מעבדה לעיבוד אותות פיזיולוגיים

דו"ח מכין- עיבוד תמונה

מגישי הדו"ח האלופים:

אוראל בטיטו

315432724

עדן קורש

206845315

מדריכת המעבדה המהוללת:

נופר ירימי

21/01/2024

תוכן עניינים

תשובות לשאלות הכנה:

1. רמות אפור:

1.1. שאלה 1

את מספר הביטים המייצגים פיקסל אחד כאשר ישנם 256 רמות אפור ניתן לחשב בדרך הבאה:

$$2^N = L \tag{1}$$

כאשר N הינו מספר הביטים (מספר שלם) ו-L הינו מספר הרמות.

- במקרה זה נחשב:

$$2^N = 256$$
$$\rightarrow N = \log_2 256 = 8$$

לכן 8 ביטים מייצגים 256 רמות אפור.

- בתמונת RGB, כל נקודה בתחום ההגדרה של התמונה תקבל לא ערך אחד, כמקודם, אלא שלושה ערכים המתייחסים לעוצמות (R-Red, B-Blue, G-Green). לכן, במקום N ביטים, תמונה צבעונית תדרוש שלושה ערכים כאלו ולכן מספר הביטים יהיה גדול פי 3.

כלומר, $8 \cdot 3 = 24$ ביטים מייצגים תמונת RGB בעלת 256 רמות.

- כאשר ישנם 20 רמות אפור נחשב:

$$2^N = 20$$
$$\rightarrow N = \log_2 20 = 4.322$$

במצב שכזה נעגל את מספר הביטים למעלה על מנת לקבל מספר שלם, כלומר נצטרך 5 ביטים על מנת לייצג 20 רמות אפור. במקרה זה יהיו רמות אשר לא יהיו בשימוש.

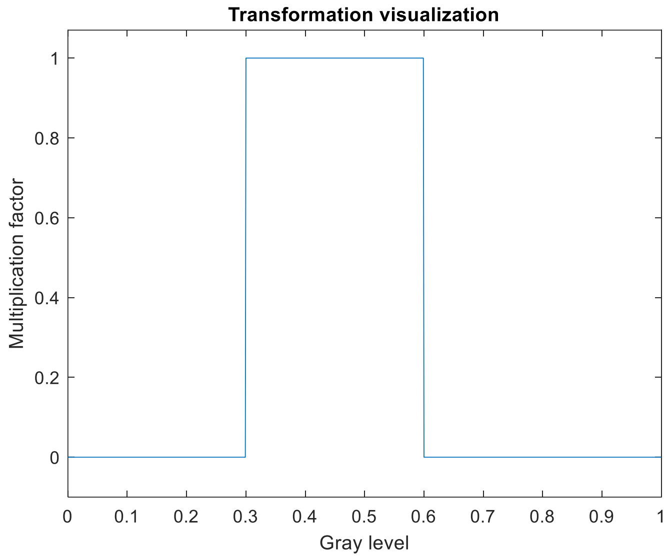
- חישבנו כי בתמונת RGB בעלת 256 רמות צריך 24 ביטים עבור פיקסל אחד. לכן על מנת לחשב את כמות הביטים הכולל לכל התמונה (1024X1024 פיקסלים), נכפול את מספר הפיקסלים הכולל במספר הביטים לכל פיקסל:

$$N = 24 \cdot 1024 \cdot 1024 = 25165824$$

1.2. שאלה 2

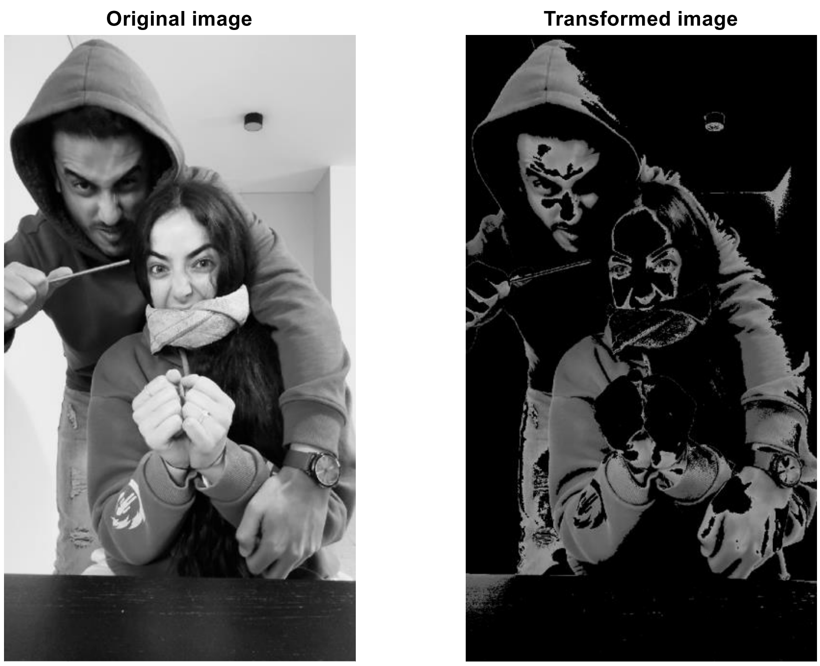
על מנת לשמור את כל ערכי האפור הנמצאים בין הערכים [0.3,0.6] בלבד ולאפס את שאר הפיקסלים מחוץ לטווח זה, נשתמש בטרנספורמציה הבאה:

$$T\{f[m,n]\} = \begin{cases} f[m,n], & 0.3 \leq f[m,n] \leq 0.6 \\ 0, & else \end{cases}$$



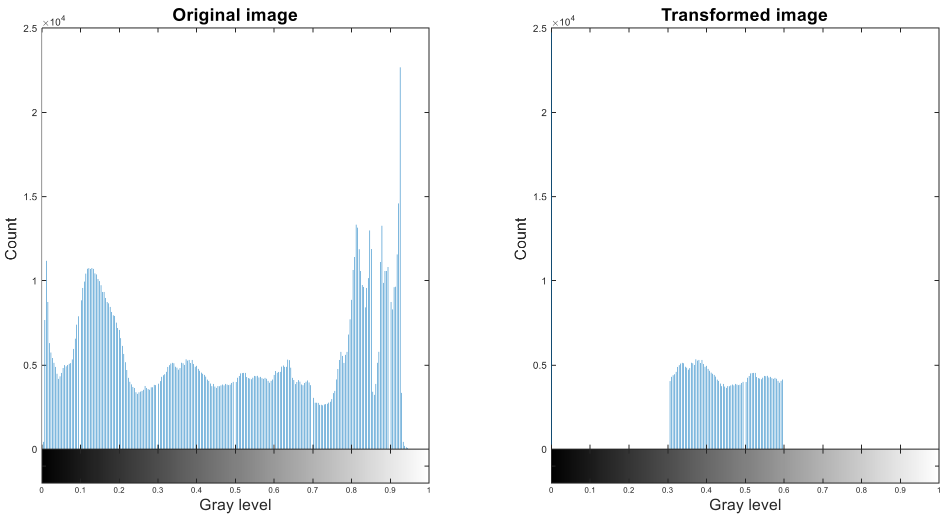
איור 1: איור הטרנספורמציה

איור זה מייצג את הטרנספורמציה אשר משמרת את הערכים (העברה מלאה) בתחום [0.3,0.6] ומאפסת את הערכים מחוץ לתחום זה.



איור 2: אנחנו לפני (שמאל) ואחרי הטרנספורמציה (ימין)

באיור זה מוצגות התמונות לפני ולאחר הטרנספורמציה. נבחין כי אכן ישנה השפעה על רמות ערכי האפור הגבוהות והנמוכות אשר התאפסו ולכן מוצגות בצבע שחור באיור.



איור 3 : תצוגת היסטוגרמה של התמונות לפני (שמאל) ולאחר הטרנספורמציה (ימין)

נבחין כי לאחר הטרנספורמציה, מספר הערכים בתחום $[0.3,0.6]$ שווה לערכו המקורי בעוד שהערכים מחוץ לתחום התאפסו ולכן אנו רואים עמודה גבוהה בערך 0.

