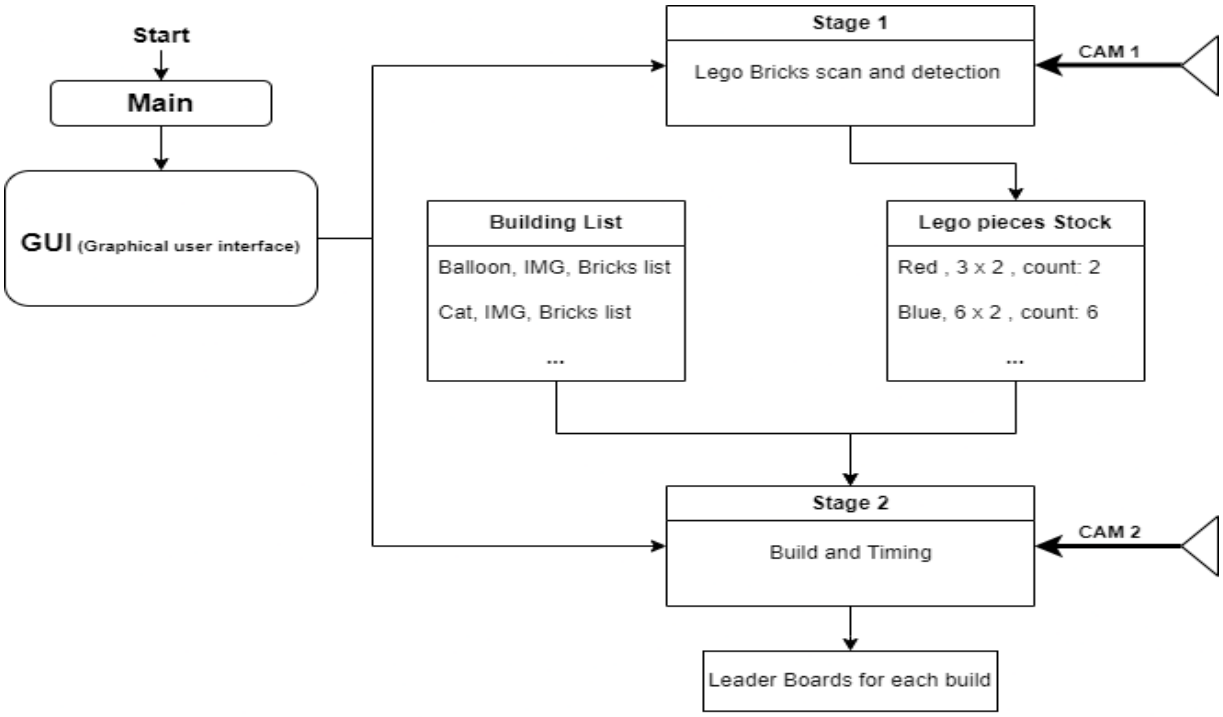
בנוסף הקמנו 2 מאגרי מידע שמשמשים לטובת ניהול המידע: מאגר מידע של סוגי החלקים השונים (כל חלק מאופיין באורך, רוחב וצבע) ומאגר מידע של הדוגמאות להרכבה (כל דוגמא מאופיינת בתמונות ההרכבה, שם ורשימת החלקים הנדרשים לבנייתה.

בשלב הסריקה לכל פריים "מעניין" מתבצע עיבוד הכולל חלוקה לסוגי חלקים וסימונם במסך (החלוקה מתבצעת על בסיס אלגוריתם עליו נפרט בהמשך).

בשלב הבנייה לכל פריים "מעניין" מתבצע זיהוי של משטח ההרכבה.
מהפריים אנחנו מוצאים את משטח ההרכבה, מבצעים עליו מניפולציות מרחביות לקבלת המשטח בגודל מתאים לדוגמא ממאגר המידע שלנו ובניהן מבצעים בדיקת התאמה בין המשטח עם החלקים המורכבים עליו לבין התמונה של הדוגמא אשר המשתמש נדרש לבנות, ברגע שהמערכת מזהה כי המשתמש סיים לבנות היא מעדכנת את המשתמש כנדרש.

המניפולציות המרחביות מאפשרות לנו להתגבר על נושא מיקום המשטח בתמונה והפרספקטיבה שלו למול הדוגמא במאגר המידע.

תרשים הזרימה של המערכת:



שלב התכנון והעבודה על הפרויקט

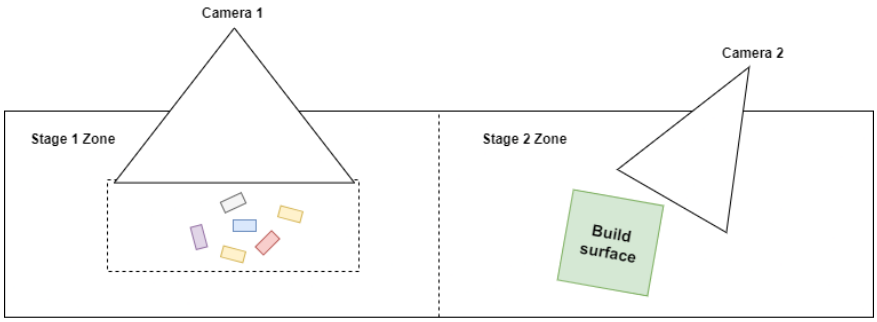
מיקום המצלמות וייעודן

לטובת המימוש אנחנו משתמשים בשתי מצלמות הממוקמות מראש.

מצלמה 1 צופה על מרחב החלקים לטובת שלב 1, את הקלט מהמצלמה אנחנו מעבדים לטובת מיפוי החלקים. המצלמה ממוקמת באופן קבוע ונרצה להימנע מלהזיז אותה על מנת להשתמש בערכים קבועים אשר ניתחנו מראש לטובת פרספקטיבה (לדוגמא גודל של חתיכת לגו מסוג מסוים).

מצלמה 2 צופה על מרחב ההרכבה לטובת שלב 2, את הקלט מהמצלמה אנחנו מעבדים לטובת ההרכבה. מיקום מצלמה זו אינו חייב להיות קבוע אך נדרש להיות בשדה ראייה מתאים להרכבה.

תרשים של מיקום המצלמות:



אתגרי הפרויקט

הפרויקט עוסק בעיבוד תמונה בזמן אמת בשני חלקיו שונים.

בחלק הראשון עסקנו בזיהוי לבנות הלגו ומיפוי על פי סוג, האתגרים בשלב זה היו:

1. מתן מענה לזיהוי החלקים בזויות שונות (חלקי הלגו יכולים להיות בתצורות שונות לדוגמא, מונחים על פאה שונה ובזוויות שונות).
2. שינוי צבע שנובעים מהתאורה, חלקי הלגו מחזירים אור ("מבריקים"), לכן הצבע של חתיכת לגו יכול להשתנות משמעותית באותה חתיכה או בין 2 חתיכות זהות במיקום שונה באותו הפריים.
3. מגע בין חלקים מאותו הצבע, לדוגמא מצב בו 2 חלקים אדומים צמודים כך שנראים כמו חלק אחד ארוך או רחב יותר.

