

# מעבדה לעיבוד אותות פיזיולוגיים

דו"ח מסכם - עיבוד תמונה

מגישי הדו"ח המדהימים:

ארן אהרוני

עדן קורש

אוראל בטיטו

209010156

206845315

315432724

מדריכת המעבדה המרנינה:

נופר ירימי

26/01/2024

1	תוכן עניינים	1
2	תקציר	2
3	ניסויים	3
3.1	ניסוי 1	3.1
3.1.1	היפותזה	3.1.1
3.1.2	מתודולוגיה	3.1.2
3.1.3	תוצאות	3.1.3
3.1.4	מסקנות	3.1.4
3.2	ניסוי 2	3.2
3.2.1	היפותזה	3.2.1
3.2.2	מתודולוגיה	3.2.2
3.2.3	תוצאות	3.2.3
3.2.4	מסקנות	3.2.4
3.3	ניסוי 3	3.3
3.3.1	היפותזה	3.3.1
3.3.2	מתודולוגיה	3.3.2
3.3.3	תוצאות	3.3.3
3.3.4	מסקנות	3.3.4
3.4	ניסוי 4	3.4
3.4.1	היפותזה	3.4.1
3.4.2	מתודולוגיה	3.4.2
3.4.3	תוצאות	3.4.3
3.4.4	מסקנות	3.4.4
4	מסקנות כלליות	4
5	ביבליוגרפיה	5
6	נספחים	6
6.1	נספח א- קוד המטלב	6.1

## 2 תקציר

במעבדה זו חקרנו את מבנה התמונה ותכונותיה במישור המרחבי ובתחום התדר, תוך התמקדות בסינון תמונות, ביצוע פעולות מורפולוגיות, סגמנטציה, זיהוי גבולות, וחילוץ מאפייני אובייקטים. כמו כן, ניסינו לעבד ולנתח סרטוני וידאו. המעבדה העמיקה את הבנתו בתחום עיבוד תמונה וידאו לזיהוי וניתוח של תמונות ואובייקטים ויזואליים, תוך הבנת חשיבות השימוש במסכות נכונות, הסרת רעשים, והתאמה בין תמונות כיוול לוידאו. כמו כן, הדגשנו את הצורך בשימוש בציוד איכותי ובטכניקות מתקדמות לקבלת תוצאות מדויקות יותר.

בניסוי הראשון, עבדנו על תמונת אורז אליה התייחסנו כטסיות דם, כשמטרתנו הייתה לבצע סינון כך שנקבל תמונה עם הטסיות הנוטות ימינה בלבד. הסרת הרקע הובילה לשיפור באיכות הסגמנטציה ואפשרה יצירת מסכה מדויקת יותר. כמו כן, השתמשנו בפילטרים ושיטות שונות לזיהוי גבולות, והדגמנו את היעילות של שימוש במאפיינים כמו אוריינטציה לזיהוי והפרדה של אובייקטים לפי כיווניות מסוימת.

בניסוי השני, סיננו רעש מלח פלפל מתמונה באמצעות מסנן חציון, וקיבלנו כי גרעין בגודל  $3 \times 3$  הינו האופטימלי לסינון הרעש תוך כדי שמירה על חדות התמונה עבור התמונה שסיננו.

בניסוי השלישי, סיווגנו כיווני הסתכלות של נבדק בסרטון וידאו, תוך שימוש בתמונות כיוול ומציאת מיקומי העיניים. התאמה נכונה לתמונות הכיוול הוכיחה את חשיבותה לאחוזי ההצלחה של זיהוי אזור ההסתכלות. בנוסף, ביצוע מבחנים סטטיסטיים יובילו לכימות טוב של טיב הזיהוי וישנה חשיבות רבה לבחירת המבחן הסטטיסטי המתאים.

בניסוי הרביעי, ניתחנו את תנועת רגלו הימנית של הנבדק באמצעות מרקרים, תוך ניתוח מיקום מרקרים, מהירות, וזוויות הרגליים. ראינו את החשיבות של איכות המצלמה, יציבותה, ותאורה מתאימה לזיהוי מדויק של המרקרים. הסקנו כי דרך זיהוי המרקרים אינה מספקת ונדרש לייעל את שיטת הזיהוי מעבר.

### 3 ניסויים

#### 3.1 ניסוי 1

##### 3.1.1 היפותזה

בניסוי זה נשתמש בתמונה של אורז, אליה נתייחס במהלך הניסוי כתמונה של טסיות דם. מטרת הניסוי היא ליצור תמונה המורכבת אך ורק מטסיות הנוטות ימינה (/) ולא הנוטות שמאלה (\), כלומר להישאר רק עם כחצי מהטסיות בתמונה. לשם כך אנו נדרשים לבצע סגמנטציה ולאחר מכן מיצוי מאפיינים של טסיות הדם. את המסכה ניצור פעם אחת בעזרת התמונה המקורית ופעם בעזרת התמונה המקורית ללא הרקע אשר נשערך באמצעות erosion. נצפה כי בשתי הדרכים נצליח להבדיל בין הטסיות לרקע, זאת בשל ההבדלים הברורים יחסית הקיימים ביניהם ומאפשרים בחירת סף טוב מספיק. כמו כן, נצפה כי המסכה המיטבית להפרדת טסיות מהרקע תתקבל מהתמונה ללא הרקע, זאת מכיוון שבהחסרתו תתקבל תמונה עם רמת רעש נמוכה יותר. לאחר מכן, נבצע זיהוי גבולות בעזרת שתי שיטות, שיטת LoG ושיטת dilation והחסרת המסכה. נצפה כי בשתי השיטות יתקבלו תוצאות טובות וכי נצליח להבדיל בצורה טובה בין הטסיות הפונות ימינה לאלו הפונות שמאלה בשל האוריינטציה השונה שלהן וכך לקבל תמונה סופית המציגה את הטסיות הפונות ימינה.

##### 3.1.2 מתודולוגיה

כלים וחומרים: תוכנת המטלב.

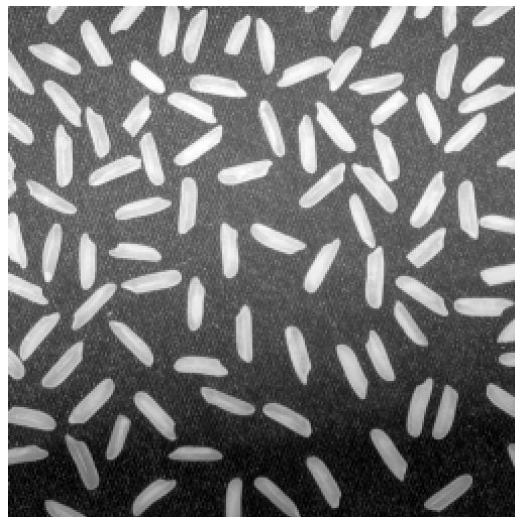
מהלך הניסוי: ראשית, נטען את קובץ התמונה הקיים במטלב. לאחר מכן נציג את רמות האפור של התמונה בעזרת היסטוגרמה, ננסה למצוא סף יחיד המבדיל בין הטסיות לבין הרקע ובעזרתו נבנה את המסכה. ניצור מבנה בצורת דיסק בעזרת פקודת strel במטלב בגודל שנקבע בעצמינו וכך נעלים את הטסיות לקבלת הרקע בלבד. בשלב השני, ניעזר בפקודות bwlabel ו-regionprops על מנת למצוא רק את הטסיות הנוטות ימינה וניצור מסיכה המכילה רק אותן.

##### 3.1.3 תוצאות

תשובה לשאלה 1.1:

נציג את התמונה המקורית:

Blood Platelets Photo

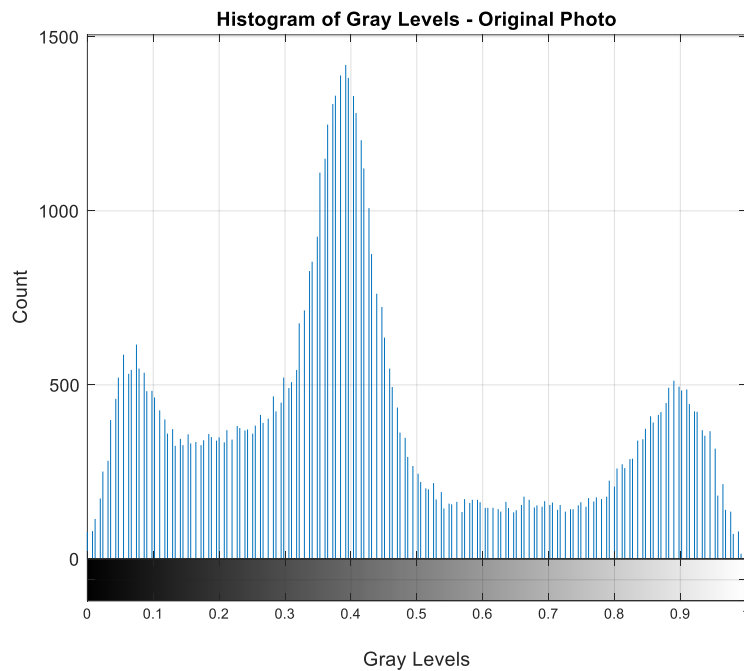


איור 1: תמונת טסיות הדם (אורז)

באיור 1 ניתן לראות את תמונת טסיות הדם עליה נבצע את הפעולות. נשים לב כי גוון הטסיות בהיר יותר מרקע התמונה. כמו כן, רקע התמונה אינו אחיד, בחלקו התחתון של התמונה הרקע שחור יותר (ככל הנראה צל) ובחלקו העליון בהיר יותר.

### תשובה לשאלה 1.2 :

נציג את היסטוגרמת רמות האפור של התמונה לעיל :

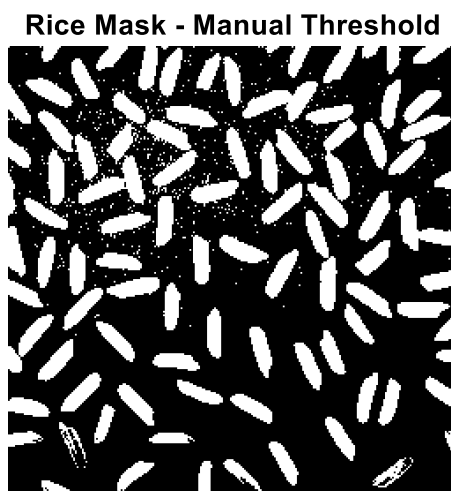


איור 2 : היסטוגרמת רמות אפור

באיור 2 ניתן לראות את היסטוגרמת רמות האפור של תמונת הטסיות. נשים לב כי ישנם 3 פיקים משמעותיים ; בסביבות 0.9 - ככל הנראה אלו גווני טסיות הדם (בהיר), 0.4 - ככל הנראה הרקע בחלקו העליון של התמונה ו-0.05 - ככל הנראה הרקע בחלקו התחתון של התמונה (כהה). למרות הפיקים הנ"ל נראה כי התפלגות רמות האפור מתפרשת על כל הטווח כלומר כל רמות האפור מופיעות ברמה זו או אחרת בתמונה.

### תשובה לשאלה 1.3 :

בחרנו את הסף להיות 0.5 לאחר שבחנו את ההיסטוגרמה באיור 2 ואת המסכה המתקבלת מספים שונים. להלן המסכה הטובה ביותר שקיבלנו :



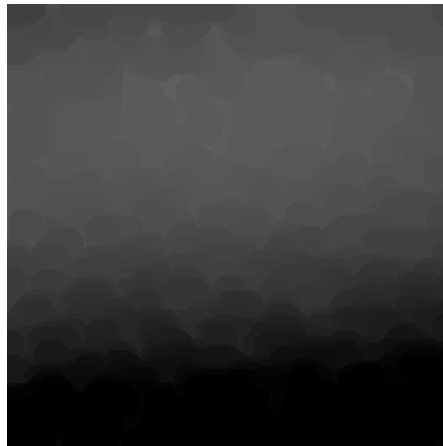
איור 3 : המסכה שהתקבלה בבחירת הסף להיות 0.5

באיור 3 ניתן לראות את המסכה שיצרנו. נשים לב כי רוב הטסיות מסומנות בצבע לבן במלואן חוץ מכמה טסיות בודדות בתחתית התמונה. למרות זאת, נקבל רעש של נקודות לבנות קטנות בחלקו העליון של התמונה. דבר זה הינו טרייד אוף, בכדי לקבל טסיות המסומנות במלואן נצטרך להקטין את הסף אך דבר זה יוסיף עוד רעש לתמונה וכאשר נגדיל את הסף להיות גדול יותר, לא יופיע הרבה רעש אך הטסיות לא יהיו מסומנות במלואן.

תשובה לשאלה 1.4 :

ניצור מבנה בצורת דיסק על מנת לסנן את הטסיות מהתמונה ולקבל את הרקע :

#### Background

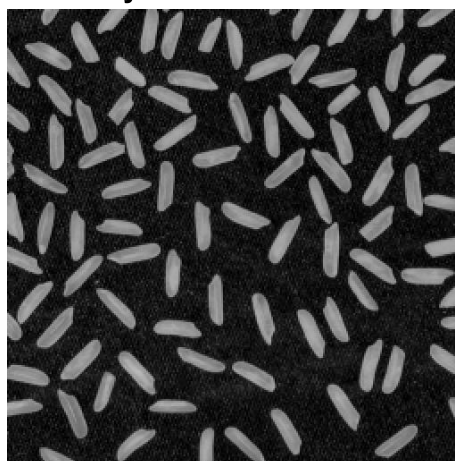


איור 4 : הרקע המתקבל לאחר סינון הטסיות

באיור 4 ניתן לראות את הרקע המתקבל ללא הטסיות. את רקע זה קיבלנו על ידי erosion עם מבנה בצורת דיסק ברדיוס של 10. הטסיות קרובות להיות בצורה אליפטית ועל ידי מדידה ידנית של התמונה המקורית קיבלנו כי רדיוס המשנה (הקטן) שלהן הינו 5. בלקיחת מבנה בעל רדיוס גדול יותר פי 2, אנו מבטיחים שהמבנה לא יתאים לשום חלק בטסית, כלומר פעולת ה-erosion תעלים את הטסיות בצורה טובה.

נפחית את הרקע מתמונת המקור :

#### Only the Blood Platlets

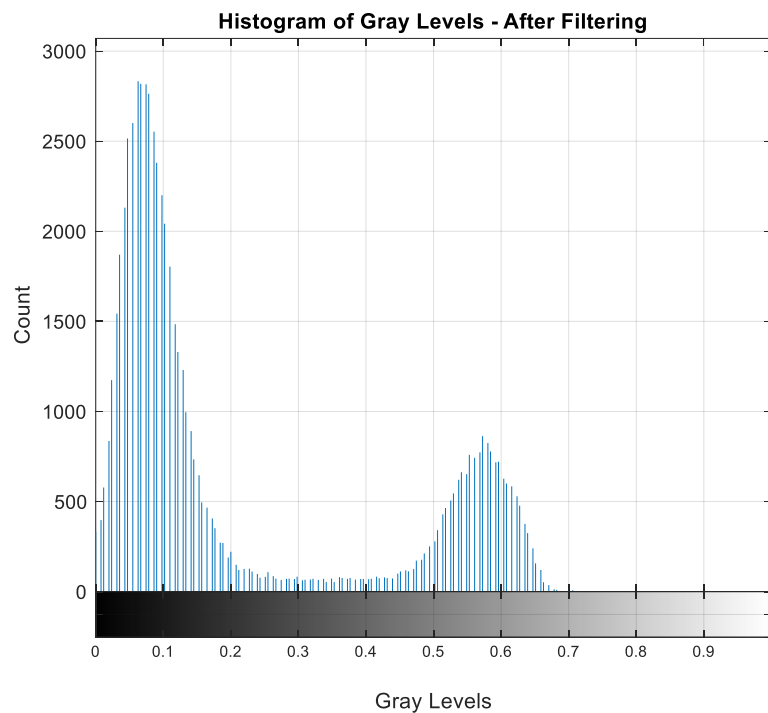


איור 5 : התמונה המתקבלת לאחר הפחתת הרקע

באיור 5 ניתן לראות את התמונה המתקבלת לאחר הפחתת הרקע מהתמונה המקורית. בשונה מאיור 1, הרקע של איור זה אחיד יותר בגוונו דבר שגורם להפרדה טובה יותר של טסיות הדם מהרקע. בנוסף, נשים לב כי גווני האפור בתמונה כהים יותר כתוצאה מהחסרת הרקע.