1.3 שאלה 3

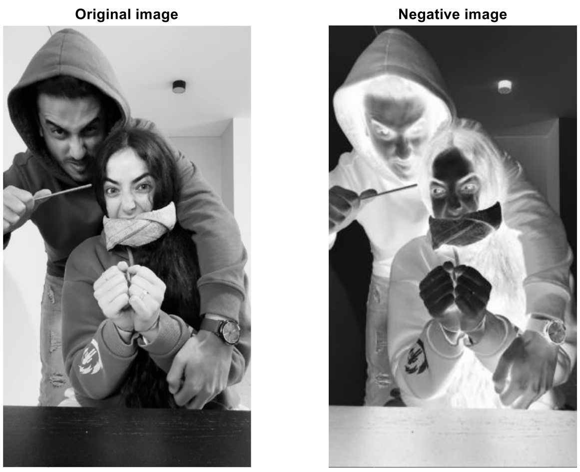
תמונת נגטיב הינה תמונה בה הצבעים מתהפכים, כך שהצבע השחור (0) הופך ללבן (1) ולהפך.
כלומר נשתמש בטרנספורמציה הבאה: $T\{f[m,n]\} = 1 - f[m,n]$

פונקציית המטלב:

```
function [N_IMG]=Negative(IMG)
% Inputs:
% IMG - Gray levels image
%
% Outputs:
% N_IMG - Negative image of IMG
N_IMG=1-IMG;
end
```

1.4 שאלה 4

נפעיל את הפונקציה שלעיל עבור התמונה המקורית:



איור 4 : התמונה המקורית (שמאל) ותמונת הנגטיב (ימין)

באיור זה ניתן לראות כי האזורים בהם גווני התמונה המקורית כהים יותר (כגון אזורי השיער) התהפכו והפכו לגוונים בהירים ולהפך.

1.5. שאלה 5

טרנספורמציה הפיכה מאפשרת להפעיל את הטרנספורמציה ההופכית שלה על ערכי המוצא על מנת לקבל את ערכי הכניסה.

$$T\{f[m,n]\} = g[m,n] \rightarrow T^{-1}\{g[m,n]\} = f[m,n]$$

התנאים ההכרחיים להפיכות הטרנספורמציה הם :

- חד חד ערכיות- לכל איבר בקבוצת הכניסה קיים איבר יחיד בקבוצת המוצא.
- תנאי הערכיות (על)- כל איבר בקבוצת המוצא הוא התמונה של לפחות איבר אחד בקבוצת הכניסה. משמע אין איברים בלתי מנוצלים בקבוצת המוצא.

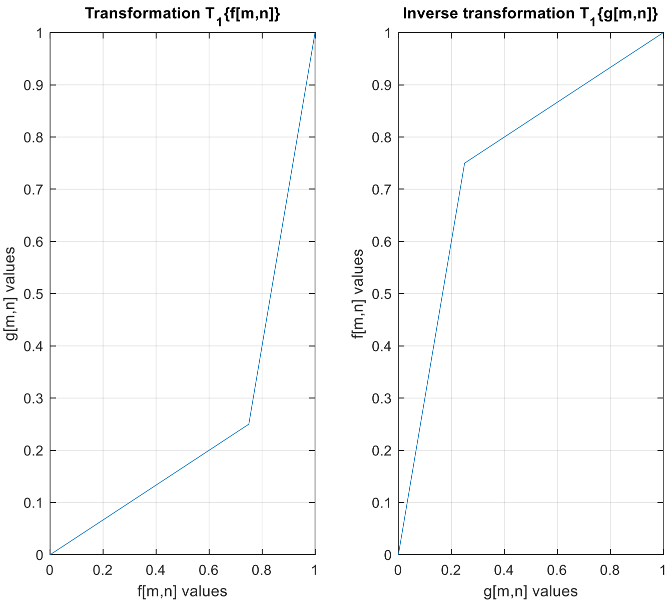
כאשר נשלב את שני תנאים הללו נקבל טרנספורמציה חח״ע ועל, כלומר הפיכה.