בחלק השני של הפרויקט עבור בניית דוגמא על ידי המשתמש האתגרים איתם התמודדנו הם:

1. עבור קביעת הדוגמאות אותם יכול המשתמש לבנות מצב בו קיימות כמה קומבינציות שונות להרכבה של אותה הצורה לדוגמא, שימוש בשני חלקי 2X2 במקום חלק יחיד בגודל 4X2 (אתגר לוגי ביישום ולא אתגר מבחינת עיבוד תמונה).
2. זיהוי התאמה למול בניית הדוגמא על ידי המשתמש בכיוון שונה או במיקום שונה על הלוח מהדוגמא המוכנה במאגר.
3. זיהוי הצלחה עבור דוגמאות שונות, עבור דוגמאות שונות קביעה האם התמונה של הדוגמא הבנויה זהה לדוגמא מהמאגר אותה המשתמש אמור לבנות יכול להיות יותר מורכבת (דוגמא עם יותר חלקים שונים תוביל לשגיאת MSE גדולה יותר אך יש בה יותר פיצ'רים להשוואת פיצ'רים לדוגמא)

לאורך שני השלבים התמודדנו עם האתגרים הבאים:

1. אתגרי פרספקטיבה סביב זווית המצלמה ביחס לאובייקטים בתמונה ומרחק מהמצלמה אשר יכול לגרור יחסי גודל בעייתיים (לדוגמא 2 חתיכות זהות כאשר אחת קרובה יותר למצלמה עלולות להתפרש כבגדלים שונים).
2. עבודת RT לזיהוי, עיבוד ותגובה לאורך יישום הפרויקט (להימנע ממצב בו המשתמש מסתכל על המסך אך לוקח זמן ארוך מידי עד לשינוי על המסך של המידע הרלוונטי – זיהוי ותיוג חלקים לדוגמא).
3. ניווט בין חלקי הפרויקט השונים, לדוגמא כאשר משתמש מבצע סריקה ובנייה של צורה ולאחר מכן רוצה לסרוק חלקים נוספים ולבצע בניית אחרת.
4. יצירת מאגרי המידע המתאימים (אלו תמונות נדרש עבור דוגמאות בנייה על מנת לזהות התאמה למול תנאי התאורה השונים ותצורות הנייה השונות או קביעת כמות הצילומים ואיזה של לבנה מסוימת צריך על מנת לאמן מודל GMM מספיק טוב).

אילוצי הפרויקט

1. בגלל קושי בזיהוי של חלקים אשר מסתרים את אחד את השני אנו דורשים מהשתמש לבצע הפרדה בין חלקי הלגו, חלקי הלגו אותם נדרש להפריד מסומנים למשתמש בריבוע אדום סביבם על המסך.
2. קביעת החלקים האפשריים מראש, לא ניתן להכניס חלק מסוג לא מוכר, לדוגמא חלק עגול (ביצענו עיבוד מקדים על מנת לקבל את התאמת הממדים בפרספקטיבה של פריים – המרת גודל בלגו לגודל בפיקסלים לארוך עיבוד המקדים של הפריים וזיהוי סוגי החלקים השונים)

הנחות הפרויקט

- המצלמות מקובעת לכן אין שינוי בפרספקטיבה מול חלקי הלגו.
- תנאי סביבה לא משפיעים דרסטית על הגוון, ביצענו התאמה של הצבעים לזיהוי לפי תמונות שצולמו בספרייה.
- לא יתבצע שינוי גבולות גזרה של הפריים הנמשך מכל אחת מהמצלמות.
- לא כל פריים מעניין – נבצע עיבוד רק על פריים לאחר שינוי ולאחר התייצבות בשביל להימנע מחישובים מיותרים.

תיאור אלגוריתם - זיהוי חתיכות הלגו

בחלק זה של הדוח נעסוק באלגוריתם הזיהוי בשלב מיפוי החלקים , אלגוריתם הזיהוי מבוסס על המידע המתקבל ממצלמה 1 אשר עליו מתבצע העיבוד.

הסבר מלא על פונ' בהן השתמשנו מספריית 2cv בפייתון מופיע בהמשך.

הסבר על האלגוריתם

מיפוי החלקים מתבססים על כמה שלבים של עיבוד המידע.

1. <u>משיכת פריים מהמצלמה</u>

על מנת לעמוד בשיקול RT נרצה לבצע עיבוד רק על חלק מהפריימים ולא על כולם מתוך ההנחה כי לא מתבצעים שינויים משמעותיים בתמונה מספר רב של פעמים בשנייה.

נמשוך פריים להמשך עיבוד כאשר הוא עומד בתנאים הבאים:

- שונה מהפריים הקודם לו (סימן שהתבצע שיוני בחלקים הפזורים)
- דומה לפריימים הבאים אחריו (בוצעה התייצבות של התמונה ואין שיוניים נוספים)

את בדיקת התנאים אנחנו מבצעים באופן רציף באמצעות השוואת השינוי בין פריימים על ידי חישוב הMSE בניהם.

2. <u>עיבוד מקדים לפריים</u>

על מנת להתגבר על אתגרי פרספקטיבה אנחנו מבצעים מניפולציות מרחביות לפריים כך שנוכל להתבסס על ממדים הנמדדים מהתמונה.

את התאמת הפרספקטיבה נבצע באמצעות הפונקציות findHomography ו-warpPerspective (הסבר מלא על הפונקציות למטה).

3. <u>חלוקת החתיכות לפי צבע</u>

את המיפוי נבצע ראשית לפי צבע, נבצע mask אשר ישאיר רק את החלקים הרלוונטים בכל פעם בתמונה.

את הmask נבנה באמצעות 2 דרכים שונות ונשלב בניהן (bitwise OR) על מנת לקבל דיוק מרבי:

1. העברת התמונה למרחב צבע hsv וסימון הפיקסלים בעלי הערכים המתאימים לצבע (לכל צבע הכנו מראש את הערכים הרלוונטים באמצעות ניתוח הצבע על תמונות שהכנו מראש בתצורה זהה – בספרייה בתנאי תאורה מתאימים באמצעות המצלמה עליה חתמנו).
2. באמצעות GMM על בסיס למידה של תמונות של חלקי הלגו מהצבע הרלוונטי על רקע השולחן, את הפרמטרים של שיטה זו התאמנו באמצעות ביצוע ניסויים רבים עד להגעה לדיוק מספק.

4. <u>מציאת גדלי החתיכות</u>

על הפריים המעובד נבצע מיפוי של הכתמים באמצעות findcounters כאשר לכל כתם נקבל אובייקט המכיל את המידע עליו.

לאחר מכן נרצה לסמן את המלבן המתאים אשר מתאר את החלק, אותה נמצא באמצעות הפונ' minAreaRecat ו boxpoints אשר נקבל מהן את 4 הקורדינאטות של כל מלבן.

כעת נבצע חישובים למציאת המרחקים בין הנקודות ומיצוע של 2 מרחקים קרובים לקבלת האורך והרחוב של הבלוק.

האורך והרחוב המתקבל הוא מספר המבטא את הגודל בפיקסלים בתמונה לכן נדרש לבצע עליו עיבוד נוסף.

5. <u>מיפוי הלבנים</u>

לכל לבנה נבצע מיפוי של ערכי האורך והרחוב לLongSide ו ShortSide, ערכים אלו מאפשרים לנו לייצר מיפוי בין זוג ערכים בפיקסלים לזוג ערכים ב"עולם הלגו".

שימוש בצד הקצר –
נרצה שהערך הקצר יהיה מתחת לערך סף, ערך סף זה עוזר לנו ל"סנן" כתמים אשר מייצגים מצב בו מספר חתיכות צמודות שלאחר עיבוד המידע מופיעות ככתם יחיד אך עדיין מאפשר לקלוט לבנה שמונחת על הצד.
עבור חתיכות אשר הצד הקצר שלהן מעל הסף הן ככל הנראה מצב בוא 2 חתיכות בעלות צבע זהה קרובות מידי – במצב זה אנחנו מבצעים עיבוד נוסף באמצעות distance transform שמטרתו הוא לפרק את אזור הממשק בניהן.
לאחר מכן אנחנו מבצעים מיפוי מחדש של החתיכות לאחר הניתוק בניהן.