### תשובה לשאלה 1.5 :

כעת נבצע את הפעולות הקודמות שביצענו על התמונה החדשה (איור 5). נציג את היסטוגרמת התמונה :

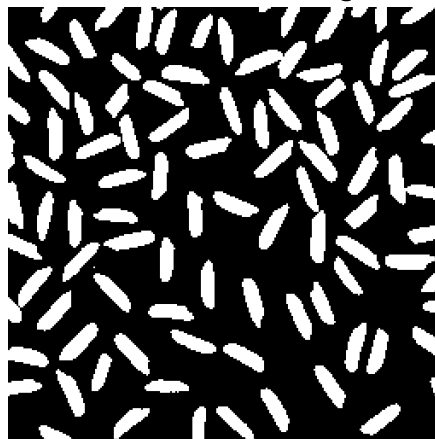


איור 6 : היסטוגרמת רמות האפור של התמונה החדשה שהתקבלה

באיור 6 ניתן לראות את היסטוגרמת רמות האפור. נשים לב כי בשונה מאיור 2, התפלגות ההיסטוגרמה כעת על טווח מצומצם יותר, בין 0 ל-0.7 בקירוב, וישנם 2 פיקים עיקריים המופרדים בצורה מובהקת יותר. כלומר, הפיק הימני (בסביבות 0.58) מייצג את טסיות הדם, בעוד שהפיק השמאלי (בסביבות 0.05) מייצג את רוב רקע התמונה.

כעת ליצירת המסכה בחרנו את הסף להיות 0.35 :

Blood Platelets Mask - After Background Removal

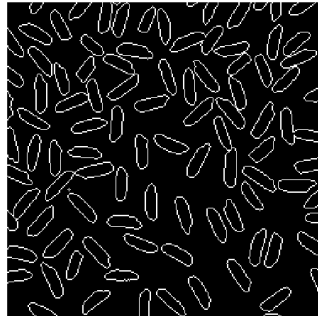


איור 7 : המסכה החדשה בעת בחירת הסף להיות 0.35

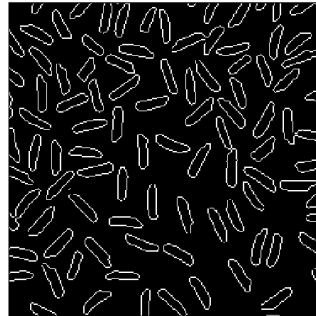
באיור 7 ניתן לראות את המסכה החדשה שיצרנו. בשונה מאיור 3, קיבלנו הפעם מסכה טובה יותר, ללא רעש וטסיות דם המסומנות במלואן. לא התחרשה בפעולה זו טרייד אוף, לכן קיבלנו מסכה טובה יותר.

כעת נבצע זיהוי גבולות בשתי דרכים; החסרת התמונה לאחר dilation מהתמונה המקורית ובעזרת אופרטור LoG (Laplacian of Gaussian). נציג את שתי השיטות אחת ליד השנייה:

Blood Platelets After Dilation-Mask



Blood Platelets After LoG



איור 8: זיהוי גבולות הטסיות בשיטת LoG (ימין) וביצוע dilation והחסרתו מהתמונה המקורית

באיור 8 ניתן לראות את זיהוי גבולות הטסיות בשתי הדרכים הנ"ל. ניתן לראות כי בשתי השיטות ניתן לזהות את טסיות הדם בצורה טובה.

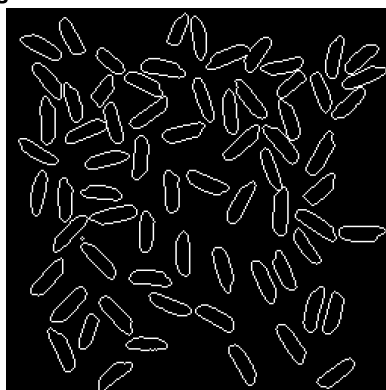
אופן היישום של שיטת LoG מתבססת על 2 מסננים מרחביים, מסנן לפלסיאן המדגיש שינויים בתדר גבוה (קצוות) אך רגיש לרעשים ומסנן גאוסייני אשר מפחית רעשים ומבצע החלקה של התמונה. יישום מסננים אלו קורה בתחום התדר, כאשר נחשב את גרעין מסנן ה-LoG באמצעות הכפלת גרעיני כל אחד מהמסננים הנ"ל וביצוע הפילטור על ידי כפל מטריצת המסנן בתמונה עצמה.

השיטה השנייה מתבססת על ביצוע dilation. בפעולה זו ישנה הרחבה של האובייקטים בתמונה באמצעות מבנה מוגדר, כאשר הרחבה זו גורמת לטסיות לגדול ולמלא חלל גדול יותר סביבם. בעת החסרת התוצאה מתמונת המקור, נקבל הדגשה של הפיקסלים הנמצאים רק באזורים שהורחבו, כלומר אלו שמהווים את גבולות הטסיות.

#### תשובה לשאלה 1.6:

נשערך את מספר הטסיות מתוך המסכה שיצרנו (איור 7). תחילה השתמשנו בפונקציית imclearborder על מנת להסיר מקצוות התמונה את הטסיות שאינן מופיעות בשלמותן. לאחר הסרת הטסיות התקבלה התמונה הבאה:

Clean Edge Detection - Number of Blood Platelets=60



איור 9: תמונת זיהוי הגבולות שהתקבלה לאחר שימוש בפונקציית imclearborder

באיור 9 ניתן לראות את תמונת זיהוי הגבולות לאחר שימוש בפונקציה הנ"ל. נשים לב כי בשונה מאיור 8, הטסיות שנמצאו בגבולות התמונה הוחסרו והקווים המתארת כל טסית הושלמו במלואן.

לאחר מכן, השתמשנו ב-bwlabel לקבלת מספר האלמנטים המצויים בתמונה. התקבל כי מספר הטסיות הינו 60.



בשלב השני של הניסוי, ביצענו מיצוי מאפיינים של טסיות הדם.

#### תשובה לשאלה 1.7 :

על מנת למצוא את הטסיות הנוטות ימינה בלבד, נשתמש בפונקציה regionprops ובעזרתה נחלק את מאפייני התמונה ובניהן אוריינטציה, תמונת הטסיות לאחר מילוי והמסגרת הקטנה ביותר שמכילה את האובייקט. נסנן את הטסיות לפי האוריינטציה כאשר נבחר את הזווית להיות גדולה או שווה ל-0.

יצרנו מטריצה אפסים שתשמש כרקע לאכלס בתוכה את תמונות הטסיות הנוטות ימינה. בעזרת המאפיין boundingbox, נמצא את אינדקסי הפינה השמאלית העליונה של כל טסית ונאכלס את תמונת הטסית המלאה, שהתקבלה בעזרת המאפיין filledimage, בתמונת הרקע שיצרנו על ידי הוספתה במקום הרצוי.

Only Right Oriented Blood Plateles - Mask



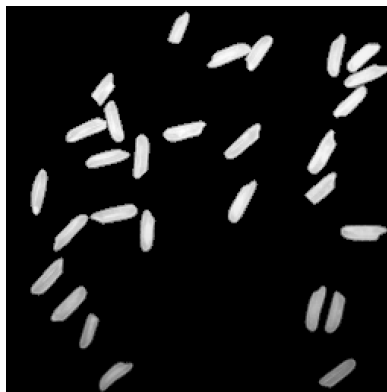
איור 10 : המסכה שהתקבלה לאחר ביצוע השלבים הנ"ל

באיור 10 ניתן לראות את המסכה שיצרנו עבור הטסיות הנוטות ימינה. נשים לב כי רוב המוחלט של הטסיות אכן נוטה ימינה חוץ מאחת אשר נוטה שמאלה. לא הצלחנו לבצע החסרה של טסית זו מכיוון שגבולה חופף עם גבול של טסית הנוטה ימינה. הפונקציה הנ"ל זיהתה את האובייקט כחיבור של שני הטסיות ולכן לא היה ניתן לבצע הפרדה ביניהם.

#### תשובה לשאלה 1.8 :

נציג את תוצאת הכפלת המסכה הנ"ל בתמונה המקורית :

Only Right Oriented Blood Plateles - After Masking



איור 11 : הטסיות הנוטות ימינה

באיור 11 ניתן לראות את תמונת הטסיות הנוטות ימינה בלבד. תוצאה זו התקבלה לאחר הכפלת המסכה (איור 10) והתמונה המקורית (איור 1). בכל מקום במסכה שלבן (ערך הפיקסל הוא 1) הפיקסל בתמונה המקורית יעבור במלואו בעוד ששאר הפיקסלים (שחור ערך 0) יתאפסו ויהיו שחורים.

#### 3.1.4 מסקנות

בניסוי זה התנסו בסינון תמונה של טסיות דם עם רקע משתנה ושימוש במאפיינים ליצירת מסיכה, במטרה ליצור תמונה המורכבת אך ורק מטסיות הנוטות ימינה (/). ניסוי זה בוצע בשני שלבים, סגמנטציה ומיצוי מאפיינים של טסיות הדם בתמונה.

בשלב הראשון של הניסוי, ביצענו סגמנטציה של טסיות הדם. בתחילה, יצירת המסכה התבצעה בעזרת יצירת היסטוגרמה עבור התמונה המקורית ובחירת סף מפריד בין רמות האפור של הרקע לטסיות. לאחר מכן חזרנו על שלבים אלו עבור התמונה המקורית לאחר החסרת הרקע ממנה. ראינו כי המסכה שהתקבלה עבור התמונה המקורית איכותית פחות בהשוואה לתמונה ללא הרקע. מכך הסקנו כי סינון הרקע משפר משמעותית את איכות המסכה. לבחירת מסכה נכונה ישנה חשיבות לביצוע סינון יעיל ולכן נעדיף להשתמש בשיטת ההחסרה לקבלת מסכה איכותית יותר. לאחר מכן, השתמשנו במסכה שיצרנו בשיטת ההחסרה לזיהוי גבולות הטסיות בשתי דרכים שונות; שימוש ב-LoG ושימוש ב-dilation של המסכה והחסרתה מהמסכה המקורית. ראינו כי שתי השיטות הניבו תוצאה טובה (איור 8). לבסוף, נעזרנו בפונקציה `imclearborder` להסרת הטסיות שאינן מופיעות בשלמותן מגבולות התמונה ובעזרת `bwlabel` התקבל כי מספר הטסיות המשוערכות בתמונה הוא 60.

בשלב השני של הניסוי, נעזרנו בחילוץ מאפיינים בעזרת `regionprops` למציאת הטסיות הנוטות ימינה בלבד, כאשר ביצענו את ההפרדה על בסיס אוריינטציה. ראינו כי התוצאות שהתקבלו אכן הניבו תוצאות טובות, כאשר התמונה הסופית שהתקבלה הכילה את כלל הטסיות הפונות ימינה וסיננה את הטסיות הפונות שמאלה, למעט טסית אחת. הופעתה של טסית בודדת זו הפונה שמאלה התרחשה בשל חפיפת שטח עם טסית הפונה ימינה הצמודה אליה וזיהוי שלהן כאובייקט יחיד. אנו סבורים כי מצב זה התרחש כתוצאה מפעולת ה-dilation שביצענו בעת יצירת המסכה. שיטת ה-dilation מתבססת על הרחבה של האובייקטים בתמונה באמצעות מבנה מוגדר, ייתכן כי בשל ההרחבה גבולות הטסיות השתנו באופן שהשפיע על הפרדת הטסיות. מכך למדנו על הצורך בשיפור שיטות הזיהוי והסגמנטציה על מנת למזער טעויות ולשפר את דיוק התוצאות.

## 3.2 ניסוי 2

### 3.2.1 היפותזה

בניסוי זה נבצע סינון של רעש 'salt&pepper' על ידי שימוש במסנן חציון בגדלים שונים ונבחן את השפעת גודל המסנן על סינון הרעש מהתמונה. נשער כי קיים גודל מסנן חציון אופטימלי עבור התמונה המורעשת אותה אנו מסננים, המביא למינימום את שגיאת ה-MSE. נצפה כי מסננים קטנים מדי לא יסננו את הרעש בצורה יעילה וכי מסננים גדולים יטשטשו את התמונה יותר מדי, זאת מכיוון שמסנן חציון מחליף כל פיקסל בערך החציון של הפיקסלים החופפים (בהתאם לגודל המסנן) בין התמונה למסנן. כמו כן, נצפה לראות הבדל ברור בערך המובהקות הסטטיסטית לאחר חזרה על הניסוי מספר רב של פעמים.

### 3.2.2 מתודולוגיה

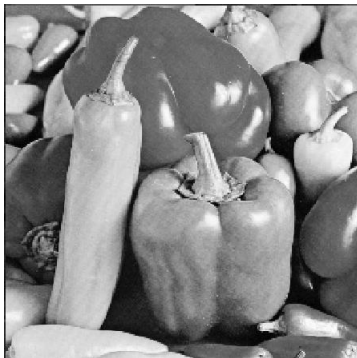
כלים וחומרים: תוכנת המטלב.

מהלך הניסוי: בניסוי זה נשתמש בתמונות המצורפות באתר הקורס, תמונת ירקות ותמונה בה תיבלו את הירקות במעט 'salt&pepper'. נשתמש בפונקציה שבנינו בדוח המכין על מנת לשפר את התמונה, כאשר נשתמש במסנן חציון בעל מימדים שונים (1x2 1x3 2x2 3x3 3x4 4x4 4x5 5x5 7x7) ועבור כל מסנן נחשב את ערך ה-MSE בין התמונה המקורית לתמונה לאחר הסינון. נבחן את ערכי השגיאה המתקבלים ונבחר מהו הגודל האופטימלי עבור מסנן זה. נחזור על הניסוי 100 פעמים, נציג ממוצע וסטיית תקן של השגיאה (MSE). נבחר מבין סטטיסטי על מנת לבחון את ההבדלים בין המסננים הריבועיים.

### 3.2.3 תוצאות

נציג את תמונות הירקות עם הרעש וללא:

Vegetables



Noisy Vegetables

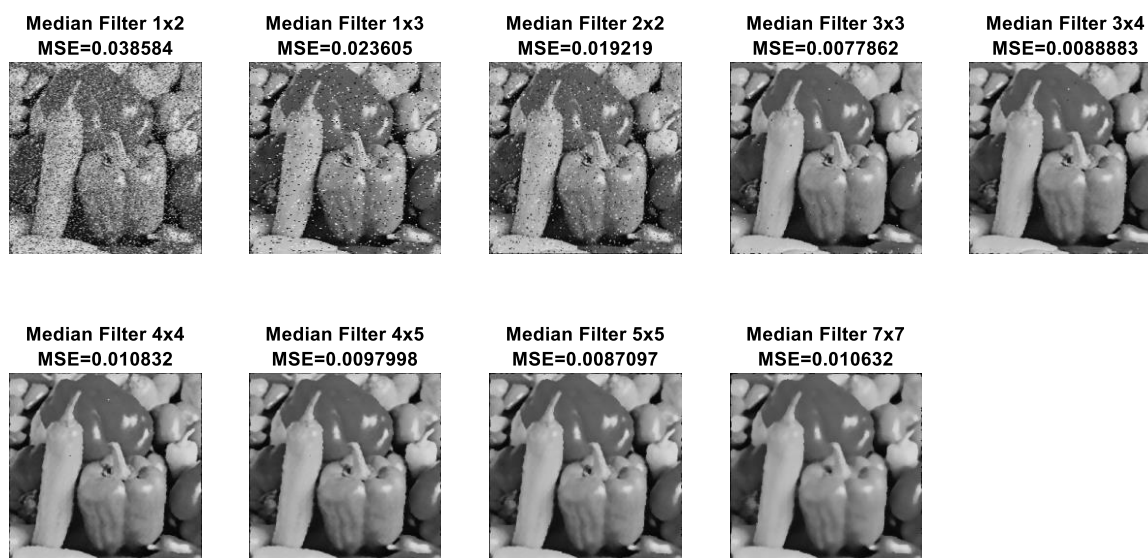


איור 12: תמונת הירקות הנקייה (שמאל) ועם הרעש המלח פלפל (ימין)

באיור 12 ניתן לראות את תמונת הירקות בשני מצבים; נקייה מרעש (מקורית) ותמונה מורעשת (רעש מלח פלפל).