נקבע טרנספורמציה לינארית למקוטעין והפיכה :

$$T_1\{f[m,n]\} = g[m,n] = \begin{cases} \dfrac{f[m,n]}{3}, & f[m,n] < 0.75 \\ 3f[m,n] - 2, & f[m,n] \geq 0.75 \end{cases}$$

הטרנספורמציה ההופכית :

$$T_1^{-1}\{g[m,n]\} = f[m,n] = \begin{cases} 3g[m,n], & g[m,n] < 0.25 \\ \dfrac{g[m,n] + 2}{3}, & g[m,n] \geq 0.25 \end{cases}$$

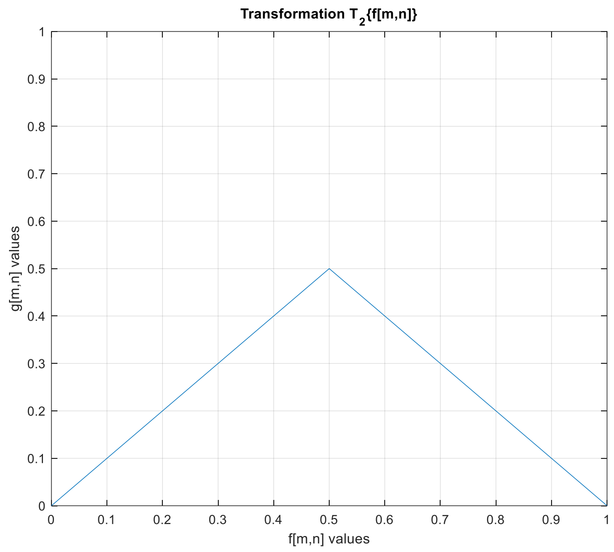


איור 5 : טרנספורמציה T_1 (שמאל) והטרנספורמציה ההופכית (ימין)

באיור זה ניתן לראות את הטרנספורמציה הלינארית למקוטעין שיצרנו. נשים לב כי בנקודה 0.75 בגרף משמאל שיפוע הטרנספורמציה משתנה. נבחין כי לכל ערך בסיגנל הכניסה f ישנו ערך יחיד בסיגנל המוצא g ולכל ערך בתמונת המוצא ישנה לפחות כניסה אחת. ניתן להבחין בזאת גם בעזרת גרף הטרנספורמציה ההופכית T_1^{-1} הנמצאת מימין כי לכל ערך כניסה g ישנו ערך מוצא f בצורה חד חד ערכית ועל.

כעת ניצור טרנספורמציה לינארית למקוטעין אך לא הפיכה:

$$T_2\{f[m,n]\} = g[m,n] = \begin{cases} f[m,n] \quad , & f[m,n] < 0.5 \\ 1 - f[m,n] \, , & f[m,n] \geq 0.5 \end{cases}$$



איור 6: הטרנספורמציה T_2

באיור זה ניתן לראות את הטרנספורמציה T_2 שיצרנו. נשים לב מהגרף כי אינה הפיכה, כך שישנם ערכי מוצא מסוימים (ערכים הקטנים מ-0.5) להם מתאימים 2 ערכי כניסה שונים. דבר זה גורר שאינה חד חד ערכית, כלומר אינה הופכית.

1.6. שאלה 6

Histogram Equalization זהו תהליך המשפר את הניגודיות של התמונה על ידי שינוי ההתפלגות בהיסטוגרמת התמונה המתארת את התפלגות רמות האפור בתמונה. מטרת השימוש ב-Histogram Equalization היא לשנות את התפלגות היסטוגרמת התמונה להתפלגות אחידה של רמות האפור. פעולה זו עשויה לשפר את ניגודיות התמונה, במיוחד כאשר התמונה מיוצגת על ידי טווח צר של גווני אפור.

הביטוי המתמטי ל-Histogram Equalization (גרסה בדידה) עבור כל ערך של רמת אפור בתמונה הינו:

$$s_k = \frac{(L-1)}{MN} \sum_{j=0}^{k} p_j$$

כאשר L - מספר רמות האפור האפשריות בתמונה. MN - מספר הפיקסלים האפשריים בתמונה בגודל $M \times N$. p_j - מספר הפיקסלים שערכם הוא j. s_k – ההיסטוגרמה החדשה, כלומר התמונה לאחר שימוש ב-Histogram Equalization.

הפקודה המתאימה לביצוע Histogram Equalization ב-MATLAB הינה histeq.