שימוש בצד הארוך –
לפי הערך של הצד הארוך מצאנו את המיפוי הלינארי אשר מתאים ערך בפיקסלים לגודל המתאים בעולם הלגו.

כעת נכניס למאגר המידע כל לבנה מתאימה שמצאנו ונסמן אותה על בפריים המוצג למשתמש בצבע לבן.

חתיכות אשר זוהו ככתם מחובר גדול מידי אך הצלחנו להפריד יסומנו למשתמש בסגול, אלו חתיכות אשר ה"ביטחון" לגבי המיפוי שלהן הוא יותר נמוך (ביצוע העיבוד הנוסף עלול לשנות את המימדים ומכך נקבל מיפוי לא נכון).

כתמים אשר הזיהוי שלהם לא החלטי (לדוגמא חתיכות צמודות) נסמן למשתמש באדום על מנת לבקש ממנו להזיז אותן ובכך לאפשר זיהוי.

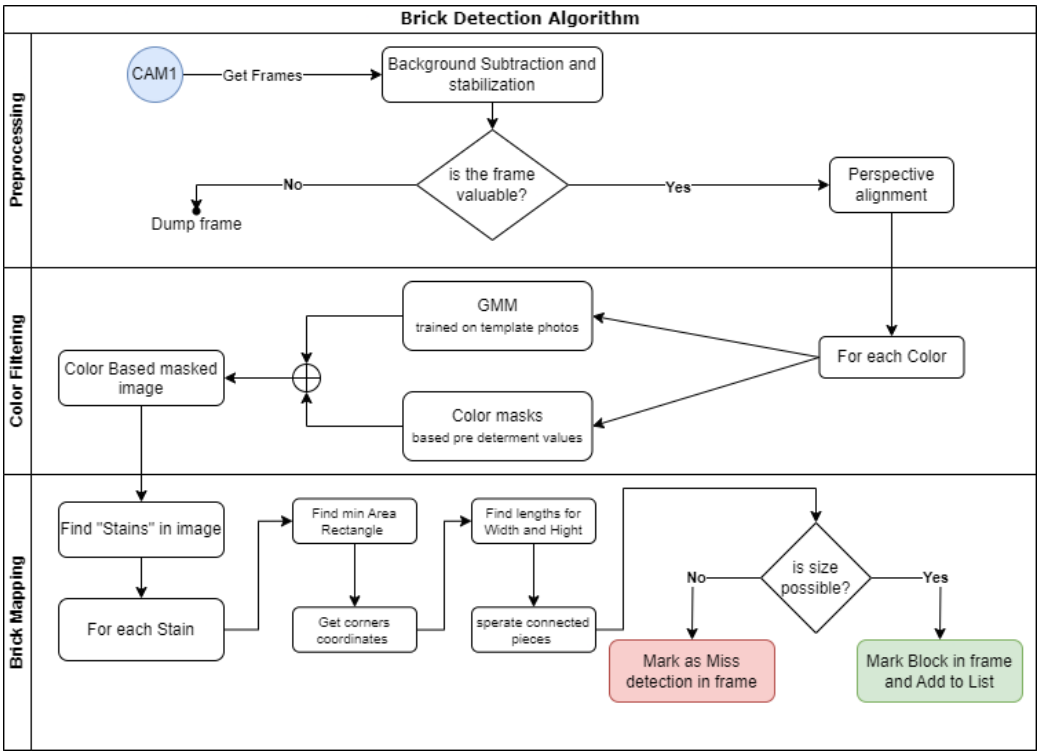
קלט, פלט ופרמטרים

הקלט עבור האלגוריתם הוא כל פריים של המצלמה, האלגוריתם מבצע סינון לפריימים המיותרים ומבצע עיבוד על הפריימים הרלוונטיים בלבד.

הפרמטרים באלגוריתם הם: ערכי הצבע האפשריים לכל צבע עבור הסינון, הפרמטרים לGmm עבור הסינון, ערכי הגודל האפשריים לטובת המיפוי ומרווחי הזמן למשיכת פריים לטובת ניצול משאבים נכון.

הפלט מהאלגוריתם הוא רשימת לבני הלגו לפי צבע, אורך, רוחב וכמות.

תרשים זרימה



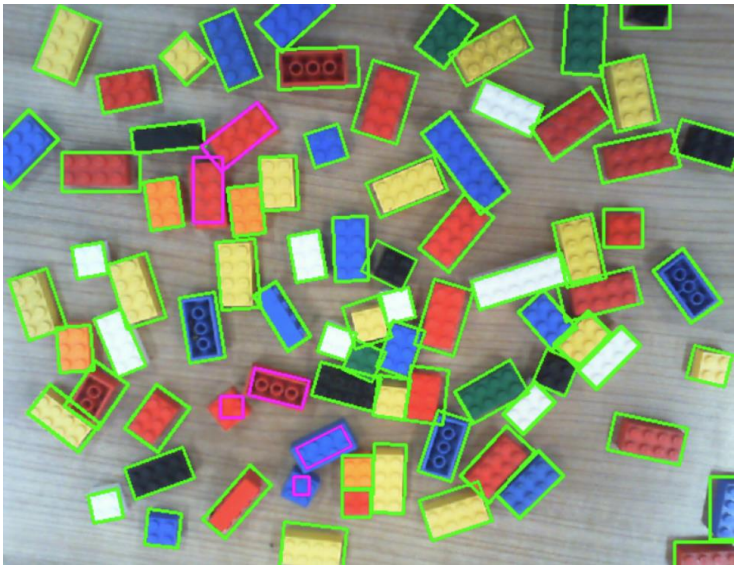
תוצאות ביניים

בתמונה זו ניתן לראות:

מימין תמונה של חלקי הלגו על רקע השולחן.

את התמונה ניתחנו עבור מציאת חלקי הלגו הכחולים בלבד, התמונה בשמאל היא המסכה שנוצרה מהעיבוד והתמונה במרכז היא התמונה המקורית כאשר חיסרנו ממנה את המסכה.

ניתן לראות כי הצלחנו למצוא את החתיכות הכחולות אך קיימים אזורים בעייתיים, לדוגמא עבור מצב בו לא ניתן לראות חלק מהחתיכה בגלל הסתרה על ידי חתיכה אחרת.



בתמונה זו ניתן לראות את חלקי הלגו לאחר המיפוי והסימון למשתמש.

באמצעות העיבוד ושיטת המיפוי הצלחנו לסווג גם חתיכות אשר מונחות על הצד (לדוגמא חתיכה שחורה בחלק השמאלי העליון של התמונה) וגם חתיכות באותו בצבע אשר יש בניהן ממשק (לדוגמא החתיכות המסומנות בצבע סגול).

תיאור אלגוריתם – זיהוי ההרכבה

בחלק זה של הדוח נעסוק באלגוריתם הזיהוי בשלב ההרכבה, האלגוריתם מבוסס על המידע המתקבל ממצלמה 2 אשר עליו מתבצע העיבוד.

הסבר מלא על פונ' בהן השתמשנו מספריית 2cv בפייתון מופיע בהמשך.

הסבר על האלגוריתם

זיהוי ההרכבה מתבסס כל כמה שלבים.

1. <u>משיכת פריים מהמצלמה</u>

על מנת לעמוד בשיקול RT נרצה לבצע עיבוד רק על חלק מהפריימים ולא על כולם מתוך ההנחה כי לא מתבצעים שינויים משמעותיים בתמונה מספר רב של פעמים בשנייה.