## תשובה לשאלה 2.1 :

נשתמש בפונקציה שכתבנו בדוח המכין (CleanSP) על מנת לסנן את הרעש ולשפר את התמונה השמאלית בצורה המיטבית. השתמשנו במסנן חציון בגדלים שונים. נציג את התוצאות :



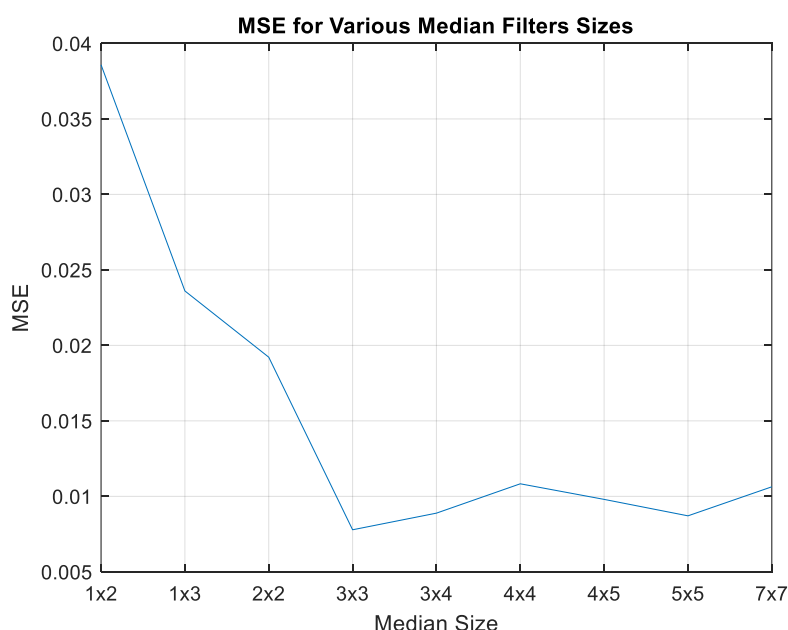
איור 13 : תוצאות הסינון בעזרת מסנני חציון בגדלים שונים

באיור 13 ניתן לראות את תוצאות הסינון בגדלי מסנן חציון שונים. נשים לב כי עבור מסננים בגודל קטן, הרעש לא סונן בצורה מיטבית ונראה לעין. ככל שגודל המסנן גדל, כך ניתן להבחין פחות ברעש, אולם בגדלי מסנן גדולים מאוד נצפה בטשטוש יתר של התמונה.

לאחר מכן, חישבנו את ערך ה-MSE בין התמונה המקורית לתמונה אחרי הסינון [1].

$$(1) \quad MSE = \frac{1}{n} \sum_i (y_i - \hat{y}_i)^2, \quad 0 \leq i \leq n - 1$$

להלן התוצאות המתקבלות :



איור 14 : ערכי ה-MSE המתקבלים עבור מסננים בגדלים שונים

באיור 14 ניתן לראות את ערכי ה-MSE שהתקבלו בשימוש במסנן חציון בגדלים שונים. ניתן לראות כי השגיאה המינימלית מתקבלת עבור מסנן בגודל  $3 \times 3$ , כאשר ניתן להבחין כי גרף ה-MSE הינו במגמת ירידה עד לנקודה זו ולאחר מכן ישנה עלייה בערכי ה-MSE המתקבלים, אולם לא באופן משמעותי. כלומר לפי גודל השגיאה המינימלי ולפי הסתכלות על מוצא המסננים (איור 13), כך שמוצא המסנן אינו מטושטש למדי ועוצמת הרעש נמוכה מאוד, גודל המסנן האופטימלי לפי קריטריון MSE הינו  $3 \times 3$ .

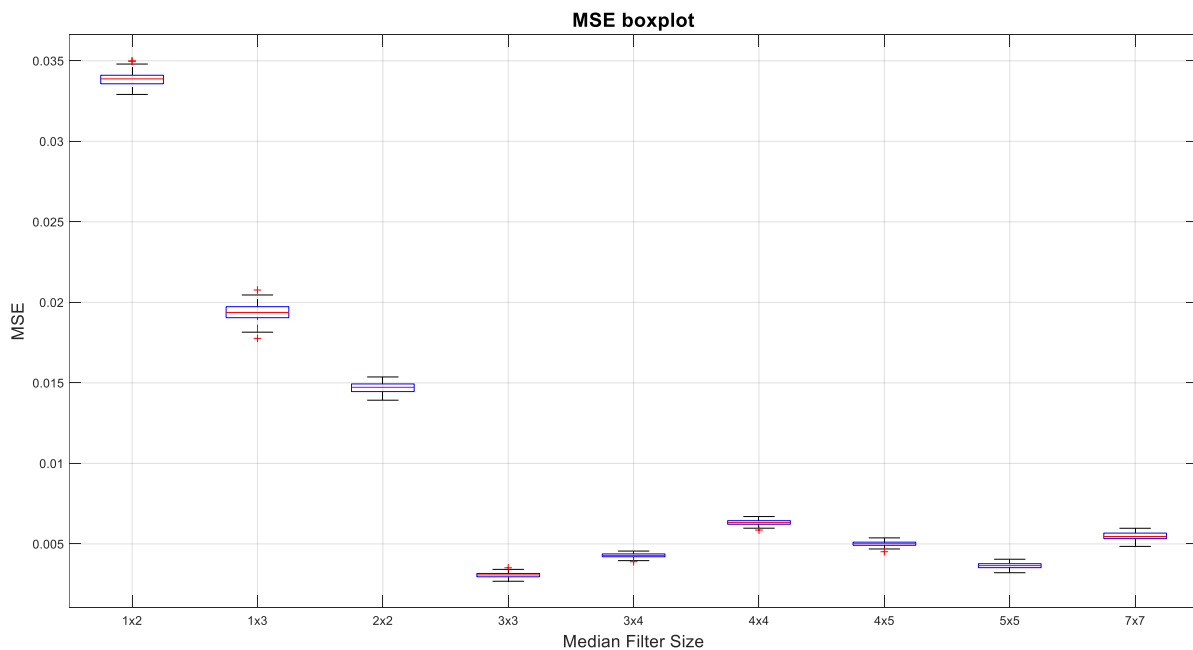
## תשובה לשאלה 2.2:

בעזרת הפונקציה `imnoise` יצרנו רעש מלאכותי אקראי של רעש מלח פלפל כאשר רמת הרעש היא 20% וחזרנו על החלק הקודם של הסינון 100 פעמים. נציג את הממוצע וסטיית התקן:

טבלה 1: ערכי ממוצע וסטיית התקן של ערכי ה-MSE

גודל מסנן	ממוצע MSE	סטיית תקן MSE
1x2	0.0338	$4.01 \times 10^{-4}$
1x3	0.0193	$5.06 \times 10^{-4}$
2x2	0.0147	$2.89 \times 10^{-4}$
3x3	0.0030	$1.41 \times 10^{-4}$
3x4	0.0043	$1.17 \times 10^{-4}$
4x4	0.0064	$1.49 \times 10^{-4}$
4x5	0.0050	$1.36 \times 10^{-4}$
5x5	0.0037	$1.63 \times 10^{-4}$
7x7	0.0055	$2.20 \times 10^{-4}$

בטבלה 1 ניתן לראות את ערכי הממוצע וסטיית התקן של ערכי ה-MSE לאחר חזרה על הניסוי 100 פעמים.



איור 15: גרף boxplot של ערכי ממוצע וסטיית תקן של ערכי ה-MSE עבור גודלי המסנן השונים

באיור 15 ניתן לראות את גרף ה-Boxplot המתקבל לאחר הוספת הרעש וסינונו 100 פעמים. מהתבוננות בגרף זה ובטבלה 1 ניתן להבחין באותה מגמה של איור 14, כלומר ביצוע החזרות לא השפיע על סדר טיב הסינון לפי קריטריון MSE. סטיית התקן (המופיעה בגרף כתחום פסים שחורים) עבור מסננים בגודל קטן (עד גודל  $2 \times 2$ ) גדולה יחסית בעוד שסטיית התקן של המסננים בגודל גדול יותר קטנה. אין חפיפה משמעותית בין תחומי סטיית התקן כלומר, נוכל להבדיל סטטיסטית בין המסננים בגדלים השונים. נשים לב כי ישנם מספר outliers, המסומנים בפלוס אדום, אך הם בודדים וקרובים מאוד לטווח סטיית התקן, כלומר אינם ישפיעו בצורה רבה על תוצאות הניסוי.

ביצענו מבחן סטטיסטי מסוג one-way ANOVA על מנת לעמוד על ההבדלים בין המסננים הריבועיים. קבענו את השערת האפס להיות כך שאין הבדלים סטטיסטיים משמעותיים בין סינון המסננים בגדלים הריבועיים השונים. נציג את תוצאות מבחן זה:

{'Source' }	{'SS' }	{'df' }	{'MS' }	{'F' }	{'Prob>F' }
{'Columns'}	{[ 0.0087]}	{[ 4]}	{[ 0.0022]}	{[5.4416e+04]}	{[ 0]}
{'Error' }	{[1.9892e-05]}	{[495]}	{[4.0185e-08]}	{0x0 double }	{0x0 double }
{'Total' }	{[ 0.0088]}	{[499]}	{0x0 double }	{0x0 double }	{0x0 double }

p-Value=0

איור 16: תוצאות מבחן one-way ANOVA ו-p-Value

באיור 16 ניתן לראות את תוצאות המבחן הסטטיסטי. מהטבלה ניתן לראות כי המבחן מראות כי יש הבדל מובהק בין הקבוצות השונות. ערך ה-F-statistic המתקבל גדול מאוד, מסדר גודל של  $10^4$ , וערך p-Value הקטן מ-0.05 (שגיאת  $\alpha$ ).

### 3.2.4 מסקנות

בניסוי זה בחנו את השפעת גודל גרעין מסוג חציון על סינון רעש מלח פלפל מתוך תמונה. לצורך כך השתמשנו בפונקציה CleanSP ליצירת מסנני חציון בגדלים שונים, הפעלנו את המסננים השונים לסינון הרעש ובחנו את השפעתם על התמונה לאחר סינון, תוך חישוב שגיאת ה-MSE בין התמונה המקורית לתמונה לאחר סינון. ראינו כי באופן ויזואלי, ככל שהשתמשנו במסנן גדול יותר, כך הרעש פחת והתמונה הפכה למטושטשת יותר. מהתוצאות שהתקבלו (איור 14) הסקנו כי במקרה זה המסנן בגודל  $3 \times 3$  הניב את התוצאה הטובה ביותר, כלומר את היחס הטוב ביותר בין סינון הרעש לשמירה על חדות התמונה. בביצוע 100 חזרות על הניסוי ראינו כי נצפתה עקביות בתוצאות, כאשר ממוצע התוצאות שהתקבלו היה קרוב לערך ה-MSE של הניסוי המקורי, עם פיזור קטן ושמירה על מגמת הגרף. בנוסף, ביצענו מבחן ANOVA להשוואת ערכי ה-MSE של מסננים חציוניים ריבועיים [2]. מבחן זה הראה כי קיים הבדל מובהק סטטיסטי בין תוצאות סינון רעש מלח פלפל על ידי מסננים חציוניים ריבועיים בגדלים שונים. מהתוצאות שהתקבלו ניתן להסיק כי שימוש במסנן חציון בגודל  $3 \times 3$  הינו האופטימלי לסינון רעש מלח פלפל מהתמונה הנתונה. אולם, לא נוכל להסיק כי מסנן זה הינו האופטימלי לכלל המקרים. ניסוי זה בחן רק תמונה אחת ורמת רעש אחת, כלומר ייתכן ויתקבלו תוצאות שונות עבור תמונות שונות ורמות רעש אחרות. כמו כן, בחנו שימוש במסנן חציון ללא השוואה למסננים אחרים לסינון הרעש.

### 3.3 ניסוי 3

#### 3.3.1 היפותזה

בתרגיל זה כתבנו מספר פונקציות לצורך ניתוח סרטון אשר נשתמש בהם על מנת לזהות את מיקומי העיניים המביטות למספר מיקומים שונים. בניסוי זה ננתח את סרטון הוידאו שצילמנו של תנועת האישון, כאשר נרצה לגלות את המספר עליו מסתכלות העיניים באותו פריים. המטרה הייתה לכתוב אלגוריתם, שיזהה על סמך בסיס תמונות כיול, לאיזה כיוון האדם בסרטון מסתכל. נצפה כי לא נצליח לקבל זיהוי מושלם של כל הפריימים בסרטון, הנובעות מחוסר אידיאליות בעת ביצוע ניסוי אמפירי מסוג זה, כגון שינוי במיקום ראש הנבדק בין תמונות הכיול לסרטון, מצמוצים ושגיאות הן בביצוע הכיול והן בביצועי הנבדק, רעש חיצוני שמשפיע על התמונה וכיול על בסיס מספר תמונות קטן יחסית.

#### 3.3.2 מתודולוגיה

כלים וחומרים: סרטון וידאו ותמונות, תוכנת המטלב.

מהלך הניסוי: ראשית ווידאנו שהפלורסנטים בחדר כבויים על מנת למנוע רעש ב-[Hz] 50. לאחר מכן קיבענו את צווארו של הנבדק לשולחן הניסוי ומיקמנו את המצלמה מול עיניו. פתחנו את קובץ המטלב ומקסמנו את איכות התמונה בעזרת כיוונון פרמטרי המצלמה דרך ה-GUI. לאחר מכן צילמנו 4 תמונות כיול אשר בכל אחת מהן העיניים מביטות אל מספר אחר מ-1 עד 4, חזרנו על פעולה זו 12 פעמים. בתמונות אלו נשתמש ליצירת מטריצת כיול למיקומי העיניים. לבסוף, צילמנו וידאו של 30 שניות כאשר בכל 2 שניות מיקום העין משתנה לפי הסדר הבא:  $1 \rightarrow 2 \rightarrow 3 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 3 \rightarrow 2 \rightarrow 1 \rightarrow 2 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 1 \rightarrow 2$ .

#### 3.3.3 תוצאות

תשובה לשאלה 3.1:

נציג את הפונקציה שכתבנו המחזירה את מיקום כל עין בפריים שנשלח אליה:

```
function [Eye_Pos]=EyePosition_FUNC(Frame)
% [Eye_Pos]=EyePosition_FUNC(Frame)
%
% This function returns the eyes position from the input frame
%
% Inputs:
% Frame      - Desired frame for extraction eye position. Matrix of
%              normalized gray scale image
%
% Outputs:
% Eye_Pos    - The eye position of both of the eyes. Returns as a matrix 2x2
%              where the first row is the Left eye and second row is the Right
%              eye. in each row there is the coordinates (x and y indexes of
%              the image)

% Assining print variable. Change to 1 for plotting image
print=0;

% Filtering the desired frame with median filter of size 45x45
adjFrame=CleanSP(Frame,'Median',45,45);
% Printing image
if print
    figure('WindowState','maximized')
    nexttile
    imshow(adjFrame)
    title('Filtered Image for Detection Eye Position')
end

% Finding circles
min_rad=20;
max_rad=30;
```

```

centers=imfindcircles(adjFrame,[min_rad max_rad],'ObjectPolarity','dark' ...
    , 'EdgeThreshold',0.01,'Method','TwoStage');
% Rounding for indices
Eye_Pos=round(centers);

% Sorting the circle indices according to their gray scale value
[~,ind]=sort(sum(adjFrame(Eye_Pos(:,2),Eye_Pos(:,1)).*eye(length(Eye_Pos))), 'ascen
d');
% Extracing only the blackest indices
Eye_Pos=Eye_Pos(ind(1:2),:);

% Sorting the rows as left in first row and right in second
[~,I]=sort(Eye_Pos, 'descend');
Eye_Pos=Eye_Pos(I(:,1),:);
end

```

נציג תוצאה אחת של מיקומי העיניים :



איור 17 : דוגמת תוצאת הפונקציה עבור מיקום העיניים

באיור 17 ניתן לראות דוגמא אחת למוצא הפונקציה עבור זיהוי מיקום העיניים. נשים לב כי התמונה זיהתה את מיקום האישונים בצורה טובה, כאשר הכוכבית האדומה נמצאת על מרכז האישון והעיגול הכחול הוא על מנת להראות את כל מיקום העין. חתכנו את התמונה לפני הכנסתה לפונקציה על מנת למקד את הפונקציה רק לאזור העיניים ולהפחית את כל רעשי הרקע למנוע זיהוי שגוי.

תשובה לשאלה 3.2 :

הפונקציה הנ"ל מוציאה את מיקום כל עין בפריים שנשלח אליה בצורה הבאה :



איור 18 : דיאגרמת בלוקים של אלגוריתם הפונקציה EyePosition\_FUNC



באיור 18 ניתן לראות את דיאגרמת הבלוקים של האלגוריתם של הפונקציה הנ"ל. תחילה אנו מסננים את התמונה במסנן חציון בגודל  $45 \times 45$ . סינון זה מבצע טשטוש של התמונה ויסנן רעשים. דבר זה יגביר את זיהוי המיקום של האישון מכיוון שמיקום העיניים שיתקבל יהיו עיגולים בולטים יותר וכהים יותר משאר חלקי התמונה. לאחר מכן נקבע את רדיוס העיניים להיות בין 20 ל-30 פיקסלים. את הממדים האלו מדדנו בצורה ידנית על ידי פונקציית `imdistline`. לאחר מכן, השתמשנו בפונקציית `imfindcircles` על מנת למצוא את כל העיגולים הכהים בתמונה. מוצא הפונקציה הינו המיקום בתמונה המקורית ולכן על מנת להפוך אותו לאינדקסים, נבצע עיגול כלפי מעלה [3]. קיבלנו מספר רב יחסית של אינדקסים המבטאים עיגולים כהים בתמונה, ולכן על מנת למצוא אך ורק את מיקומי העיניים, נמצא את המיקומים שהתקבלו בהם גווני האפור הינם הכהים ביותר. נחלץ את שני הערכים הכהים ביותר ונסדר את מטריצת המוצא להיות כך שתתאים לערך המצופה. השורה הראשונה (עין שמאל של הנבדק) תהיה בעלת ערך אינדקס ה-X הגבוה יותר והשורה השנייה (עין ימין של הנבדק) תהיה בעלת ערך ה-X הקטן יותר.