1.7. שאלה 7

עבור טרנספורמציית Histogram Equalization נסמן $T(a_n) = b_n$ כאשר a_n מייצג דרגת אפור. טרנספורמציה זו הינה חד חד-ערכית. כאשר נשתמש בטרנספורמציה פעם נוספת, כלומר $T(T(a_n))$ נקבל:

$$T(b_n) = \frac{(L-1)}{MN}\sum_{j=0}^{k} p_{b_n} = \frac{(L-1)}{MN}\sum_{j=0}^{k} p_{a_n} = T(a_n) = b_n$$

המעבר בהוכחה לעיל התאפשר מכיוון שהטרנספורמציה הינה חד חד-ערכית, $p_{a_n} = p_{b_n}$ ומכך קיבלנו כי $T(T(a_n)) = T(b_n) = b_n$.

1.8. שאלה 8

יצרנו פונקציה אשר מוסיפה תמונה של הקבוצה שלנו לתמונה שקיבלנו, לצידה של מנדי.

```
function [ResultPic]=AddToMandi(Picture)
% Inputs:
% Picture - Gray levels image
%
% Outputs:
% ResultPic - Mandi and us image

% Load background picture
background=imread("images\mandi.tif");
background=rgb2gray(background);
background=mat2gray(background);
% Remove white frame
frame=70;
background=background(frame:end-frame,frame:end-frame);
% Show the background
figure
imshow(background)

% Resize the input picture to be 0.8 time the background in the length
Picture=imresize(Picture,size(background,1)/size(Picture,1)/1.25);

% Creating the mask
mask=Picture;
ind=mask>0.8;
% 0 for brighter parts, 1 for darker parts
mask(ind)=0;
mask(~ind)=1;
% Extracting us using multiplication with the mask
Portrait=Picture.*mask;
% Show the portrait
figure
imshow(Portrait)

% Creating background mask to zero the portrait part
bgmask=ones(size(background));
bgmask(end-size(mask,1)+1:end,1:size(mask,2))=1-mask;
% The backgroung without the portrait parts (blackened)
background=background.*bgmask;

% continuation of the portrait in terms of length and width to mach the
% background size (blackened)
addPicture=zeros(size(background));
```

```
addPicture(end-size(Portrait,1)+1:end,1:size(Portrait,2))=Portrait;

% Adding the background and the portrait together
ResultPic=background+addPicture;
end
```

פונקציה זו תחילה טוענת את תמונת הרקע של מנדי, הופכת אותה לגווני אפור ומנורמלת, מורידה את המסגרת הלבנה ולבסוף מציגה אותה.

איור 7 : תמונת הרקע- מנדי במוזיאון

לאחר מכן הפונקציה משנה את מימדי התמונה של הקבוצה שלנו כך שיתאימו לחלק הספציפי בתמונת הרקע (לחלק השמאלי) ויוצרת את המסכה- גווני לבן (הרקע) משתנים להיות ערכי אפס וגווני שחור (אנחנו) משתנים להיות ערכי 1. לאחר מכן, אנו כופלים את התמונה המקורית שלנו במסכה על מנת לחלץ אותנו מהרקע.

$$Mask[m,n] = \begin{cases} 0\,, & OriginalImage[m,n] > 0.8 \\ 1\,, & OriginalImage[m,n] \leq 0.8 \end{cases}$$

איור 8 : חילוץ האנשים מהתמונה

באיור זה ניתן לראות כי הרקע נהפך לשחור בעוד שדיוקנינו נשאר כמו שהוא. נשים לב כי ישנם מקומות ספציפיים קטנים אשר האלגוריתם של המסכה זיהה כרקע (לדוגמת אזורים ביד ואזורים על הבגדים) מכיוון שהיו בהירים מידי.

לאחר מכן, אנו מאפסים את הפיקסלים הרלוונטיים בתמונת הרקע אשר יחפפו עם תמונתנו על מנת שלא תהיה חפיפה בערכים לאחר החיבור בין שתי התמונות. ולבסוף, נרחיב את התמונה שלנו בעזרת הוספת אפסים שיתאימו לגודל תמונת הרקע ונחבר ביניהם.

Stinky Mandi and us at the museum

איור 9 : מוצא הפונקציה- מנדי ואנחנו במוזיאון

באיור זה ניתן לראות את תמונת המוצא שיצאה מהפונקציה. נראה כי היה חיבור בין איור (7) לאיור (8) כך שאנו נמצאים על תמונת הרקע ביחד עם מנדי.