נמשוך פריים להמשך עיבוד כאשר הוא עומד בתנאים הבאים:

- שונה מהפריים הקודם לו (סימן שהתבצע שיוני בחלקים הפזורים)
- משטח ההרכב מופיע בו באופן מלא.

את בדיקת התנאי הראשון אנחנו מבצעים באמצעות השוואה לפריים הקודם על ידי חישוב הMSE וקביעה האם הוא מעל סף שקבענו, את התנאי השני אנחנו מבצעים על ידי עצירת העיבוד אם זיהוי המשטח לא התבצע בהצלחה.

את זיהוי משטח ההרכבה ועיבודו ואת עיבוד ההרכבה בתמונה אנחנו מבצעים לפריים הנמשך מהמצלמה ולתמונות מוכנות של הדוגמא אותה המשתמש בונה על מנת לבצע השוואה כמה שיותר "נקייה מרעש".

2. <u>זיהוי משטח ההרכבה</u>

לאחר קבלת פריים רלוונטי נרצה לזהות את משטח ההרכבה בפריים, זיהוי המשטח בלבד מאפשר לנו לבצע סינון לרעש מהרקע ולהכין אותו לעיבוד נוסף לקראת הבדיקה.

את הזיהוי אנחנו מבצעים באמצעות מציאת מרובע כללי באמצעות approxPolyDP, על האוקבייקט המתקבל אנחנו מבצעים בדיקה האם הוא מרובע (מכך כל המשטח בתמונה), במידה ולא נעצור את העיבוד ונחזור לתחילת האלגוריתם.

3. <u>עיבוד מקדים למשטח ההרכבה</u>

על מנת להתגבר על אתגרי פרספקטיבה והמימדים אנחנו מבצעים מניפולציות מרחביות למשטח כך שנוכל להתבסס על התאמה למול תמונת ההרכבה הסופית.

את התאמת הפרספקטיבה נבצע באמצעות הפונקציות findHomography ו-warpPerspective (הסבר מלא על הפונקציות למטה).

על מנת לקבל יכולת התאמה לתמונות מוכנות של הדוגמא אותה המשתמש בונה אנחנו מבצעים התאמה באמצעות שינוי פרספקטיבה של המשטח בתמונה לריבוע בגדול 400X400.

4. <u>עיבוד של ההרכבה בתמונה</u>

כעת עבור התמונה של המשטחה בלבד בגודל קבוע מראש נבצע סינון על מנת להישאר עם ההרכבה בלבד.

ראשית, נבצע סינון של הרקע, את הסינון אנחנו מבצעים על ידי סינון הצבע של משטח ההרכבה מהתמונה.

ניתחנו מראש את ערכי הגוון של משטחי ההרכבה השונים ואנחנו מבצעים סינון שלהם על מציאת הפיקסלים בתחום הגוון הזה ולאחר מכן חיסור של פיקסלים אלו מהתמונה עצמה.

לאחר מכן, נמרכז את ההרכבה למרכז התמונה (על מנת להתגבר על מצב בוא המשתמש בנה את הצורה לא במרכז המשטח), מירכוז ההרכבה מאפשר לבצע בהמשך השוואה יותר מדויקת בין התמונות.

5. <u>בדיקת התאמה</u>

בדיקת ההתאמה מתבססת על שלוש שיטות השוואה כאשר לכל אחת ערף סף שונה בהתאמה לדוגמא אשר המשתמש בונה.

שיטות ההשוואה:

1. חישוב MSE – נבצע חישוב שגיאה בנורמת 2L בין התמונות, את החישוב נבצע לכל ערוץ צבע בנפרד.
2. השוואת היסטוגרמות – נבצע השוואה בין ההסיטוגרמות בכל ערוץ צבע בין התמונה.
3. השוואת פיצ'רים – נבצע השוואת פיצ'רים בין התמונה של ההרכבה לתמונה מהמצלמה, את הפיצ'רים אנחנו מוצאים באמצעות ORB feature detection הסבר מלא על השימוש בהמשך.

לאחר ניתוח של התוצאות עבור בנייה של דוגמאות שונות קבענו את תנאי הסף להצלחה לדוגמאות שונות.

שימוש בתנאי סף שונים מאפשר לנו לזהות הצלחה בצורה מתאימה להרכבה שונות, מניתוח זה הגענו למסקנות שונות לדוגמא כך שעבור הרכבה מסובכת וגדולה ערכי הMSE יהיו גבוהים כי קיים שטח גדול לחישוב מתוך המשטח, אך קיימים מספר גדול של פיצ'רים לכן השוואת הפיצ'רים יתנו לנו אינדיקציה טובה לבנייה.

את השוואה בין ההרכבה אשר נמשכה מהמצלמה לבין 4 תמונות מוכנות מראש של ההרכבה (כל תמונה בכיוון שונה, סיבוב ב90 מעלות) ההשוואה ל4 התמונות מאפשר לנו להתגבר על מצב שהמשתמש בנה כאשר המשטח בזווית ולאחר עיבוד המשטח התמונה המתקבלת מסובבת.

ערכי השגיאות מתייחסים לתוצאה הבעלת ההתאמה הגבוה ביותר בין כלל ההשוואות.

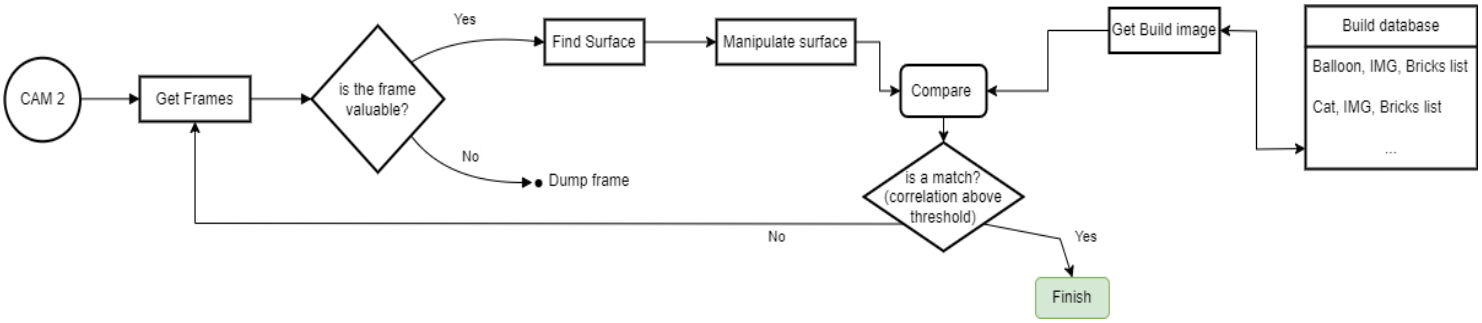
קלט, פלט ופרמטרים

הקלט עבור האלגוריתם הוא כל פריים של המצלמה, האלוגריתם מבצע סינון לפריימים המיותרים ומצבע עיבוד על הפריימים הרלוונטים בלבד.