### תשובה לשאלה 3.3 :

כתבנו את הפונקציה המחזירה מטריצת כיוול עבור מערך התאים שנשלח אליה. קלט מערך התאים שיצרנו הינו מערך תאים בגודל  $1 \times 4$  אשר כל תא מייצג את המספר עליו הנבדק הסתכל (1,2,3,4). בכל תא נמצא מערך נוסף של תאים בגודל  $1 \times 12$  כאשר בכל תא ישנה מטריצה המייצגת כל פריים חתוך של מיקום העיניים המסתכל על הערך המספרי בהתאמה. נציג את הפונקציה שכתבנו :

```
function [EyeCalib] = EyeCalibration_FUNC(CalibFrame)
% [EyeCalib] = EyeCalibration_FUNC(CalibFrame)
%
% This function returns the eyes position calibration matrix from the input
%
% Inputs:
% CalibFrame      - Cell array 4x1 in which each cell is 1x12 cell array of
%                  cropped frames
%
% Outputs:
% EyeCalib        - The eye position calibration matrix of both of the eyes.
%                  Returns as a matrix (N-1)x8x2 where N is the number of
%                  calibration frames, 8 is the number of coordinates in each
%                  position (2 [coordinates]*4 [positions]) and 2 is the
%                  dimention of the eyes (1 for each eye for calibration).

% Assining print variable. Change to 1 for plotting image
print=0;
% Assigning variables and Preallocating memory
N=length(CalibFrame{1});
left=1;
right=2;
EyeCalib=nan(N-1,8,2);
for i=1:4
    for j=1:N-1
        % Extracting frames
        Frame=CalibFrame{i}{j};
        % Finding eye position through our position function
        Eye_Pos=EyePosition_FUNC(Frame);
        % Allocating the eye position to our output
        EyeCalib(j,[2*i-1 2*i],left)=Eye_Pos(left,:);
        EyeCalib(j,[2*i-1 2*i],right)=Eye_Pos(right,:);
        % Printing results
        if print
            nexttile
            imshow(Frame)
            hold on
            viscircles(Eye_Pos,28,'EdgeColor','b');
            plot(Eye_Pos(1,1),Eye_Pos(1,2),'r*')
```

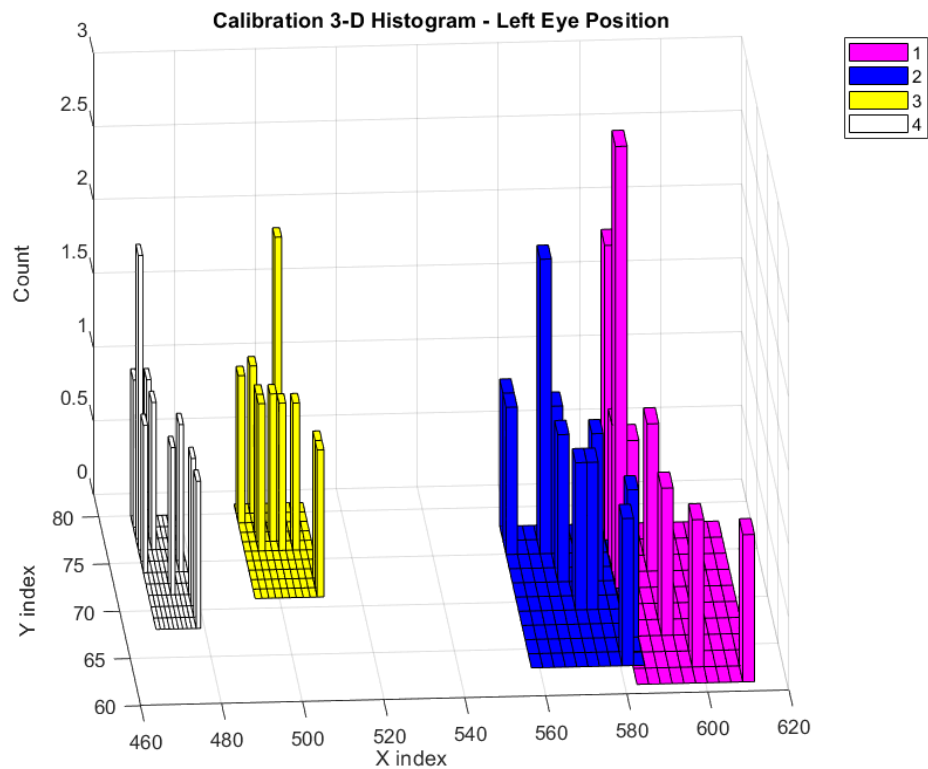
```

        plot(Eye_Pos(2,1),Eye_Pos(2,2), 'r*')
    end
end
end
end

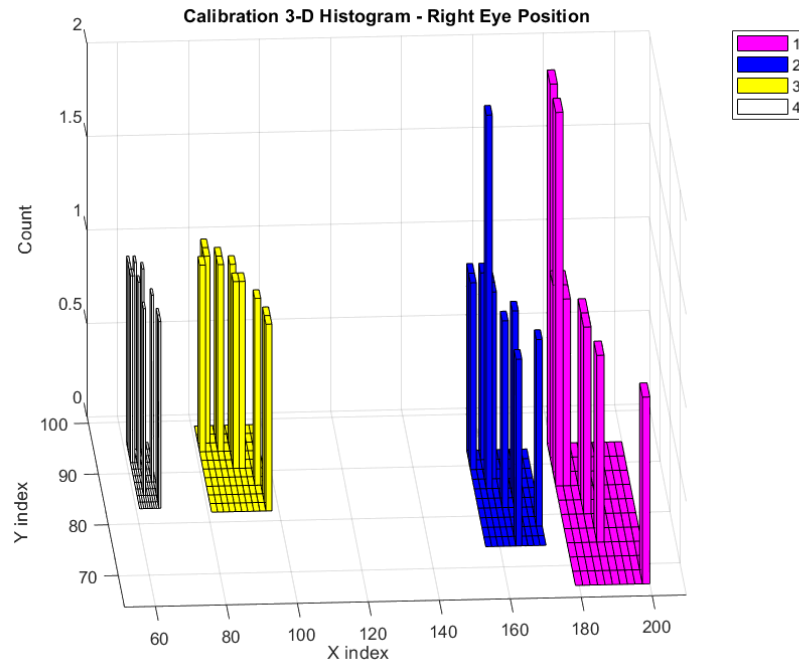
```

פונקציה זו מקבלת את מערך התאים כפי שהסברנו לעיל ומוציאה את מטריצת הכיול כמטריצה תלת ממדית. המימד השלישי הינו מימד מסדר 2 כאשר בכל אחד מהן תהיה מטריצת כיול לעין אחת. המטריצת כיול תסדר את אינדקסי מיקומי העיניים, שהתקבלו כתוצאה מהפעלת הפונקציה הקודמת, בהתאמה למספר עליו הנבדק הסתכל. בנוסף, נבחן רק  $N-1$  תמונות פריימים (במקרה שלנו 11 מתוך 12 תמונות פריימים) עבור כל הסתכלות של הנבדק על מספר ספציפי ליישום מטריצת הכיול.

נציג היסטוגרמות תלת מימדיות של מטריצות הכיול עבור 2 העיניים :



איור 19: היסטוגרמת כיול עין שמאל



איור 20: היסטוגרמת כיול עין ימין

באיור 19 ואיור 20 ניתן לראות את היסטוגרמות מטריצות הכיול עבור 2 העיניים, שמאל וימין בהתאמה. תחילה נשים לב כי ישנה הפרדה רחבה מאוד בין מיקומי 1 ו-2 לבין 3 ו-4 במיקומי אינדקסי ציר ה-X בעוד שמיקומי ציר ה-Y בעין ימין ועין שמאל אינם מופרדים כמעט כלל בכל פוזיציה של מיקום העיניים. כמו כן, נבחין כי בין מיקומי 3 ו-4 ישנה הפרדה מספקת לחלוטין באינדקסי ציר ה-X בשתי העיניים בעוד שבין מיקומי 1 ו-2 ההפרדה היא קטנה ואפילו חופפת במקצת במקרה של עין שמאל. יתר על כן, נשים לב כי בכל מקבץ של מיקום, התפלגות מיקום העין הינה רחבה בציר ה-Y בעוד שבציר ה-X הינה צרה. בנוסף, בכל מקבץ ישנו פיק הגדול מ-1 המסמל כי העין חזרה בערך לאותו מיקום בציר ה-X וה-Y.

תשובה לשאלה 3.4 :

נציג את הפונקציה שכתבנו המקבלת פריים בודד ומטריצת כיול ומחזירה את המספר עליו מסתכלות העיניים :

```
function [Eye_Look] = EyeLook_FUNC(Frame, EyeCalib)
% [Eye_Look] = EyeLook_FUNC(Frame, EyeCalib)
%
% This function returns the eyes position number from a single frame and a
% specific calibration matrix
%
% Inputs:
% Frame          - Desired frame for extraction eye position. Matrix of
%                  normalized gray scale image
% EyeCalib       - The eye position calibration matrix of both of the eyes.
%                  Matrix of (N-1)x8x2 where N is the number of calibration
%                  frames, 8 is the number of coordinates in each position
%                  (2 [coordinates]*4 [positions]) and 2 is the dimention
%                  of the eyes (1 for each eye for calibration).
%
% Outputs:
% Eye_Look       - A matrix of 1x2 housing the number the left and right eye
%                  has looked on
%
% Assigning print variable. Change to 1 for plotting image
```

```

print=1;
% Left and right indices
Left=1;
Right=2;
% Eye position calculation
Eye_Pos=EyePosition_FUNC(Frame);

% Calculating the mean of each index from the calibration matrix
LeftMean=mean(EyeCalib(:, :, Left));
RightMean=mean(EyeCalib(:, :, Right));
% Preallocation memory and loop for calculating the distance between the
% current eye position from the mean position for each state (number)
[LeftError, RightError]=deal(zeros(1,4));
for i=1:4
    LeftError(i)=vecnorm(Eye_Pos(Left, :)-LeftMean([2*i-1 2*i]));
    RightError(i)=vecnorm(Eye_Pos(Right, :)-RightMean([2*i-1 2*i]));
end
% Finding the minimum distance index
 [~,LeftInd]=min(LeftError);
 [~,RightInd]=min(RightError);
% Returning the number each eye was looking on
Eye_Look=[LeftInd, RightInd];
end

```

#### תשובה לשאלה 3.4.1 :

פונקציה זו מקבלת פריים ספציפי ואת מטריצת הכיול שיצרנו סעיף קודם ומחזירה את המספר עליו כל עין הסתכלה. את המיקום נקבל באמצעות ניתוח סטטיסטי. הניתוח הסטטיסטי הנבחר מבוסס על חישוב מרחק אוקלידי בין ממוצעי הקוארדינטות (x,y) לבין מיקום העין שהתקבל בפריים הנבחר. המרחק האוקלידי מחושב כך [4]:

$$(2) \quad d(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2}$$

כאשר  $p$  ו- $q$  הינן נקודות במרחב דו מימדי  $p = (p_1, p_2)$ ,  $q = (q_1, q_2)$  כאשר כל קורדינטה היא של ציר ה-x וה-y בהתאמה. בחירת הנקודה למדידת מרחק להיות  $p$  או  $q$  הינה שרירותית, שכן ההעלאה החיסור בריבוע אינה משנה את סדר קביעת הפרמטרים.

בחירת ניתוח זה היה בשל פשטותו ויעילותו. כלומר, חישובו פשוט ואינו דורש הנחות סטטיסטיות מיוחדות. ביישום ניתוח זה נוכל בדרך פשוטה יחסית לבחור את המספר בעל ההסתברות הגבוהה ביותר עליו הסתכל המשתמש, כיוון שככל שהמרחק האוקלידי בין מיקום העין שהתקבל לבין ממוצעי הקוארדינטות שהתקבלו בכיול קטן יותר כך גדל הסיכוי שבחירת כיוון ההסתכלות יהיה מדויק.

#### תשובה לשאלה 3.4.2 :

בחנו את הפונקציה שיצרנו באמצעות 4 פריימים שונים אשר כל אחד מסתכל על מספר שונה. הפריימים הללו לא שומשו ביצירת מטריצת הכיול. נציג את התוצאות המתקבלות עבור כל עין :

טבלה 2 : השוואה בין זיהוי מיקום הסתכלות העיניים ולאזור ההסתכלות הידוע

אזור הסתכלות	עין ימין	עין שמאל
1	1	2
2	2	2
3	3	3
4	4	4

בטבלה 2 ניתן לראות את תוצאות הפונקציה שהתקבלו מארבעת הפריימים הנ"ל. נשים לב כי עבור אזור הסתכלות 2,3 ו-4 הפונקציה מצאה את המספר הנכון עבור 2 העיניים בהצלחה. עבור אזור הסתכלות 1 קיבלנו הבדל בין עין שמאל לעין ימין. עין שמאל זיהתה אזור שגוי אך סמוך בעוד שעין ימין זיהתה את האזור בהצלחה.

### תשובה לשאלה 3.6 :

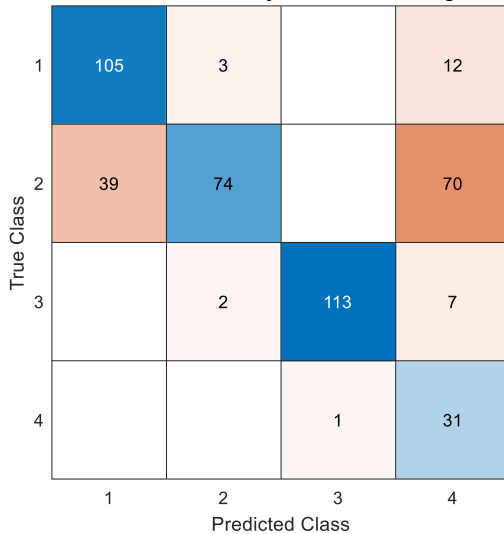
חישבנו את אחוז השגיאות עבור כל עין :

טבלה 3 : אחוז שגיאות סיווג כל עין

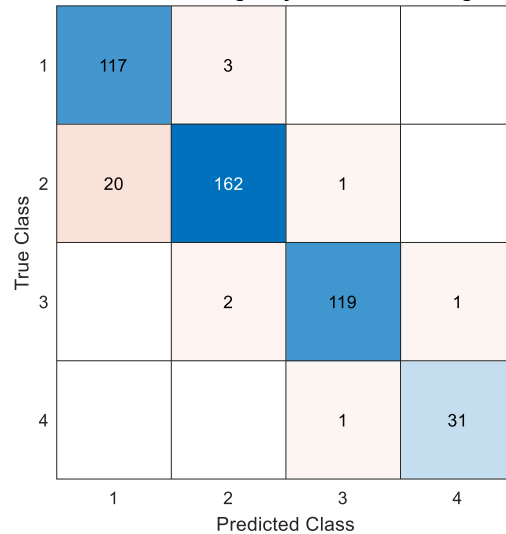
שגיאה עין שמאל	שגיאה עין ימין
29.32%	6.13%

בטבלה 3 ניתן לראות את אחוז השגיאות עבור כל עין. נשים לב כי עבור עין ימין התקבל אחוז שגיאות נמוך מאוד בעוד שעבור עין שמאל התקבל אחוז שגיאות גבוה מאוד יחסית לעין ימין. על מנת לבחון את התוצאות הללו נציג את התוצאות בצורת מטריצת מבוכה :

Confusion Matrix for Left Eye - Error Percentage=29.3217



Confusion Matrix for Right Eye - Error Percentage=6.1269



איור 21 : מטריצות המבוכה עבור 2 העיניים

באיור 21 ניתן לראות את מטריצות המבוכה אשר התקבלו עבור סיווג 2 העיניים. נשים לב כי עבור עין ימין (ימין) קיבלנו זיהוי גבוה מאוד של ה-true classes במיוחד ב-classes 1,3,4. עבור class 2 קיבלנו דיוק גבוה אך עם הרבה זיהויים שגויים. האלגוריתם זיהה את class 2 כ-class 1 הרבה פעמים יחסית לשאר ה-classes. לעומת זאת, עבור עין שמאל (שמאל) קיבלנו הרבה זיהויים שגויים. עבור class 2, האלגוריתם זיהה אותו רבות כ-class 1 או class 4. בנוסף class 1 זוהה גם כ-class 4 מספר פעמים. דבר זה הגדיל לנו את השגיאה של עין זו בצורה רבה.