2. סינון לינארי ולא לינארי ותחום התדר:

2.1. שאלה 1

כאשר אנו מבצעים סינון מרחבי, אנו מבצעים קונבולוציה של אזורים בתמונה עם מטריצה מסדר נמוך בהשוואה לתמונה (למשל 3×3). בביצוע הקונבולוציה אנו מעבירים פילטר על כל פיקסל בתמונה ולמעשה מבצעים חישוב של ערך חדש לפיקסל בהתאם לערכים של הפיקסל הנתון והפיקסלים שסביבו. החישוב מתבצע באמצעות פונקציה שנקראת "קרנל" או "מסנן", שמגדירה את הדרך שבה הערכים של הפיקסלים הסמוכים משפיעים על הערך החדש של הפיקסל. אנו משתמשים בסינון מרחבי על מנת להשפיע על התכונות המרחביות של התמונה, כגון החלקה, הסרת רעשים, הדגשת קווי מתאר בתמונה ופעולות נוספות. הבעיה עם קצוות התמונה בעת ביצוע סינון מרחבי היא שהסינון מחייב סביבה של פיקסלים סביב הפיקסל אותו אנו מעוניינים לסנן. אולם, בקצוות התמונה, אין לנו מספיק פיקסלים סביב ולכן אנו נתקלים בבעיה כיצד להתמודד עם הפיקסלים שלא קיימים. על מנת לפתור בעיה זו, נוכל להיעזר בשיטות הבאות:

- המשכה מחזורית של התמונה.
- הרחבת התמונה, כלומר ניתן להוסיף שורות ועמודות של פיקסלים בקצוות התמונה. הערכים של הפיקסלים החדשים יכולים להיות 0 (שחור), 1 (לבן) או הערך של הפיקסל הקרוב ביותר בתמונה המקורית.
- התמקדות באזור המרכזי של התמונה בלבד. כלומר, ישנה התעלמות מהפיקסלים בקצוות התמונה והסינון יתבצע רק על האזור המרכזי, אשר יש מספיק פיקסלים של התמונה סביבו לצורך הסינון.

עבור הפתרון של הרחבת התמונה, היתרון של שיטה זו הוא שהיא פשוטה למימוש ומאפשרת לשמר את התמונה המקורית. החיסרון הוא שהיא יכולה ליצור תוצאות מעוותות, במיוחד כאשר הפיקסלים שנוספו אינם מתאימים לתמונה המקורית.

2.2. שאלה 2

אופרטור הלפלסיאן הוא כלי שמשמש כמסנן תדרים גבוהים (HPF) למטרת חידוד תמונה. האופרטור מתבצע על ידי בחינת פיקסל ביחס לשכניו, כאשר ההבנה היא שאם הפיקסל משתנה משכניו, זה יתבטא בתוצאת הקונבולוציה. אם לעומת זאת, הפיקסל אינו משתנה מהסביבה שלו, התוצאה של הקונבולוציה תהיה מספר שקרוב לאפס. כאשר אנו מדברים על תדר וסינון באמצעות HPF ו-LPF, אנו מתייחסים למהירות שינוי של ערכי הפיקסלים בתמונה. לפני שנבחן את המטריצה שמייצגת את האופרטור, נבחן את הפיתוח המתמטי של האופרטור והקשר שלו לנגזרת השנייה. אופרטור הלפלסיאן מוגדר כך [1]:

$$\Delta f(x,y) = \nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2} \tag{2}$$

מאחר ונגזרת ברצף מייצגת הפרשים בין ערכים סמוכים במרחק אינפיניטסימלי, במרחב התמונה היא מוגדרת על ידי ההפרש בין פיקסלים סמוכים באותה שורה או עמודה, תוך הנחה שהמרחק ביניהם הוא 1. כאשר אנו עומדים על הדגימה [m,n] ישנן שתי אפשרויות: גזירה אחורית וגזירה קדמית.

$$\frac{\partial f}{\partial x} \to f[m,n] - f[m-1,n] \quad or \quad f[m+1,n] - f[m,n]$$

משוואה זו עבור גזירה אופקית. באופן דומה ניתן להגדיר גזירה אנכית.