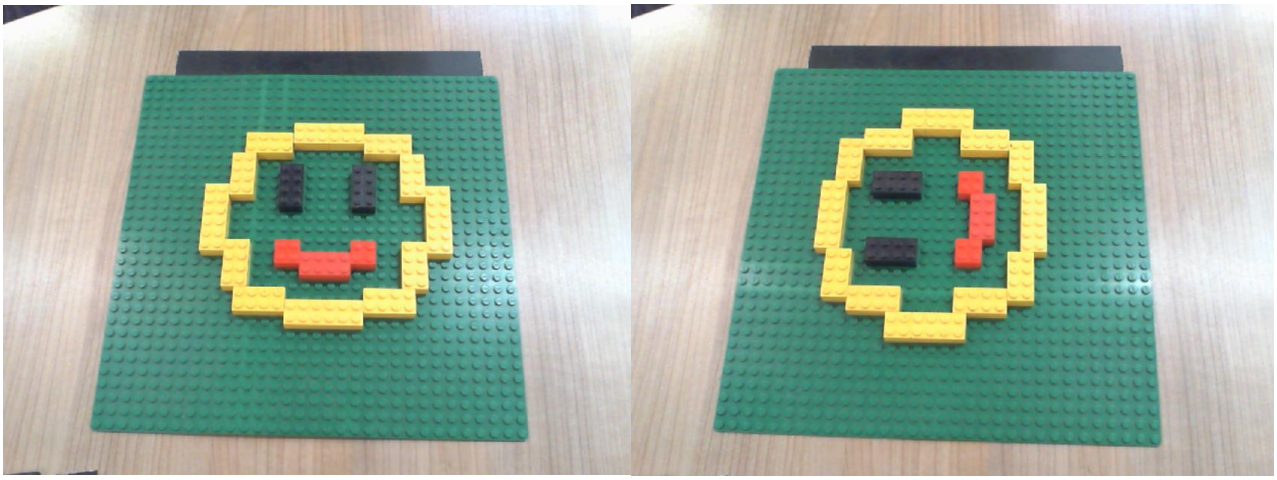
הפרמטרים באלגוריתם הם: ערכי הסף עבור התאמת ההרכבה לדוגמא המוכנה.

הפלט מהאלגוריתם הוא מידת "הביטחון" של המערכת האם ההרכבה זהה לדוגמא.

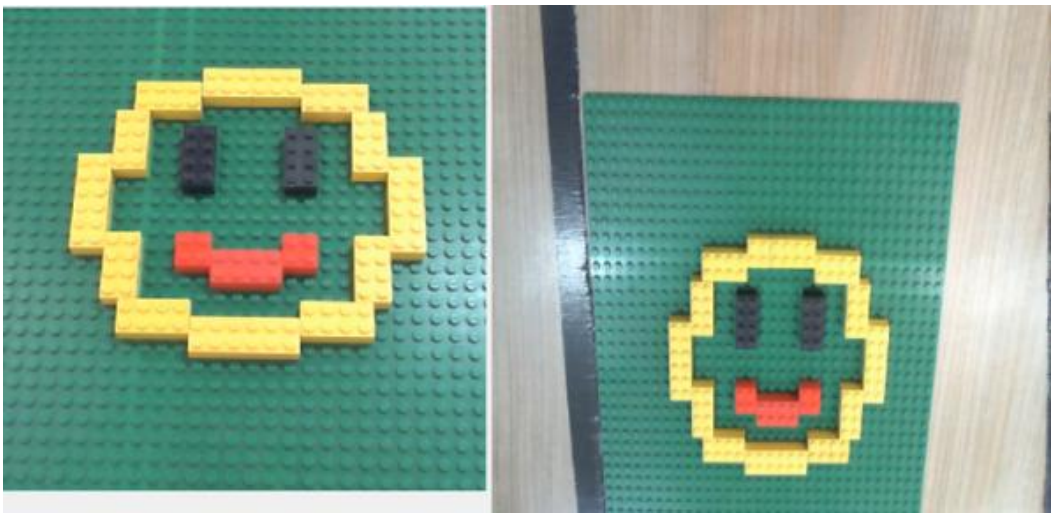
תרשים זרימה



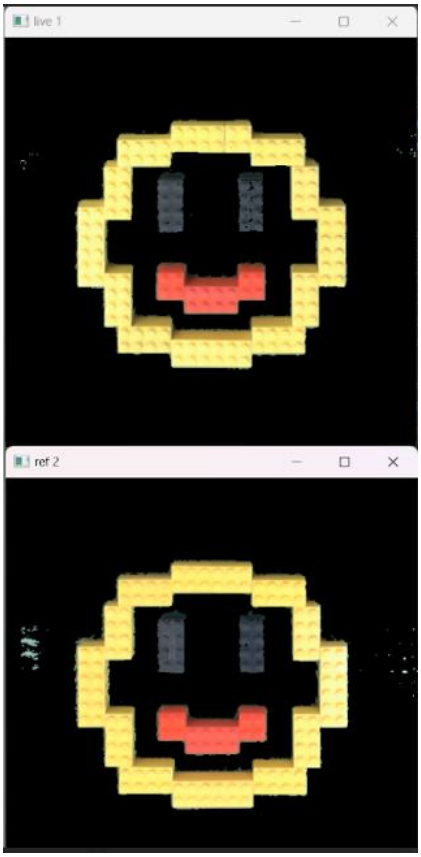
תוצאות ביניים



התמונות המוכנות של ההרכבה לפני עיבוד (2 זוויות מתוך ה4).



תמונה מתוך האפליקציה (צד ימין זו התמונה הנמשכת מהמצלמה שמוצגת למשתמש לפני עיבוד ומצד שמאל תמונה לדוגמא של ההרכבה).



התמונות לאחר עיבוד – העליונה היא מהפריים הנמשך מהמצלמה, התחתונה מהתמונות המוכנות להשוואה.

ניתן לראות כי ההרכבה בשתי התמונות ממורכזות, לאחר סינון של הרקע ולאחר יישור של התמונה מבחינת פרספקטיבה.

עם זאת, קיים שוני בין התמונות – לדוגמא הגוונים (נובע מהתאורה) ושיטת ההרכבה (שימוש שונה בלבני הלגו). מכך על מנת לוודא התאמה השתמשנו בשלושת השיטות אשר מאפשרות יחד דיוק יותר גבוה.

בין 2 התמונות מתבצעת השוואה לפי השיטות שציינו למעלה.

(את ההשוואה מבצעים למול 4 זוויות שונות של ההרכבה למקרה שהמשתמש הרכיב בזווית שונה מהדוגמא המוכנה מראש).

הסבר על פונקציות משמעותיות

- פונקציות cv2.findHomography היא חלק מספריות OpenCV בשפת פייתון. היא משמשת לאיתור הומוגרפיה (הקשר המרחבי לדוגמא שינוי פרספקטיבה, סיבוב מתיחה וכדומה) בין שתי תמונות או בין חלקים של תמונה אחת.
הפונקציה מקבלת כפרמטרים שתי רשימות של נקודות בקורדינטות (תמונות או נקודות בתמונה), ומחזירה את המטריצה של ההומוגרפיה שמתאימה בין הנקודות בשתי התמונות (כפי שלמדנו בהרצאות עבור הזזה סיבוב וכדומה), כך שלכל זוג נקודות:
$(x_i, y_i) \in src\ image$, $(u_i, v_i) \in dst\ image$ נקבל $\forall i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = H \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$