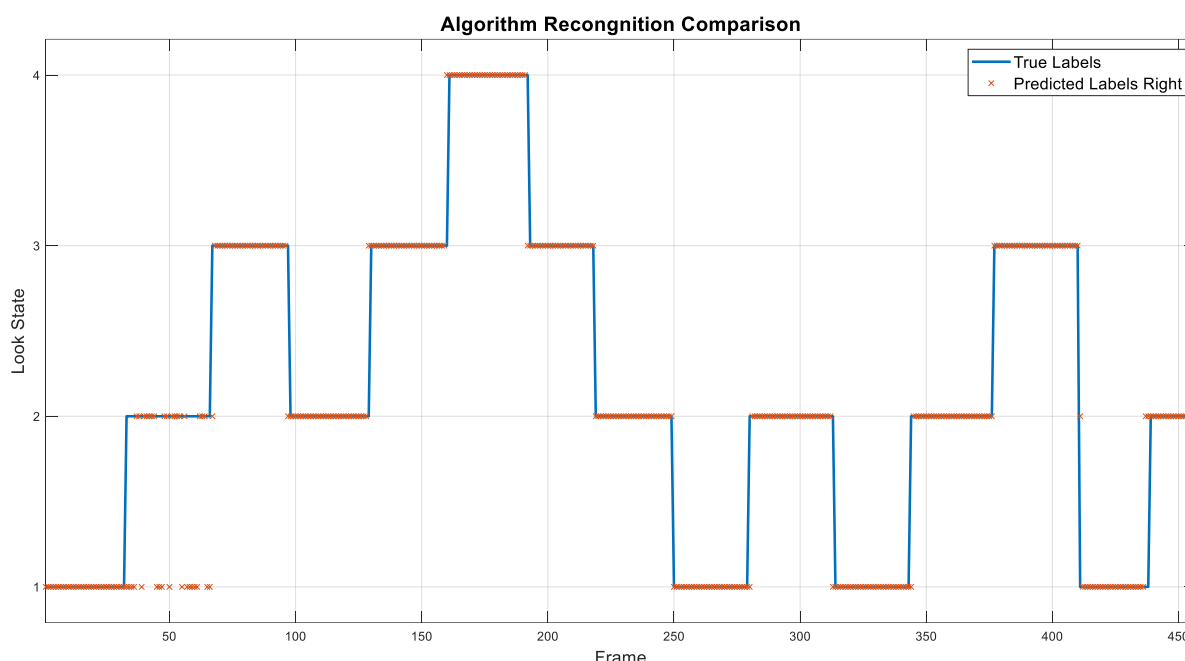
Failed Detection of Left Eye From a Video Frame - Class 2



איור 22 : דוגמא למציאת מיקום העיניים עבור פריים בודד הנמצא במיקום 2

באיור 22 ניתן לראות דוגמא כושלת למציאת מיקום עין שמאל באחד מהפריימים של class 2. נשים לב כי מיקום עין שמאל זיהה את מיקום המיקום הלא נכון, מיקום שהוא חלק מעין ימין. דבר זה גורם לכך שערך אינדקס ציר ה-x היה רחוק למדי (קטן מאוד) מערכי מטריצת הכיול עבור העין הימין (איור 19), כלומר המרחק המינימלי מממוצע הקבוצה הקרובה ביותר לעין ימין יוביל לכך שהפריימים מסוג זה יסווגו כקבוצה 4.

נציג גרף המתאר את תוצאות האלגוריתם עבור עין ימין אל מול ה-true labels.



איור 23 : השוואה בין תוצאות האלגוריתם עבור עין ימין לערכים האמיתיים

באיור 23 נראה את השוואת תוצאות האלגוריתם (מיוצגות כאיקס כתום) לבין הערכים האמיתיים (מיוצגים כגרף רציף כחול). נשים לב כי תוצאות האלגוריתם תואמות לערכים האמיתיים ברוב מהמקרים כאשר ישנן 2 מקרים בהן יש שגיאות; בעת מעברים בין מספר למספר ב-true labels ובחלקו הראשון של הניסוי בעיקר סביב פריים 50.

### 3.3.4 מסקנות

בניסוי זה ביצענו סיווג של כיווני הסתכלות שונים. לשם כך השתמשנו ב-11 תמונות כיול עבור כל כיוון, מציאת מקום העיניים בכל כיוון ויצירת מטריצת כיול לכל עין. לבסוף, השתמשנו במטריצת הכיול שבנינו והתאמות ידניות לצורך סיווג כיווני הסתכלות של הנבדק בסרטון וידאו. בניתוח תמונות הכיול, בהצגת היסטוגרמות פריסת האזורים לכל כיוון ראינו כי קיימת הפרדה ברורה יותר בין כיווני ההסתכלות עבור עין ימין (איור 20), לכן נסיק כי הסיווג בעזרת עין ימין יהיה איכותי יותר.

בחנו את טיב הפונקציה שיצרנו בעזרת מוצא הפריים שלא נכלל (הפריים ה-12 עבור כל עין). ראינו כי עבור עין ימין הדיוק שהתקבל הינו 100%, כלומר זיהוי ארבעת הכיוונים באופן מדויק, ואילו עבור עין שמאל הדיוק שהתקבל הינו 75%, כאשר התקבלה שגיאת סיווג של אזור הסתכלות 1 כאזור הסתכלות 2. הסקנו כי סביר ששגיאה זו נובעת בשל אזור החפיפה הקיים בין אזורים סמוכים אלו עבור עין שמאל (איור 19).

בניתוח הסרטון קיבלנו אחוזי הצלחה עבור 2 העיניים; 93.87% עבור עין ימין ו-70.68% עבור עין שמאל. ניתן להסיק כי ההבדל באחוזי ההצלחה בין 2 העיניים נובע מ-2 סיבות; קבוצות התפלגות אינדקסי מיקומי העיניים בתמונות הכיול היו מופרדות יותר לפי ציר ה-x בעין ימין מאשר בעין שמאל וזיהוי העיניים בעין שמאל עבור סיווג class 4 אינו היה מדויק. על פי איור 22 ניתן לראות כי מיקום עין שמאל עבור פריים בודד מ-class 2 סווג כ-class 4 עקב כשל בזיהוי מיקום עין שמאל. נסיק בעזרת איור 21 כי דבר זה קרה מספר רב של פעמים עקב כך שישנם הרבה סיווגים שגויים ל-class 4 מכל שאר ה-classes (במיוחד מ-class 2). ייתכן כי חוסר הדיוק בזיהוי מיקום עין שמאל נובע ממשתנים כגון תאורה אחרת בשל מנח הפנים של הנבדק במהלך צילום הסרטון.

הסיבות הנ"ל פגעו רבות באחוז ההצלחה של זיהוי עין שמאל, אך עין ימין זוהתה באחוזי הצלחה גבוהים מאוד אולם לא מושלמים. ניתן להסיק כי ישנן מספר סיבות אפשריות לשגיאות הסיווג הכלליות שהתקבלו; ראשית, מיקום הנבדק בצילום הוידאו השתנה ביחס למיקומו בתמונות הכיול. ניסינו לתקן היסט זה באמצעות חיתוך התמונה ושינוי הגודל על מנת לקבל את מיקום הנבדק הזהה ביותר למיקום הנבדק בתמונות הכיול ככל הניתן. אולם גם לאחר ביצוע התאמות בצורה ידנית, נסיק כי התיקון אינו

מדויק מספיק. כמו כן, הבחנו כי מרבית השגיאות הינן בחלקו הראשון של הסרטון, בסיווג שגוי של אזור הסתכלות מספר 2- השכיח מבין אזורי ההסתכלות בסרטון, כאזור הסתכלות מספר 1- אזור הסמוך לאזור הסתכלות 2 (איור 23). נסיק כי באזור זה היו הפרעות הנובעות ממנח הנבדק כתזוזה, מצמוץ או רעש חיצוני רב שפגעו ביכולת הזיהוי במקטע ספציפי זה. למרות זאת, קיבלנו אחוזי הצלחה גבוהים מאוד לכן ניתן להסיק כי נוכל להסתמך בעיקר על עין ימין לצורך זיהוי הסתכלות הנבדק.

בנוסף, בניגוד לציפיותינו, כי נראה שגיאות בולטות לעין בעת מעברים בין כיווני הסתכלות שונים בשל טשטוש התמונה וקושי בזיהוי, ראינו כי ישנן שגיאות מועטות בחלקים אלו (איור 23).

## 3.4 ניסוי 4

### 3.4.1 היפותזה

בניסוי זה ננתח את תנועת רגל ימין במהלך הליכה במהירות של 2 קמ"ש, תוך שימוש במרקרים עיגוליים לבנים על רקע שחור על מנת להקל על ביצוע הניתוח. נצפה כי נצליח לזהות את המרקרים אך לא באופן מדויק, זאת מכיוון שהפריימים המנותחים הינם מתוך סרטון שאינו סטטי, כלומר כאשר הנבדק בתנועה הפריים המתקבל עשוי להיות מטושטש ולמנוע זיהוי מיטבי של המרקרים. לכן בפריימים המטושטשים נצפה לראות יותר שגיאות. כמו כן, גורמים כמו הרקע וגוון הלבוש של הנבדק ישפיעו על טיב הזיהוי.

בבית עיבדנו את הקובץ בכמה דרכים: תחילה ביצענו עיבוד מקדים לפריימים הכולל הפיכת הפריים לגווני אפור, יצירת תמונה בינארית וחיתוך של האזור הרלוונטי (ROI). לאחר מכן מצאנו את מיקומי המרקרים בפריימים השונים וחישובנו את המהירות של כל מרקר בכל מקטע, את המרחק הכולל שהרגל עברה וכן את זוויות הירך והברך לאורך המקטעים השונים. נצפה כי מהירות המרקר תהיה גבוהה יותר ככל שהוא ממוקם נמוך יותר ברגל הנבדק בשל אופי תנועת הרגל בכך שהחלק התחתון יעבור מרחק רב יותר בזמן זהה לחלקים העליונים של הרגל [5]. כמו כן, נצפה כי החישוב שיתקבל למרחק הכולל שהרגל עברה יהיה שווה לערך לדרך הכוללת של ההליכון עד כדי שגיאה מסוימת התנבע ממספר סיבות; בשל שגיאות זיהוי של המרקר וכן מכיוון שתנועת הרגל אינה ישרה ומתנהגת כפרבולה כלומר תעבור מרחק רב יותר. בנוסף, בשל אופי התנועה, נצפה כי זווית הירך תנוע בין 0 מעלות (מצב עמידה ישרה) לבין זווית הקטנה מכ-60 מעלות וכי זווית הברך תהיה בשינוי של עד כ-90 מעלות ממצבה ההתחלתי שהוא 180 מעלות. כמו כן, נצפה כי זוויות הברך והירך ינועו בצורה מחזורית עקב מחזוריות ההליכה של הנבדק.

### 3.4.2 מתודולוגיה

כלים וחומרים: סרטון וידאו, תוכנת המטלב.

מהלך הניסוי: ראשית ווידאו כי המצלמה ממוקמת כך שתזהה את ההליכון וניתן לזהות את האובייקטים בצורה טובה. מיקמנו 4 מרקרים על רגלו של המשתתף במיקומים שונים לאורך הרגל, 2 מעל הברך ו-2 מתחת לברך. צילמנו וידאו של כ-70 שניות בקצב הליכה של 2 קמ"ש. נשתמש בוידאו זה לניתוח תנועת רגלי אדם במהלך ביצוע פעולה מוטורית.

על מנת למצוא את מיקומי המרקרים בפריימים השונים, ביצענו מספר שלבים. תחילה ביצענו עיבוד מקדים לפריימים הכולל הפיכת הפריים לגווני אפור, יצירת תמונה בינארית עם סף של 0.4 הנבחר בצורה אמפירית וחיתוך האזור הרלוונטי (ROI). בנוסף, השתמשנו במסנן חציון בגודל  $3 \times 3$  לניקוי רעשים מהתמונה. לאחר מכן, השתמשנו בפונקציה `imfindcircles` תוך הגדרת פרמטרים שונים, לזיהוי המרקרים הממוקמים לאורך רגל הנבדק, זאת בשל צורתם העגולה.



### 3.4.3 תוצאות

#### תשובה לשאלה 4.1.1 :

נשרטט דיאגרמת בלוקים עבור האלגוריתם שבנינו למציאת מיקומי המרקרים על רגל הנבדק :

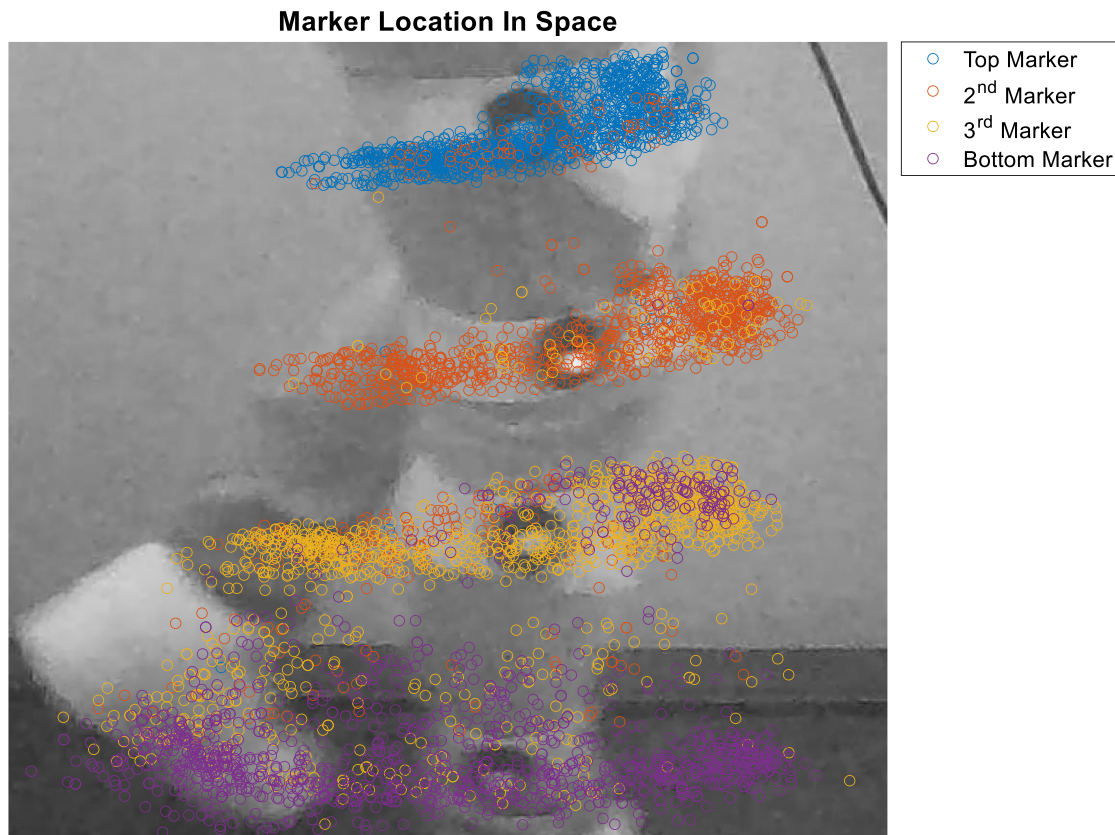


איור 24 : דיאגרמת בלוקים של האלגוריתם שבנינו

באיור 24 ניתן לראות את דיאגרמת הבלוקים למציאת מיקומי המרקרים על רגלו של הנבדק. התהליך מתחיל בהעלאת סרטון והמרת הפריימים שלו לגווני אפור. לאחר מכן, הפריימים מומרים לתמונה בינארית, כאשר נקבע סף של 0.4 באופן אמפירי, כדי להפריד בין האובייקטים לרקע. השלב הבא הוא חיתוך הפריימים לאזור הרלוונטי (ROI) בלבד, כלומר אזור הרגליים של הנבדק על מנת להימנע מזיהוי אובייקטים לא רלוונטיים, צעד זה מאפשר להתמקד בחלק החשוב של התמונה ולהפחית הפרעות מיותרות. לאחר מכן, משתמשים במסנן חציון בגודל  $3 \times 3$  לניקוי רעשים מהתמונה, שיטה יעילה לשמירה על חדות התמונה תוך הסרת רעשים. השלב הבא הוא שימוש בפונקציית `imfindcircles`, עם קביעת פרמטרים שונים (כגון טווח רדיוס המרקר אשר נקבע בעזרת שימוש בפונקציית `imdistline`, זיהוי אובייקט בהיר/כהה, רגישות הזיהוי ועוד) למציאת המרקרים לאורך רגל הנבדק, שלב קריטי לזיהוי האובייקטים הרצויים בתמונה. לאחר מציאת המרקרים, מתבצעת בחירה של ארבעת המיקומים הראשונים שזוהו ומיונם על פי ציר Y, כדי לקבוע את הסדר שלהם בצורה מדויקת. לבסוף, שמירת ערכי המיקומים בקואורדינטות  $(x,y)$  מאפשרת להשתמש בנתונים אלו למעקב אחר פרמטרים נוספים בהמשך, כגון מהירות הנבדק.

#### תשובה לשאלה 4.1.2 :

נציג כעת גרף של מיקום המרקר במרחב :



איור 25 : מיקום המרקרים במרחב

באיור 25 ניתן לראות את מיקום המרקרים במרחב. המרקרים מחולקים ל-4 צבעים ; מרקר עליון (כחול), מרקר שני מלמעלה (כתום), מרקר שלישי מלמעלה (צהוב) ומרקר תחתון (סגול). נשים לב כי המרקר הכחול והכתום נעים במרחב בתחום סגור, פיזור המרקרים מרוכז יחסית ונע יותר בכיוון ציר ה-x מאשר ציר ה-y. לעומת זאת, המרקר הצהוב והסגול נעים במרחב גדול יותר ופיזור המרקרים יותר מפוזר, אך התבנית הכללית של הפיזור דומה לתבנית של שני המרקרים הראשונים, תנועה יותר בכיוון ציר ה-x מאשר ציר ה-y. נשים לב כי פיזור המרקרים והתנועה של כל מרקר עולה ככל שמיקום המרקר יורד במיקומו על הרגל.

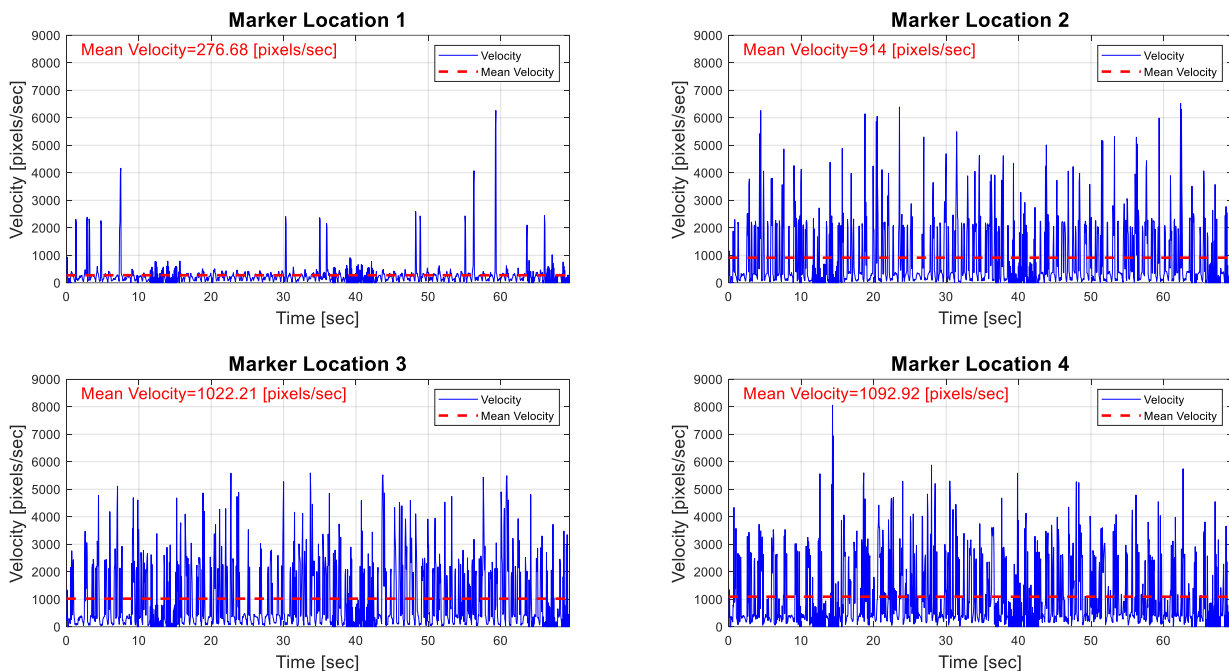
#### תשובה לשאלה 4.2 :

נמצא את מהירות הרגל בערך מוחלט בכל נקודת זמן. נחשב זאת באמצעות שיערוך המהירות לפי מרחק בין פריימים סמוכים חלקי הזמן. עבור כל מרקר נחשב את המרחק האוקלידי לפי נוסחא (2) בין כל 2 פריימים סמוכים. על מנת לחשב את המהירות, נחלק את המרחק שקיבלנו באורך כל פריים. זאת נעשה על פי הנוסחא הבאה [6] :

$$(3) \quad \bar{v} = \frac{\Delta s}{\Delta t}$$

כאשר  $\bar{v}$  הינו המהירות הממוצעת,  $\Delta s$  הינו שינוי במיקום ו- $\Delta t$  זהו משך השינוי במיקום.

נציג גרף של מהירות בערך מוחלט כתלות בזמן:



איור 26: גרף מהירות הרגל כתלות בזמן

באיור 26 ניתן לראות את גרפי המהירות כפונקציה של הזמן, כאשר כל אחד מייצג מיקום שונה של מרקר על רגל הנבדק. מרקר 1 הינו המרקר העליון הממוקם בירך, מרקר 2 ממוקם מעל הברך, מרקר 3 ממוקם מתחת לברך ומרקר 4 הינו המרקר התחתון הממוקם בחלק התחתון של השוק. הגרפים מראים את המהירות של המרקרים במהלך הזמן, כאשר הציר האופקי מייצג את הזמן בשניות והציר האנכי מייצג את המהירות ביחידות של פיקסלים לשנייה. הקו המקווקו האדום מייצג את המהירות הממוצעת לכל מרקר, והוא מאפשר השוואה נוחה בין המהירויות הממוצעות של המרקרים השונים. עבור מרקר 1, הממוקם בחלק העליון ביותר של הרגל, ניתן לראות שהמהירות הממוצעת היא הנמוכה ביותר (276.68 [pixels/sec]) וכי ככל שמיקום המרקר נמוך יותר (בכיוון הרגל) אזי ניתן לראות עלייה במהירות הממוצעת, כאשר המהירות הממוצעת הגבוהה ביותר מתקבלת עבור מרקר 4 (1092.92 [pixels/sec]). בנוסף, הנתונים מציגים פיזור רחב של ערכי המהירות לכל מרקר, מה שעשוי להצביע על השגיאות בסיווג וכן גורמים נוספים המשפיעים על התנועה, כגון תנודות בקצב התנועה של הנבדק או חוסר יציבות במהלך הניסוי.

#### תשובה לשאלה 4.3:

את המרחק הכולל שהרגל עברה בעת ההליכה נוכל לחשב על ידי מדידת המרחק הכולל שהמרקר התחתון עבר. נחשב זאת על ידי סכימת כל הפרשי המרחקים שחישבנו בסעיף הקודם. כעת נרצה להמיר את המרחק מפיקסלים לסנטימטרים, זאת נעשה על ידי מציאת יחס  $\frac{cm}{pixel}$ . ידוע כי גודל הכדור של המרקר הינו 1 סנטימטר ולכן על ידי מדידת מספר הפיקסלים המכילים את גודל הכדור (בעזרת הפונקציה `imdistline` בצורה ידנית) נוכל למצוא את היחס הנ"ל. על ידי כפילה של היחס במרחק הפיקסלים שמצאנו, נקבל כי המרחק הכולל שהרגל עברה הינו 3163.34 [cm].

נשווה את המרחק שקיבלנו למרחק ההליכה האמיתי. נחשב מרחק זה לאחר העברת אגפים בנוסחא (3) והמרת יחידות:

$$2 \left[ \frac{km}{h} \right] * \frac{10^5}{1} \left[ \frac{cm}{km} \right] * 69.52 [sec] * \frac{1}{3600} \left[ \frac{h}{sec} \right] = 3862.29 [cm]$$

#### תשובה לשאלה 4.4 :

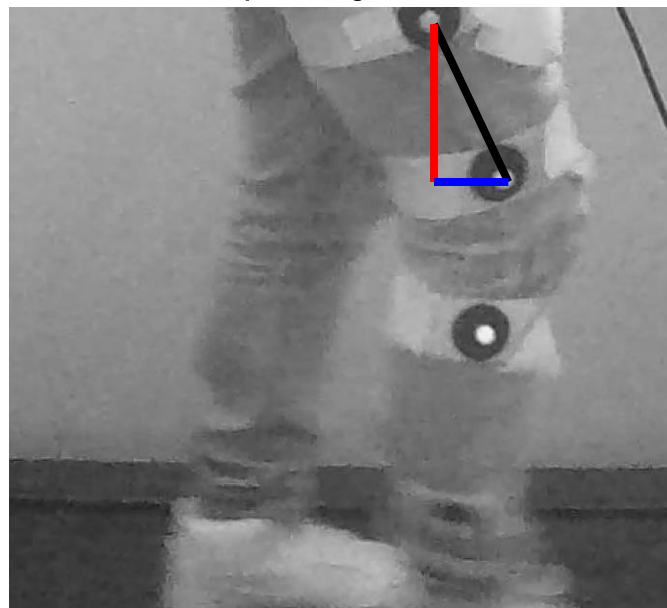
כעת נחשב את זווית הירך בכל שלב בתמונה. את הזווית נחשב בעזרת יחסים טריגונומטרים של משולש ישר זווית [7].

$$(4) \quad \sin(A) = \frac{h}{b}$$

כאשר A הינה הזווית הרצויה לחישוב, h הינו צלע המשולש ממול לזווית A ו-b הינו היתר של המשולש ישר הזווית.

נציג את המשולש הבא על גבי הירך :

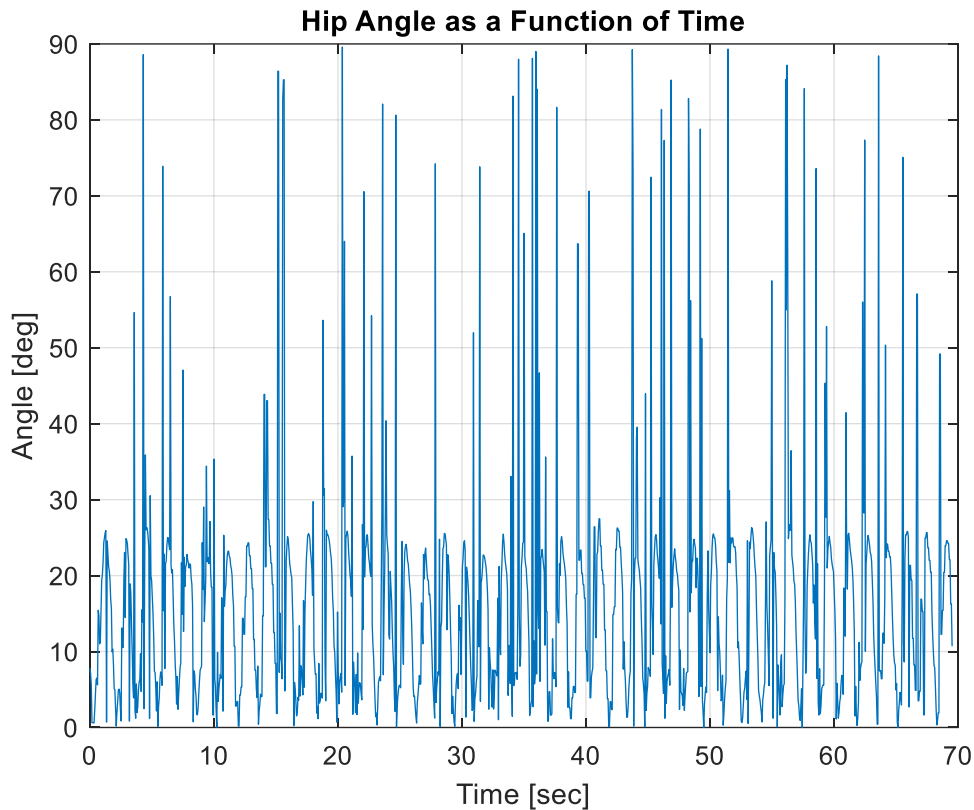
Example of Angle Detection



איור 27 : דוגמא למציאת זווית הירך בעזרת משולש ישר זווית

באיור 27 ניתן לראות את המשולש הישר זווית שהשתמשנו בו על מנת למצוא את זווית הירך. זווית הירך היא הזווית העליונה, בין הקו האדום (הנורמל לרצפה) והקו השחור (המרחק בין 2 המרקרים העליונים). נחשב לפי נוסחא (4) כאשר h הינו הקו הכחול ו-b הינו הקו השחור. בצורה הזו נחשב את הזווית A ביחס לנורמל לרצפה (הקו האדום).

נציג גרף המתאר את השתנות זווית הירך כתלות בזמן :



איור 28: גרף השתנות זווית הירך כתלות בזמן

באיור 28 ניתן לראות את גרף זווית הירך כתלות בזמן. נתייחס לזווית 0 כאשר הנבדק עומד בצורה ישרה. נשים לב כי זווית הירך נעה בין 0 ל-90 מעלות, כאשר רוב הזוויות נעות בין 0 לכ-30 מעלות. בנוסף, נבחין כי השתנות הזווית הינה מחזורית, עד כדי התעלמות מהפיקים החדים, כאשר הזווית מתחילה מ-0 מעלות, מגיעה לפיק מסוים וחוזרת חזרה לכ-0 מעלות.

תשובה לשאלה 4.5 :

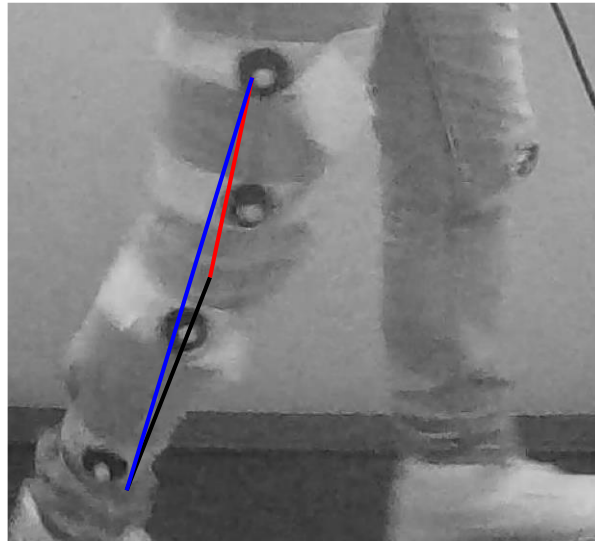
כעת נחשב את זווית הברך בכל שלב בתמונה. את הזווית נחשב בעזרת חוק הקוסינוסים המחושב בדרך הבאה [8]:

$$(5) \quad c^2 = a^2 + b^2 - 2ab \cos(C)$$

כאשר C הינה הזווית הרצויה לחישוב, c הינו צלע המשולש ממול הזווית הרצויה C ו-a ו-b הינן צלעות המשולש האחרות.

נציג את המשולש הבא על גבי הירך :

Example of Angle Detection



איור 29 : המשולש על גבי הברך על מנת לזהות את זווית הברך

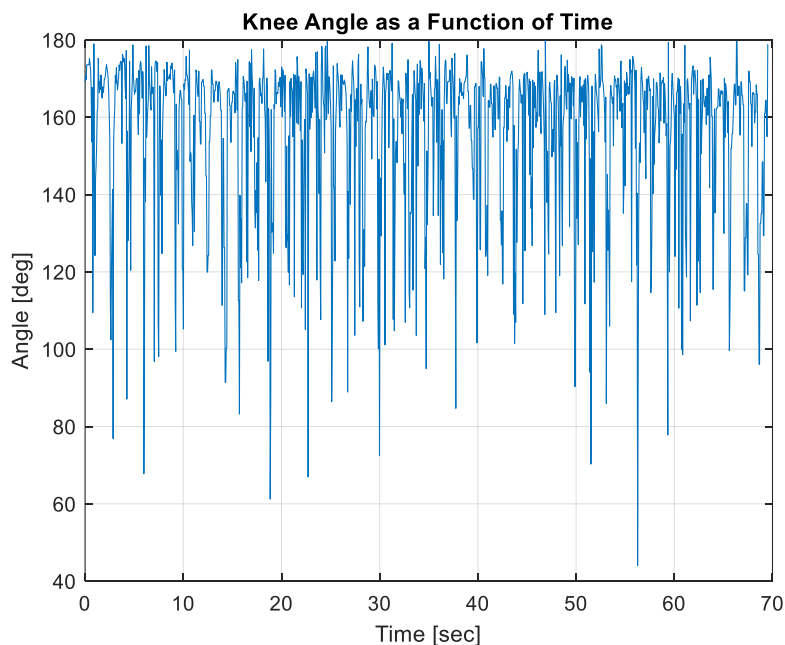
באיור 29 ניתן לראות את המשולש בו שהשתמשנו בו על מנת למצוא את זווית הברך. זווית הברך היא הזווית האמצעית בגובהה, בין הקו האדום ולקו השחור. את נקודת החיתוך בין שני קוים אלו חישובנו על ידי חישוב אמצע קו בצורה הבאה [9]:

$$(6) \quad M = \left( \frac{X_A + X_B}{2}, \frac{Y_A + Y_B}{2} \right)$$

כאשר M הינה נקודת אמצע הקו, A ו-B הינן נקודות הקצה ו-X ו-Y הינן קורדינאטות כל נקודה.