- פונקציית cv2.warpPerspective היא חלק מספריות OpenCV בשפת פייתון. היא משמשת לביצוע התאמה פרספקטיבית (perspective transformation) על תמונה.
משמעות המילה פרספקטיבה היא שהתמונה משתנה כך שנראית כאילו צופים בה מזווית אחרת, זה נחוץ לעיתים כאשר אנו רוצים להתאים תמונה של אובייקט שצולם מכמה זוויות או מרחקים שונים.
הפונקציה מקבלת כפרמטרים את התמונה שאנו רוצים להתאים, את המטריצה של ההומוגרפיה, וגודל התמונה החדשה אליה נרצה להתאים את התמונה המקורית (בפרויקט השתמשנו באינטרפולציה לקביעת ערכי פיקסלים "חדשים" מההזזה).ע"י
$(x', y') \in src\ image$, $(x, y) \in dst\ image, M\ Transformation\ Matrix$ נקבל
$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = M \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$ ולאחר מכן מתבצעת אינטרפולציה לחישוב ערכי (x,y) אינרפולציה זאת מתבצעת מכיוון שאנו יכולים לשנות את גודל התמונה ומיקומי (x,y) לא יהיו בדיוק במיקומי הפיקסלים של תמונת המקור (קואורדינטות הן מספרים שלמים)

- פונקציית cv2.findContours היא חלק מספריית OpenCV בשפת פייתון. היא משמשת לאיתור קווים סגורים בתמונה, שיכולים להיות קווי מתאר צורות, או שטחים מוגדרים.
בעזרת הפונקציה, ניתן למצוא אובייקטים בתמונה ולהפעיל עליהם פעולות כמו סיבוב, חיתוך, וכו'.
היא מחזירה רשימה של קווים סגורים, כל קו נכלל ברשימה מכיל את הנקודות המרכזיות שלו, גודל השטח והקו' של הקודקודים.
הפונ' מצבעת מעבר רקורסיבי על הפיקסלים בתמונה כך שכל פיקסל לבן שהיא מוצאת היא מתחילה לעבור עליו בצורה רקורסיבית עד לקבלת גודל כל האובייקט (או קונטור) וכן שומרת לכל אובייקט את מספר הפיקסלים שאותם מצאה ששייכים לו וכן את קצוות האובייקט לשימוש בפונקציות אחרות.

- פונקציית cv2.distanceTransform היא חלק מספריית OpenCV בשפת פייתון. היא משמשת לחישוב הטרנספורמציה שמטרתה היא צמצום הגבולות הדקים והמחוברים של שני אובייקטים שזוהו כאחד בעזרת הפונקציה הקודמת, כלומר במקרה בו יש לנו שני חלקים שנוגעים בפינה נקבל כי החלק של הפינה יהיה דק יותר ולכן בעזרת פונקציה זו נוכל להפריד בין שני החלקים ולסווג אותם כראוי.
הפונקציה עושה פעולה זו על ידי חישוב המרחק של כל פיקס מהפיקסל השחור הקרוב ביותר כך שהתמונה המתקבלת בסוף היא בעצם תמונה של החלקים העבים יותר באובייקטים בעוד הקצוות והחלקים הדקים (או במקרה שלנו הפינות המחוברות) נעלמים.

- פונקציית minAreaRect היא חלק מספריית OpenCV בשפת פייתון. היא משמשת למציאה של המלבן המינימלי המקיף אובייקט בתמונה.אנחנו משתמשים בפונקציה הנל על מנת לקבל ריבוע שעל פיו נוכל לסמן כל חתיכת לגו בדיוק בגודל של החתיכה.
הפונקציה בעצם בודקת את הערך המקסימלי של כל ציר וכן את המינימלי וככה מקבלת מלבן המאורך לצירים. וכן בודקת זאת על גבי כל אחת מהזוויות האפשריות (180)וכן מוצאת את הזווית שתחזיר את הערך המינימלי מבחינת גודל האובייקט.

אנליזה של אחד החלקים

בחלק זה ביצענו אנליזה של פונקציה בתוכנית שלנו.

את האנליזה ביצענו על פונקציית ה-scan אשר מבצעת זיהוי של חלקים על משטח בחלוקה לפי צבע וגודל החלק.

ראשית, את הבדיקות לטובת האנליזה ביצענו במיקום שונה אשר אינו בספרייה ועל משטח בצבע שונה מהשולחן בספריה (השולחן בספרייה בצבע חום והמשטח עליו ביצענו את האנליזה היה שולחן אשר עליו מפה לבנה).

את האנליזה ביצענו ע"י השוואת תוצאות ה-scan בתנאים סביבתיים שונים אשר כוללים שינוי תאורה ושינוי גובה המצלמה.

את תוצאות האנליזה הצגנו בעזרת חלוקה ל-3 פרמטרים:

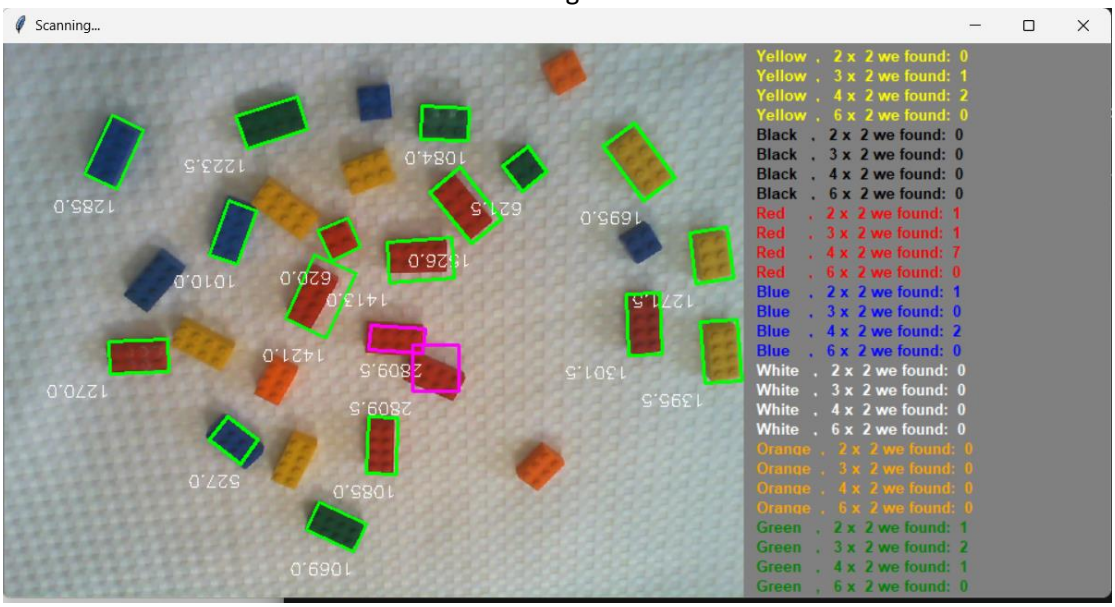
1. True positive- המערכת זיהתה חלק נכון אשר קיים על המשטח.
2. False positive- המערכת זיהתה חלק אשר לא קיים על המשטח.
3. False negative- המערכת לא זיהתה חלק אשר מופיע על המשטח.

את התוצאות עבור ה- True negative לא הצגנו מכיוון שהן אינן מדידות במערכת שלנו בשל העובדה שהן מציינות חלקים שהמערכת לא זיהתה ואכן לא קיימים, דבר שלא מדיד מבחינתינו.

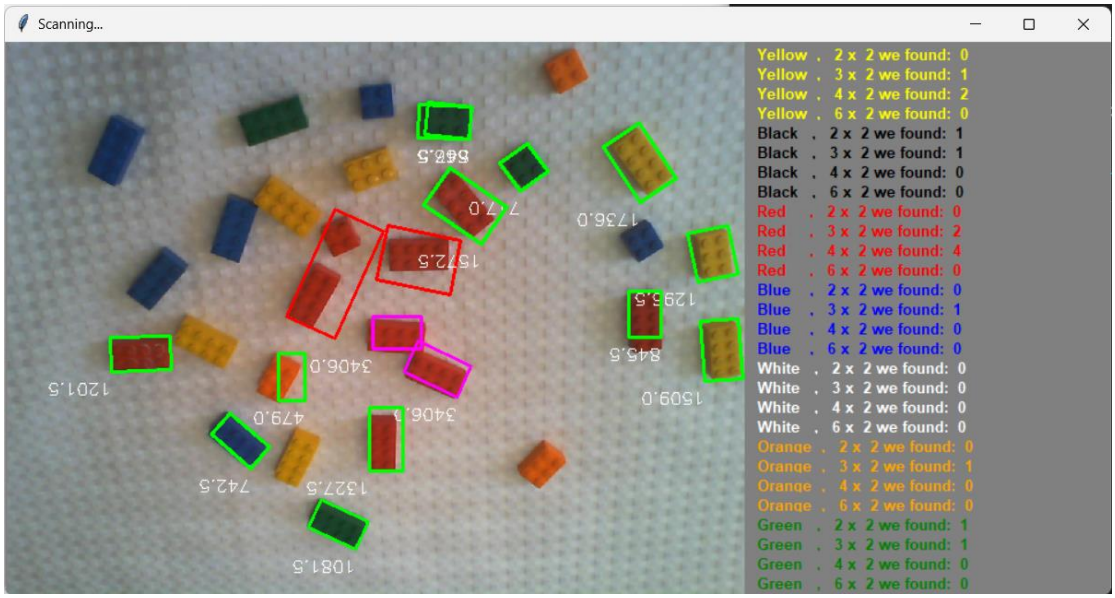
לכל אחת מהבדיקות נציג תמונות לדוגמא מהבדיקה, גרף המרכז את התוצאות ומסקנות מתוך התוצאות.

1. שינוי תנאי התאורה:
   תמונות לדוגמא-

Light 3

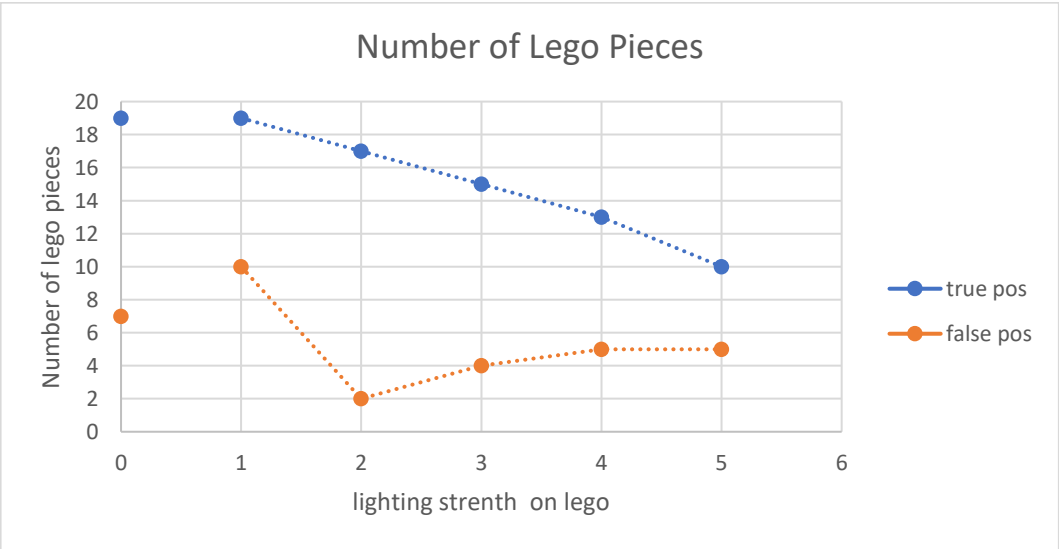


Light 1



ניתן לראות כי תנאי התאורה השונים משנים את התוצאות, בדוגמא זו ניתן לראות שהתאורה הנמוכה יותר מקשה על המערכת לזהות את ההבדל בין 2 החלקים האדומים ולכן מאחדת אותם לחלק 1.

כעת נסתכל על גרף התוצאות המרוכזות:

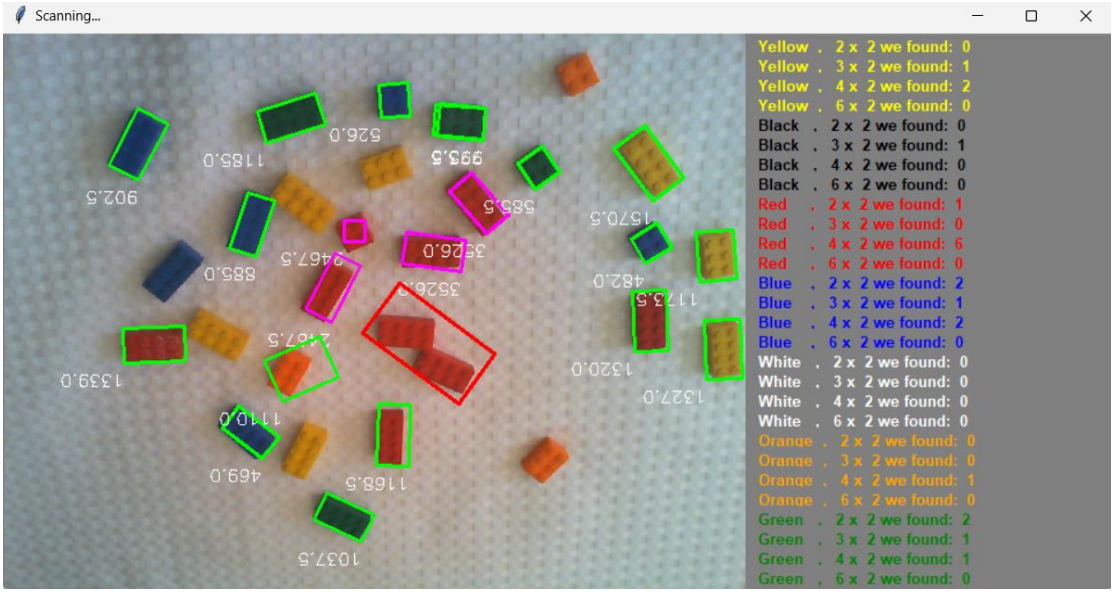


נשים לב כי ככל שמקורות האור רבים יותר נקבל כי מספר הזיהויים הנכונים יפחתו וכן מספר הזיהויים הלא נכונים יעלו. זאת ניתן להסביר כי כנראה תנאי התאורה שהאלגוריתם שלנו אומן עליהם פועלים במיטבם על מקורות אור מועטים, כמו בספריה, ופחות טוב עם הרבה מקורות אור. נשים לב כי תאורה תשפיע על mask שכל צבע צריך לקבל ולכן התוצאות ישתנו ונצטרך להתאים את האלגוריתם למצב החדש.