נחשב את זווית הברך לפי נוסחא (5) כאשר c הינו הקו הכחול ו-a ו-b הינם הקווים השחור והאדום. בצורה הזו נחשב את הזווית C שהינה הזווית שממול הקו הכחול.

נציג גרף המתאר את השתנות זווית הברך כתלות בזמן :



איור 30 : גרף השתנות זווית הברך כתלות בזמן

באיור 30 ניתן לראות את גרף זווית הברך כתלות בזמן. נתייחס לזווית 180 לזווית כאשר הנבדק עומד בצורה ישרה. נשים לב כי זווית הברך נעה בין 40 ל-180 מעלות, כאשר רוב הזוויות נעות בין 120 ל-180 מעלות. בנוסף, נבחין כי השתנות הזווית הינה מחזורית, כאשר הזווית מתחילה מ-180 מעלות, מגיעה לפיק מסוים (מינימום מקומי) וחוזרת חזרה ל-180 מעלות.

#### 3.4.4 מסקנות

בניסוי זה ביצענו ניתוח של תנועת רגלו הימנית של הנבדק, באמצעות 4 מרקרים שמוקמו לאורך רגלו הימנית במיקומים שונים (2 מעל לברך ו-2 מתחת לברך). בעזרת הפונקציה שבנינו לזיהוי המרקרים, מצאנו את מיקומם וביצענו חישוב של מהירות, מרחק, זווית הירך וזווית הברך של הנבדק.

מציאת מיקומי המרקרים התבצעה תוך שימוש בפריימים, חיתוך כל פריים כך שיתמקד ברגל (ROI) ועיבוד מקדים של התמונה אשר כלל המרת התמונה לגווני אפור, מציאת סף בינארי להבלטת המרקרים וסינון רעשים בעזרת מסנן חציון. השתמשנו בפונקציה `imfindcircles` למציאת המרקרים (צורתם עגולה וצבעם לבן) ושמרנו את המיקומים. ראינו כי בהתאם לציפיותינו, המהירות של כל מרקר גדלה ככל שמיקומו נמוך יותר, דבר זה הגיוני מכיוון שבתנועת הרגל החלק התחתון עובר מרחק גדול יותר בהשוואה לחלק העליון בפרק זמן זהה [5]. בנוסף, ראינו כי המרחק הכולל שהרגל עברה, כפי שנמדד בניסוי, הוא 3163.34 סנטימטרים, בעוד שהמרחק האמיתי שנמדד היה 3862.29 סנטימטרים. הפער בין שני המדדים תואם לציפיותינו כי יתקבלו שגיאות במהלך המדידה, וסביר כי ערך המרחק שהתקבל גדול יותר בשל שגיאות בזיהוי. בנוסף, הנתונים המתקבלים מהגרפים של זוויות הירך והברך מצביעים על פיזור רחב יחסית ופיקים שחורגים מהערכים הצפויים, דבר המחזק את קיומן של שגיאות בזיהוי המרקרים במהלך הניסוי.

מאיור 25 נראה כי הפונקציה שבנינו לזיהוי המיקומים הייתה יעילה במרבית המקרים. הסקנו כי לקבלת תוצאות טובות ואמינות יותר יש לדאוג שהרקע וגוון הלבוש של הנבדק יהיו אחידים ובגוון מתאים כך שיהיו הפרעות מינימליות ולא יפגמו בטיב הזיהוי. בנוסף, ראינו כי התקבלו פריימים רבים בהם איכות התמונה הייתה ירודה (טשטוש) והתרחשו שגיאות בזיהוי הסמנים. מכך נסיק כי ישנה חשיבות לאיכות המצלמה, יציבותה, מיקומה ותאורה מתאימה. כמו כן, ישנה חשיבות רבה לבחירת מסיכה נכונה. שינויים מזעריים יחסית בבניית המסכה (בחירת סף) השפיעו על הזיהוי באופן ניכר. כמו כן, ישנם פרמטרים רבים, אשר שינוי, הוספה והתאמתם אותם בצורה המיטבית שהצלחנו, אך סביר כי ניתן יהיה לשנות פרמטרים אלו ולהוסיף אחרים על מנת להגיע למצב אידיאלי יותר וקבלת מסכה איכותית יותר. לכן, ישנו צורך בשיפור טיב המסכה והכרת טכניקות נוספות.



## 4 מסקנות כלליות

במעבדה זו הכרנו את מבנה התמונה ותכונותיה במקום ובתדר. התנסינו בסינון תמונה בדרכים שונות, ביצוע פעולות מורפולוגיות, סגמנטציה וזיהוי גבולות, מיצוי מאפייני אובייקטים בתמונה וכן עיבוד וידאו.

בניסוי 1 עבדנו עם תמונה של אורז אליה התייחסנו כתמונת טסיות. מטרתנו הייתה ליצור תמונה המורכבת אך ורק מטסיות הנוטות ימינה (/). ביצענו זאת בעזרת סגמנטציה ומיצוי מאפיינים של טסיות הדם. ראינו כי הסרת הרקע שיפרה את איכות הסגמנטציה ואפשרה יצירת מסכה מדויקת יותר לזיהוי האובייקטים הרצויים. דבר זה הצביע על כך שעיבוד מקדים של תמונה עשוי להיות בעל חשיבות רבה בהצלחת הליך הסגמנטציה. בנוסף, השתמשנו בפילטרים ושיטות שונות לזיהוי גבולות. ראינו כי שיטות כמו LoG ו-dilation יכולות להיות יעילות לזיהוי גבולות האובייקטים. עם זאת, ייתכן כי במקרים מסוימים פעולות אלו עלולות לשנות את גבולות האובייקטים ולגרום לזיהוי שגוי של אובייקטים סמוכים כאחד. כמו כן, ראינו כי ניתן להשתמש במאפיינים כמו אוריינטציה לזיהוי והפרדה של אובייקטים לפי כיוונית מסוימת. שיטה זו נמצאה כיעילה בסינון אובייקטים בתמונה לפי תכונה גאומטרית מסוימת.

בניסוי 2 ביצענו סינון של רעש מלח פלפל מתמונה בעזרת מסנני חציון בגדלים שונים. ראינו כי סוג מסנן החציון אפקטיבי לסינון רעש מסוג זה וכי גודל הגרעין משפיע על התוצאה המתקבלת. קיבלנו כי גרעין בגודל  $3 \times 3$  אופטימלי במקרה זה לשמירה על חדות התמונה תוך סינון הרעש (איזון בטרייד אוף) והניב ערך מינימלי עבור מדד ה-MSE. השתמשנו בניתוח סטטיסטי לבחינת עקביות התוצאה. קיבלנו כי קיים הבדל מובהק בין מסנני החציון הריבועיים השונים. ראינו כי חזרות ניסויים וניתוחים סטטיסטיים יכולים להצביע על עקביות התוצאות ולהבדיל בין גדלים שונים של מסננים, דבר המחזק את מוצקות התוצאות.

בניסוי 3 סיווגנו את כיווני ההסתכלות של הנבדק ל-4 אזורי הסתכלות. ביצענו זאת תוך שימוש בתמונות כיוול, מציאת מיקומי העיניים בתמונות אלו ואפיון מיקומי העיניים. בניתוח הסרטון ביצענו התאמות של התמונה כך שתתאים למטריצת הכיוול שבנינו, זאת בשל שינוי במנח הפנים של הנבדק בסרטון ביחס לתמונות הכיוול. ראינו כי להתאמה נכונה לתמונות הכיוול ישנה חשיבות רבה באחוזי ההצלחה של זיהוי אזור ההסתכלות. בנוסף, קיבלנו כי הסיווג בעזרת עין ימין סיפק תוצאות מדויקות יותר בשל זיהוי שגוי של מיקום עין שמאל במקרים רבים, אשר סברנו כי נובעת בשל שינויים כגון תאורה אשר הקשו על זיהוי מיקום עין שמאל בצורה טובה. לכן, במקרה זה נעדיף להסתמך על עין ימין בסיווג אזור ההסתכלות, אשר גם בבדיקת תמונות הכיוול הניבה תוצאה טובה יותר. כמו כן, ראינו כי חפיפת אזורים עלולה לגרום לשגיאות סיווג. עבור עין ימין קיבלנו הפרדה טובה למדי בין אזורי ההסתכלות השונים.

בניסוי 4 ניתחנו סרטון של תנועת רגלו הימנית של הנבדק בעזרת מרקרים שמוקמו לאורך הרגל בגבהים שונים. אפיון המידע כלל את מיקום המרקרים, מהירות, זווית הברך וזווית הירך. נמצא כי המהירות של כל מרקר על הרגל גדלה ככל שמיקומו נמוך יותר, דבר התואם לציפיותינו מתנועת הרגל. כמו כן, ראינו כי המרחק הכולל שהרגל עברה, כפי שנמדד בניסוי, הוא 3163.34 סנטימטרים, בעוד שהמרחק האמיתי שנמדד היה 3862.29 סנטימטרים. הפער בין שני המדדים תאם לציפיותינו לקבלת שגיאות בתהליך המדידה, והיה סביר כי ערך המרחק שהתקבל גדול יותר בשל שגיאות בזיהוי. הנתונים שהתקבלו בגרפים של זוויות הירך והברך הציגו פיזור רחב יחסית ופיקים שחורגים מהערכים הצפויים, דבר אשר חיזק את קיומן של שגיאות בזיהוי המרקרים במהלך הניסוי. כמו כן, ראינו כי איכות הזיהוי של המרקרים תלויה באיכות הרקע ובגוון הלבוש של הנבדק. רקע ולבוש אחידים ובגוון מתאים יכולים לשפר את טיב הזיהוי. ניסוי זה הדגים את החשיבות של איכות המצלמה, יציבותה, מיקומה ותאורה מתאימה בזיהוי מרקרים ובניתוח תנועה. פרמטרים אלו משפיעים באופן ניכר על דיוק הזיהוי ועל איכות התוצאות הסופיות. בנוסף, ראינו כי השימוש במסנן חציון לסינון רעשים ובחירת סף בינארי מתאים להבלטת המרקרים הוא קריטי לזיהוי נכון של המרקרים, שכן שינויים קטנים בבחירת המסכה יכולים להשפיע משמעותית על דיוק הזיהוי.



- [1] J. Frost, "Mean Squared Error (MSE)," Statistics By Jim. Accessed: Mar. 26, 2024. [Online]. Available: <https://statisticsbyjim.com/regression/mean-squared-error-mse/>
- [2] "One-Way ANOVA." Accessed: Mar. 26, 2024. [Online]. Available: [https://www.jmp.com/en\\_be/statistics-knowledge-portal/one-way-anova.html](https://www.jmp.com/en_be/statistics-knowledge-portal/one-way-anova.html)
- [3] "Image Coordinate Systems - MATLAB & Simulink." Accessed: Mar. 26, 2024. [Online]. Available: <https://www.mathworks.com/help/images/image-coordinate-systems.html>
- [4] D. Cohen, *Precalculus: A Problems-Oriented Approach*. Cengage Learning, 2004.
- [5] J. M. Winters and S. L.-Y. Woo, Eds., *Multiple Muscle Systems: Biomechanics and Movement Organization*. New York, NY: Springer, 1990. doi: 10.1007/978-1-4613-9030-5.
- [6] "The Feynman Lectures on Physics Vol. I Ch. 8: Motion." Accessed: Mar. 26, 2024. [Online]. Available: [https://www.feynmanlectures.caltech.edu/I\\_08.html](https://www.feynmanlectures.caltech.edu/I_08.html)
- [7] "2.1: The Law of Sines," Mathematics LibreTexts. Accessed: Mar. 26, 2024. [Online]. Available: [https://math.libretexts.org/Bookshelves/Precalculus/Elementary\\_Trigonometry\\_\(Corral\)/02%3A\\_General\\_Triangles/2.01%3A\\_The\\_Law\\_of\\_Sines](https://math.libretexts.org/Bookshelves/Precalculus/Elementary_Trigonometry_(Corral)/02%3A_General_Triangles/2.01%3A_The_Law_of_Sines)
- [8] "2.2: The Law of Cosines," Mathematics LibreTexts. Accessed: Mar. 26, 2024. [Online]. Available: [https://math.libretexts.org/Bookshelves/Precalculus/Elementary\\_Trigonometry\\_\(Corral\)/02%3A\\_General\\_Triangles/2.02%3A\\_The\\_Law\\_of\\_Cosines](https://math.libretexts.org/Bookshelves/Precalculus/Elementary_Trigonometry_(Corral)/02%3A_General_Triangles/2.02%3A_The_Law_of_Cosines)
- [9] "Midpoint of a Line Segment." Accessed: Mar. 26, 2024. [Online]. Available: <https://www.mathsisfun.com/algebra/line-midpoint.html>

```

% Adjusting the default figure settings
set(0,'defaultAxesXGrid','on')
set(0,'defaultAxesYGrid','on')
print=0;
%% EXP 1
% 1.1
% Loading blood platelets picture
rice=imread('rice.png');
rice=mat2gray(rice);
% Showing the picture
figure
imshow(rice)
title('Blood Platelets Photo')

% 1.2
% Plotting histogram
figure
imhist(rice)
title('Histogram of Gray Levels - Original Photo')
xlabel({'',' ',' ','Gray Levels'})
ylabel('Count')
lim=ylim;
ylim(lim+[0 500])

% 1.3
% Creating mask manually
TH=0.5;
riceMask=rice;
riceMask(riceMask>=TH)=1;
riceMask(riceMask<TH)=0;
% Showing the mask
figure
imshow(riceMask)
title('Rice Mask - Manual Threshold')

% 1.4
% Cheacking manually what are the dimentions of the blood platlets
h=imdistline;
delete(h)
% Creating the disk structure with the smaller blood platlets radius
disk=strel('disk',10);
% Using erosion to remove the blood platelets
notRice=imerode(rice,disk);
% Showing the background
figure
imshow(notRice)
title('Background')
% Removing background
onlyRice=rice-notRice;
figure
imshow(onlyRice)
title('Only the Blood Platlets')

% 1.5
% Plotting new histogram
figure

```

```

imhist(onlyRice)
title('Histogram of Gray Levels - After Filtering')
xlabel({'',' ',' ','Gray Levels'})
ylabel('Count')
lim=ylim;
ylim(lim+[0 1500])
% New threshold
TH=0.35;
riceMaskOR=onlyRice;
riceMaskOR(riceMaskOR>=TH)=1;
riceMaskOR(riceMaskOR<TH)=0;
% Plotting new mask
figure
imshow(riceMaskOR)
title('Blood Platelets Mask - After Background Removal')

% Edge detection dilation-mask
edgeD_M=imdilate(riceMaskOR,strel('diamond',1))-riceMaskOR;
% Plotting the results
figure
nexttile
imshow(edgeD_M)
title('Blood Platelets After Dilation-Mask')
% Edge detection laplacian of gaussian
nexttile
edge(onlyRice,'log')
title('Blood Platelets After LoG')

% 1.6
% Calculating number of blood plateles
[ricEdges,nObjs]=bwlabel(imclearborder(edgeD_M));
figure
imshow(ricEdges)
title(['Clean Edge Detection - Number of Blood Plateles=' num2str(nObjs)])

% 1.7
% Finding the blood platlets with right orientation
orientInfo=regionprops(ricEdges,'Orientation','FilledImage','BoundingBox');
RightOrientInfo=orientInfo([orientInfo.Orientation]>0);
% Creating The mask. Starting with blank photo
RightMask=zeros(size(rice));
for i=1:length(RightOrientInfo)
    % Indexes for up left corner
    LeftCornerX=ceil(RightOrientInfo(i).BoundingBox(1));
    LeftCornerY=ceil(RightOrientInfo(i).BoundingBox(2));
    % All the indices for the right oriented blood platletes
    indX=LeftCornerX:LeftCornerX+RightOrientInfo(i).BoundingBox(3)-1;
    indY=LeftCornerY:LeftCornerY+RightOrientInfo(i).BoundingBox(4)-1;
    % Adding back the parts containing the full right blood platlets
    RightMask(indY,indX)=RightMask(indY,indX)+RightOrientInfo(i).FilledImage;
end
% Showing the mask
figure
imshow(RightMask)
title('Only Right Orianted Blood Plateles - Mask')

% 1.8
% Applying the mask to the original photo
OnlyRightRice=rice.*RightMask;

```

```

% Showing results
figure
imshow(OnlyRightRice)
title('Only Right Oriented Blood Platelets - After Masking')

%% Exp 2
% Loading photos
RegVeg=imread('images\vegetables.tif');
RegVeg=mat2gray(RegVeg);
NoiseVeg=imread('images\Nvegetables.tif');
NoiseVeg=mat2gray(NoiseVeg);
% Showing the photos
figure
nexttile
imshow(RegVeg)
title('Vegetables')
nexttile
imshow(NoiseVeg)
title('Noisy Vegetables')

% 2.1
% Filter dimentions
var1=[1 1 2 3 3 4 4 5 7];
var2=[2 3 2 3 4 4 5 5 7];
% Allocating memory
MSE=zeros(length(var2),1);
figure(WindowState="maximized")
for i=1:length(var2)
    % Median filtering
    FilteredVeg=CleanSP(NoiseVeg, 'Median', var1(i), var2(i));
    % Calculating MSE
    MSE(i)=mean((FilteredVeg-RegVeg).^2, "all");
    % Showing results
    nexttile
    imshow(FilteredVeg)
    title({'Median Filter ' num2str(var1(i)) 'x' num2str(var2(i))}, ['MSE='
num2str(MSE(i))]], FontSize=16)
end
% Plotting MSE graph
figure
plot(MSE)
title('MSE for Various Median Filters Sizes')
xlabel('Median Size')
ylabel('MSE')
xticklabels({'1x2' '1x3' '2x2' '3x3' '3x4' '4x4' '4x5' '5x5' '7x7'})

% 2.2
% Allocating memory
MSE=zeros(100,length(var2));
% Loop for 100 different noise and filtering simulation
for i=1:100
    % Adding salt&peper noise - 20%
    SPNoisyVeg=imnoise(RegVeg, 'salt & pepper', 0.2);
    for j=1:length(var2)
        % Filtering with all the median filters sizes
        FilteredVeg=CleanSP(SPNoisyVeg, 'Median', var1(j), var2(j));
        % Calculating MSE
        MSE(i,j)=mean((FilteredVeg-RegVeg).^2, "all");
    end
end

```

```

end
% Calculating mean and std
meanMSE=mean(MSE);
stdMSE=std(MSE);

% Plotting boxplot
figure(WindowState="maximized")
boxplot(MSE,'Labels',{ '1x2' '1x3' '2x2' '3x3' '3x4' '4x4' '4x5' '5x5' '7x7'})
title('MSE boxplot',FontSize=16)
xlabel('Median Filter Size',FontSize=14)
ylabel('MSE',FontSize=14)

% Statistical test for square filters
squareMSE=MSE(:,var1==var2);
% One-way ANOVA statistical test
[p,tbl]=anova1(squareMSE,[],'off');
disp(tbl)
disp(['p-Value=' num2str(p)])

%% Exp 3
% 3.1
% Load Data
ImageFolder='image processing lab\eyes\';
% Get a list of all files in the folder with the desired file name pattern
ImagData=dir(fullfile(ImageFolder,'*.jpg'));

lenIm=length(ImagData);
% Preallocate a cell array to hold the images
EyeImag=cell(lenIm, 1);
for i=1:lenIm
    % Read the current image
    imgPath=fullfile(ImageFolder,ImagData(i).name);
    img=imread(imgPath);
    img=im2gray(img);
    img=mat2gray(img);

    EyeImag{i}=img;
end

% 3.3
% Preallocating
N=lenIm/4;
CalibFrames={cell(1,N);cell(1,N);cell(1,N);cell(1,N)};
% 3.3.1
% Creating CalibFrame cell array
for i=1:lenIm
    Frame=EyeImag{i};
    Frame=imcrop(Frame,[110 3.5 343.5 702 129]);

    CalibFrames{mod(i-1,4)+1}{ceil(i/4)}=Frame;
end

% 3.1
% Plotting an example
Eye_Pos=EyePosition_FUNC(Frame);
% Plotting
figure
imshow(Frame)
hold on

```

```

viscircles(Eye_Pos,28,'EdgeColor','b');
plot(Eye_Pos(1,1),Eye_Pos(1,2),'r*')
plot(Eye_Pos(2,1),Eye_Pos(2,2),'r*')
title('Eye Detection Example')

% 3.3
% Calculating the calibration matrix
CalibMat=EyeCalibration_FUNC(CalibFrames);

% 3.3.2
% 3-D Histogram of Left Eye
figure
hist3(CalibMat(:,[1 2],1),'FaceColor','magenta')
hold on
hist3(CalibMat(:,[3 4],1),'FaceColor','blue')
hist3(CalibMat(:,[5 6],1),'FaceColor','yellow')
hist3(CalibMat(:,[7 8],1),'FaceColor','white')

title('Calibration 3-D Histogram - Left Eye Position')
xlabel('X index')
ylabel('Y index')
zlabel('Count')
xlim padded
ylim padded
zlim tight
legend('1','2','3','4')

% 3-D Histogram of Right Eye
figure
hist3(CalibMat(:,[1 2],2),'FaceColor','magenta')
hold on
hist3(CalibMat(:,[3 4],2),'FaceColor','blue')
hist3(CalibMat(:,[5 6],2),'FaceColor','yellow')
hist3(CalibMat(:,[7 8],2),'FaceColor','white')

title('Calibration 3-D Histogram - Right Eye Position')
xlabel('X index')
ylabel('Y index')
zlabel('Count')
xlim padded
ylim padded
zlim tight
legend('1','2','3','4')

% 3.4.2
% Testing the calibration matrix
for i=1:4
    Eye_Look=EyeLook_FUNC(CalibFrames{i}{end},CalibMat);
    disp(['Left Eye Look Number: ' num2str(Eye_Look(1))])
    disp(['Right Eye Look Number: ' num2str(Eye_Look(2))])
    disp(' ')
end

% 3.5
% Loading eye movement video
vid=VideoReader([ImageFolder 'WIN_20240123_05_11_14_Pro.mp4']);
% Segmentation
segmentation=ceil([0 2.09,4.26,6.23,8.27,10.25,12.30,13.98,15.95,17.91 ...
    ,20.06,22.01,24.10,26.30,28.04,29.31]+1.61)*vid.FrameRate);

```

```

% Calculate look duration
lookDuration=diff(segmentation);
% The number on screen look order
numOrder=[1,2,3,2,3,4,3,2,1,2,1,2,3,1,2];
numOrderFrame=repelem(numOrder,lookDuration);

% 3.6
% For removing the beggining part of now eye movement
displace=segmentation(1);
% Preallocating memory
Eye_Look=zeros(vid.NumFrames-displace,2);
for i=displace+1:vid.NumFrames
    % Extracting frame by frame
    Frame=read(vid,i);
    Frame=im2gray(Frame);
    Frame=mat2gray(Frame);

    % Finding the relevant ROI
    cropPosition=[1103.5 343.5 702 129]-[320 75 702/4 129/4];
    currFrame=imcrop(Frame,cropPosition);
    % Resizing the current frame
    targetSize=size(CalibFrames{1}{1});
    currFrame=imresize(currFrame,targetSize);

    % Applying our functions
    Eye_Look(i-displace,:)=EyeLook_FUNC(currFrame,CalibMat);
end

% Plotting confusion matrixes
figure
nexttile
confusionchart(numOrderFrame, Eye_Look(:,1))
errLeft=nnz(logical(numOrderFrame.'-Eye_Look(:,1)))/length(numOrderFrame)*100;
title(['Confusion Matrix for Left Eye - Error Percentage=' num2str(errLeft)])

nexttile
confusionchart(numOrderFrame, Eye_Look(:,2))
errRight=nnz(logical(numOrderFrame.'-Eye_Look(:,2)))/length(numOrderFrame)*100;
title(['Confusion Matrix for Right Eye - Error Percentage=' num2str(errRight)])

% Plotting comparison
figure(WindowState="maximized")
plot(numOrderFrame,LineWidth=2)
hold on
scatter(1:length(numOrderFrame),Eye_Look(:,2),Marker="x")
title('Algorithm Recongnition Comparison',FontSize=16)
xlabel('Frame',FontSize=14)
ylabel('Look State',FontSize=14)
legend('True Labels','Predicted Labels Right',FontSize=13)
xlim tight
ylim padded
yticks([1 2 3 4])

%% EXP 4
% Loading the walking video
vid=VideoReader('image processing lab\WIN_20240123_14_45_03_Pro.mp4');
% 4.1
[AllLegPositionX,AllLegPositionY]=deal(zeros(4,vid.NumFrames));
for i=1:vid.NumFrames

```

```

clc
% Read frame
Frame=read(vid,i);
Frame=im2gray(Frame);
Frame=mat2gray(Frame);
% Creating a mask and cropping the ROI
currFrame=imbinarize(Frame,0.4);
currFrame=imcrop(currFrame,[589.5 380 487 440]);
% Applying median filter
currFrame=CleanSP(currFrame,'Median',3,3);
% Applying imfindcircles function
min_rad=4;
max_rad=15;
[centers, radii, metric]=imfindcircles(currFrame,[min_rad max_rad] ...
    , 'ObjectPolarity','bright','EdgeThreshold',0.7,'Sensitivity',0.88 ...
    , 'Method','TwoStage');
% Find indices
Leg_Pos=ceil(centers);

% Sorting the centers according to Y index
centers4=centers(1:4,:);
[~,Ind]=sort(centers4(1:4,2));
% Allocating centers
AllLegPositionX(:,i)=centers(Ind,1);
AllLegPositionY(:,i)=centers(Ind,2);
end
% 4.1.2
% Plotting walk representation
FrameEx=imcrop(Frame,[589.5 380 487 440]);
figure(WindowState="maximized")
nexttile
imshow(FrameEx)
% 4.3 Measuring marker size manually
h=imdistline;
delete(h)
% End
hold on
scatter(AllLegPositionX(1,:),AllLegPositionY(1,:))
scatter(AllLegPositionX(2,:),AllLegPositionY(2,:))
scatter(AllLegPositionX(3,:),AllLegPositionY(3,:))
scatter(AllLegPositionX(4,:),AllLegPositionY(4,:))

title('Marker Location In Space',FontSize=16)
legend('Top Marker','2^{nd} Marker','3^{rd} Marker','Bottom Marker' ...
    ,Location='bestoutside',FontSize=13)

% 4.2
% Find the absolute velocity
FrameTime=vid.Duration/vid.NumFrames; % Time for each frame
% Speed Calculation
N=size(AllLegPositionX,2)-1;
% Initialize matrices to store distances and velocities
[distances,velocities]=deal(zeros(4,N));

% Calculate distances and velocities
for i=1:4 % For each location
    for j=1:N % For each frame, excluding the last one
        % Calculate distance between consecutive frames
        distances(i,j)=sqrt((AllLegPositionX(i,j+1)-AllLegPositionX(i,j))^2 ...

```



```

        +(AllLegPositionY(i,j+1)-AllLegPositionY(i,j))^2);
        velocities(i,j)=distances(i,j)/FrameTime;
    end
end
% distances matrix now contains the distance between each consecutive frame for
each location
% velocities matrix contains the velocity in pixels/frame for each location

% Absolute velocities
AbsVelocities=abs(velocities);
% Mean Velocity Calculation
MeanVel=mean(AbsVelocities,2);

% Assuming velocities is a 4xN matrix with the velocities for each location
% Assuming MeanVel is a 4x1 vector with the mean velocities for each location
% Assuming FrameTime is the time for each frame

% Calculate the time points for the x-axis
time=linspace(0,vid.Duration,N);
% Create subplots for each location
figure(WindowState="maximized")
for i=1:4
    subplot(2,2,i)
    plot(time,velocities(i,:), 'b') % Plot absolute velocity for location i
    hold on
    plot(time,MeanVel(i)*ones(size(time)), 'r--', 'LineWidth',2) % Plot mean
velocity line for location i
    text(2,8500,['Mean Velocity=' num2str(round(MeanVel(i),2)) '
[pixels/sec]'],color='r',FontSize=13)

    % Add title and labels
    title(sprintf('Marker Location %d',i),FontSize=16)
    xlabel('Time [sec]',FontSize=14)
    ylabel('Velocity [pixels/sec]',FontSize=14)
    xlim tight
    ylim([0 9000])
    % Add legend
    legend('Velocity','Mean Velocity')
end

% 4.3
% Converting from pixels to cm
MarkerSize=1; % [cm]
PixelMarkerSize=24; % [Pixels]
% Ratio
ratioPixel2Cm=MarkerSize/PixelMarkerSize;
% Calculating cumulative distance according to the bottom marker in cm
walkDist=sum(distances(4,:))*ratioPixel2Cm;
realWalkDist=2*vid.Duration*1/3600*10^5;
disp(['Walking Distance = ' num2str(walkDist) ' [cm]'])
disp(['Real Walking Distance = ' num2str(realWalkDist) ' [cm]'])

% 4.4
% Calculating distance between the top markers
hipAngleCalcX=diff([AllLegPositionX(1,:);AllLegPositionX(2,:)]).^2;
hipAngleCalcY=diff([AllLegPositionY(1,:);AllLegPositionY(2,:)]).^2;

hypoten=sqrt(hipAngleCalcX+hipAngleCalcY);
% Calculating the parallel to the hypotenuse

```

```

para=sqrt(hipAngleCalcX);

% Using a right triangle trigonometric ratio, calculate the apex angle
hipAngle=rad2deg(asin(para./hypoten));

% Calculate the time vector and plot the hip angle
time=linspace(0,vid.Duration,N+1);
figure
plot(time,hipAngle)
title('Hip Angle as a Function of Time')
xlabel('Time [sec]')
ylabel('Angle [deg]')

% Example of hip angle
Nframe=400;
Frame=read(vid,Nframe);
Frame=im2gray(Frame);
Frame=mat2gray(Frame);
Frame=imcrop(Frame,[589.5 380 487 440]);
% Example of angle detection
figure
imshow(Frame)
hold on
plot([AllLegPositionX(1,Nframe),AllLegPositionX(2,Nframe)], ...
     [AllLegPositionY(1,Nframe),AllLegPositionY(2,Nframe)], 'k',LineWidth=3.5)
plot([AllLegPositionX(1,Nframe),AllLegPositionX(1,Nframe)], ...
     [AllLegPositionY(1,Nframe),AllLegPositionY(2,Nframe)], 'r',LineWidth=3.5)
plot([AllLegPositionX(1,Nframe),AllLegPositionX(2,Nframe)], ...
     [AllLegPositionY(2,Nframe),AllLegPositionY(2,Nframe)], 'b',LineWidth=3.5)
title('Example of Angle Detection')

% 4.5
% Calculating all the triangle legs distance
kneeAngleCalcX=diff([AllLegPositionX(1,:);
    (AllLegPositionX(2,:)+AllLegPositionX(3,:))/2; % Middle leg calculated as the
mean between the 2 middle markers
    AllLegPositionX(4,:); ...
    AllLegPositionX(1,:)]).^2;
kneeAngleCalcY=diff([AllLegPositionY(1,:);
    (AllLegPositionY(2,:)+AllLegPositionY(3,:))/2; % Middle leg calculated as the
mean between the 2 middle markers
    AllLegPositionY(4,:); ...
    AllLegPositionY(1,:)]).^2;

kneeAngleCalc=sqrt(kneeAngleCalcX+kneeAngleCalcY);
% Calculating the knee angle using the law of cosines
a=kneeAngleCalc(1,:);
b=kneeAngleCalc(2,:);
c=kneeAngleCalc(3,:);

kneeAngle=rad2deg(acos((a.^2+b.^2-c.^2)./(2*a.*b)));
% Plotting knee angle
figure
plot(time,kneeAngle)
title('Knee Angle as a Function of Time')
xlabel('Time [sec]')
ylabel('Angle [deg]')

% Example of knee angle

```

```

figure
Nframe=362;
Frame=read(vid,Nframe);
Frame=im2gray(Frame);
Frame=mat2gray(Frame);
Frame=imcrop(Frame,[589.5 380 487 440]);
% Example of angle detection
imshow(Frame)
hold on
plot([AllLegPositionX(4,Nframe),mean([AllLegPositionX(2,Nframe),AllLegPositionX(3,
Nframe)])], ...

[AllLegPositionY(4,Nframe),mean([AllLegPositionY(2,Nframe),AllLegPositionY(3,Nfram
e)])], 'k', LineWidth=2)
plot([mean([AllLegPositionX(2,Nframe),AllLegPositionX(3,Nframe)]),AllLegPositionX(
1,Nframe)], ...

[mean([AllLegPositionY(2,Nframe),AllLegPositionY(3,Nframe)]),AllLegPositionY(1,Nfr
ame)], 'r', LineWidth=2)
plot([AllLegPositionX(1,Nframe),AllLegPositionX(4,Nframe)], ...
[AllLegPositionY(1,Nframe),AllLegPositionY(4,Nframe)], 'b', LineWidth=2)
title('Example of Angle Detection')

```

### CleanSP פונקציית

```

function out_I = CleanSP(in_I, Type, var1, var2)
    if strcmp(Type, 'Gaussian')
        h = fspecial('gaussian', var1, var2);
        out_I = filter2(h, in_I, 'same');
    elseif strcmp(Type, 'Median')
        out_I = medfilt2(in_I, [var1 var2]);
    else
        error('Invalid filter type');
    end
end

```