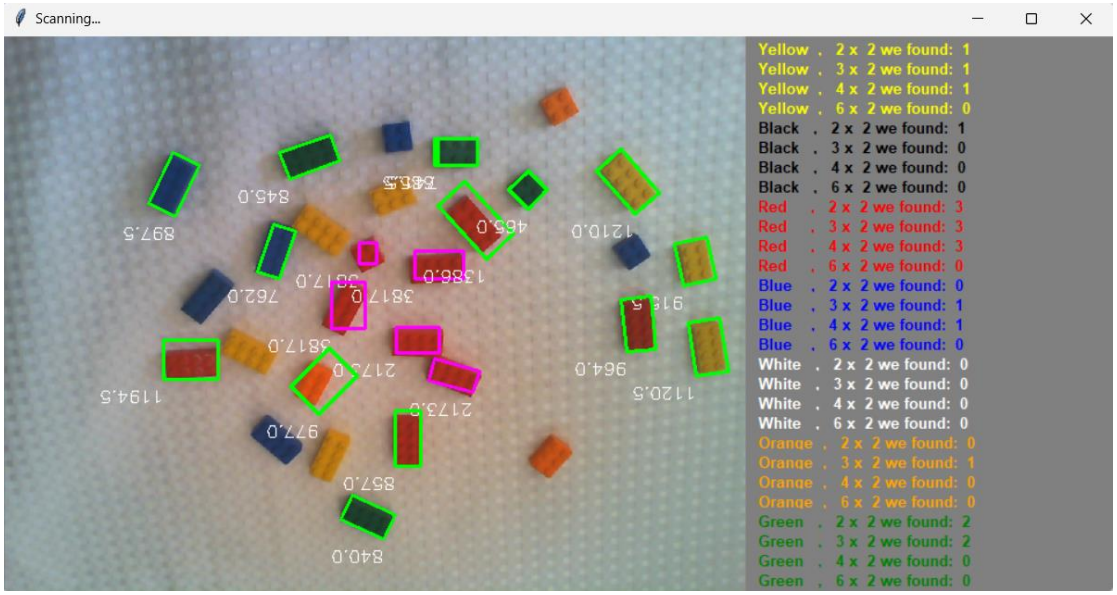
2. שינוי המרחק:

תמונות לדוגמא-

Distance 0

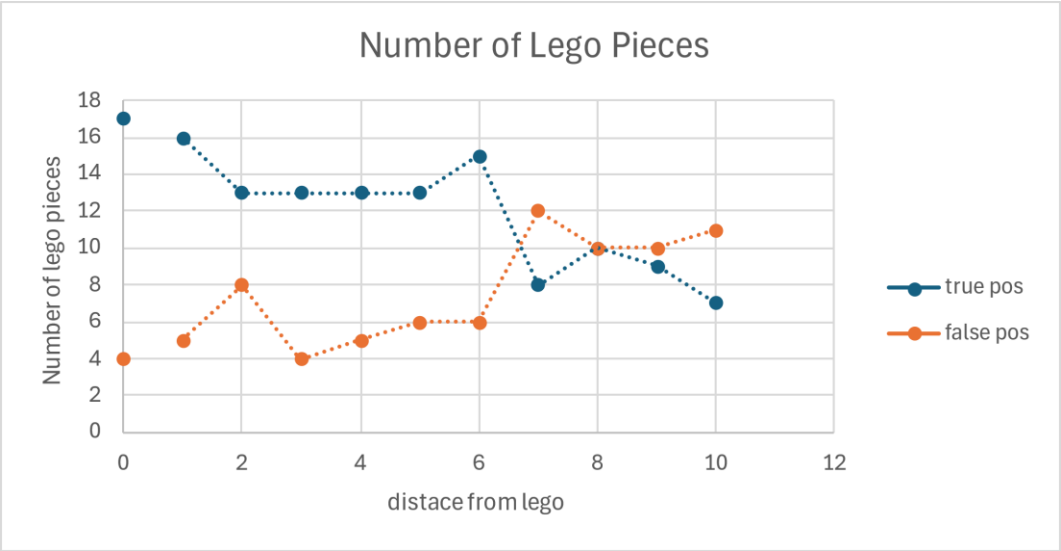


distance 8



ניתן לראות כי עבור המרחקים השונים קיבלנו תוצאות מעט שונות, נראה כי עבור החלקים האדומים במרכז הלוח במרחק של distance 8 נראה כי המערכת הצליחה להפריד בין החלקים האדומים, אך עם זאת ניתן לראות בהסתכלות על התוצאות כי החלקים ב-distance 8 מזוהים כקטנים יותר בשטחם ומכך ישנה אפשרות שחלקים אלו יזוהו כקטנים יותר ממה שהם במציאות (כלומר חלק שיזוהה כחלק של 2X2 במקום 2X4).

כעת נסתכל על גרף התוצאות המרוכזות:



0 זה המרחק איתו אימנו את האלגוריתם שלנו.

נשים לב כי כפי שציפינו נקבל כי ככל שנרחיק את המצלמה מהלגו נקבל שמספר החתיכות שהאלגוריתם שלנו מזהה נכון ירד וכן מספר החתיכות שהוא מזהה לא נכון יעלה, זאת משום שהאלגוריתם מחשב את גודל החתיכה על פי כמות הפיקסלים ולכן במרחק גדול הוא יקבל פחות פיקסלים ויחשוב שהחתיכה קטנה יותר מהמציאות. כלומר נקבל מקרים שחתיכות גדולות יזוהו כקטנות ואז גם ירד לנו סך החתיכות שמזוהה נכון וגם יעלה סך החתיכות שמזוהה סתם.

סיכום

הפרויקט עסק בכמה תחומים שונים של עיבוד תמונה ושל תהליך היישום של רעיון עד אפליקציה עובדת באופן כללי.

כחלק מהעבודה על פרויקט התנסנו בכמה דברים עיקריים:

1. עיבוד תמונה בזמן אמת למול ממשק אינטראקטיבי, והאתגרים וה- "Trade-off" הנובעים מכך.
2. בחירת שיטות עיבוד תמונה מתאימות ויעילות ביותר לכל משימה, תוך התחשבות בדרישות הביצועים, הדיוק והמורכבות ללא שימוש בלמידה עמוקה (לדוגמא השוואת תמונה של אובייקטים וסיווג האם זהים מבלי רשת CNN אשר הייתה "מקלה" עלינו).
3. אינטגרציה חלקה של רכיבי עיבוד התמונה השונים, ממשקי משתמש, החומרה הייעודית (המצלמות) ורכיבים נוספים בתוך הפרויקט.
4. התנסות ומימוש של שיטות ורעיונות שונים בעיבוד תמונה אשר למדנו בביתה למול התמודדות עם רעשים, תאורה לא אחידה, עיוותים גיאומטריים וגורמים אחרים המשפיעים על איכות התמונה ועל אמינות התוצאות של שיטות שונות.

הצגת הפרויקט כחלק מכנס הפרויקטים אפשר קבלת "פידבק" לגבי המערכת כולה מגורמים שונים ולמידה מכך.
בנוסף, להסתובב ולחוות פרויקטים של קבוצות אחרות אפשר לנו לראות את דרכי הפתרון ומימושם לבעיות בעיבוד תמונה אשר הן נתקלו בה.

ככלל, הפרויקט אפשר התנסות מעשית רחבה בתחום עיבוד תמונה ופיתוח יישומים ו"עבודה עם הידיים" בשיטות והחומר הנלמד בקורס.

ההתמודדות עם האתגרים טכניים מורכבים תרמה רבות להבנת החומר הנלמד ולרכישת מיומנויות יישומיות. הפרויקט הדגיש את החשיבות של שילוב בין ידע תיאורטי, מיומנויות תכנות ופתרון בעיות יצירתי לצורך פיתוח יישומים מוצלחים בתחום עיבוד התמונה.

ביבליוגרפיה

[1] Utami, Sri, Nuzul Qur'aniati, and Erlita Kusuma. "Playing Lego Increase Cognitive Development on Preschool Child (4-5 Years Old)." *Jurnal Ners* 3.2 (2008): 121-127.

[2] Sari, D., et al. "Improving Visual Spatial Ability of Children 3-4 Years Through Playing Lego." *Proceedings of the 2nd International Conference on Local Wisdom, INCOLWIS 2019, August 29-30, 2019, Padang, West Sumatera, Indonesia*. 2019.

קישור לסרטון