גזירה שנייה היא הפרש על הנגזרת עצמה. לכן, גזירה שנייה אופקית תהיה:

$$\frac{\partial^2 f}{\partial x^2} \to (f[m+1,n] - f[m,n]) - (f[m,n] - f[m-1,n])$$
$$\frac{\partial^2 f}{\partial x^2} \to f[m+1,n] + f[m-1,n] - 2f[m,n]$$

מכאן, אופרטור הלפלסיאן הדיסקרטי מתקבל על ידי:

$$\Delta f[m,n] = f[m+1,n] + f[m-1,n] + f[m,n+1] + f[m,n-1] - 4f[m,n]$$

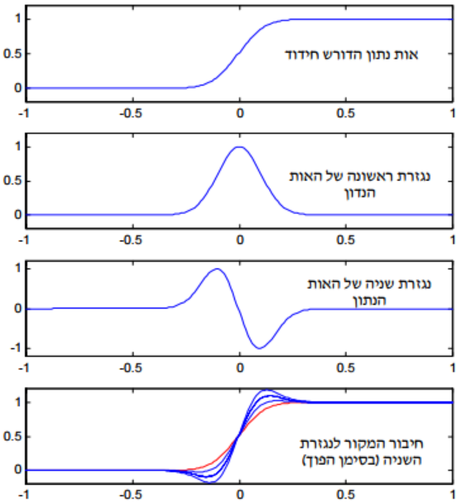
בצורה מטריציונית, המסנן ייכתב כך:

$$K_{Laplacian} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

השימוש במסנן הלפלסיאן שהוזכר לעיל אינו מתחשב בערכי הפיקסלים שנמצאים באלכסונים. לכן, ניתן להשתמש במטריצה אחרת שמשקללת את האלכסונים.

$$K^*_{Laplacian} = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

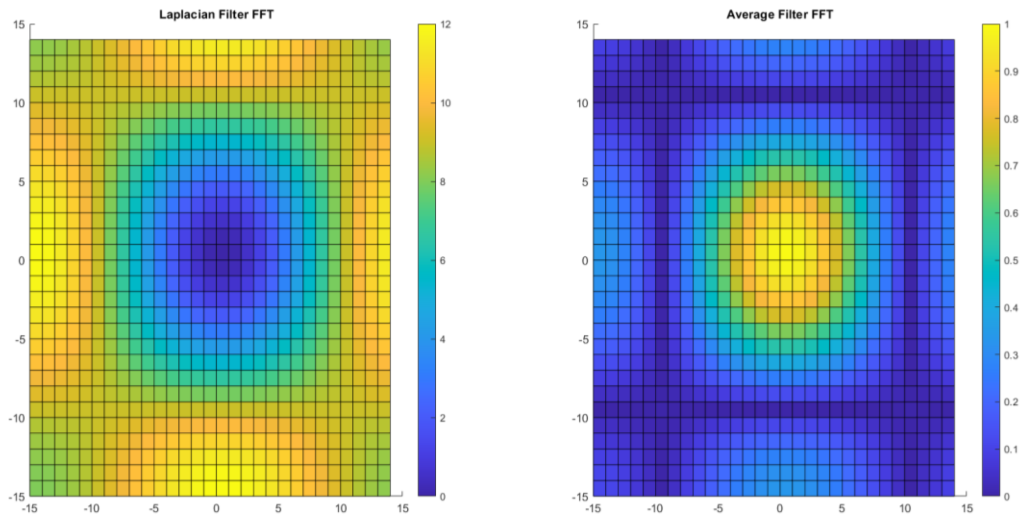
האיור הבא מסכם את התהליך וממחישו באופן מיטבי :



איור 10 : המחשת תהליך החידוד לאותות חד-ממדיים

2.3. שאלה 3

התבקשנו להציג את התמרת פורייה של שני המסננים (לפלסיאן ומיצוע) בעזרת MATLAB. לשם כך ריפדנו כל מסנן באפסים מסביב למטריצות לקבלת מטריצה 30×30 לפני ביצוע ההתמרה. להצגת התוצאות שהתקבלו נעזרנו בפונקציה 'surface' וקיבלנו כי :



איור 11 : מסנן לפלסיאן בתחום התדר

באיור לעיל ניתן לראות את המסננים (לפלסיאן ומיצוע) בתחום התדר, כפי שהתקבלו על ידי שימוש בפונקציה 'surface'. מהתבוננות בחלק השמאלי, בו מוצג מסנן לפלסיאן, ניתן להסיק כי מסנן זה הינו מסנן HPF. זאת משום שבמרכז התמונה, שם התדרים נמוכים, נראה צבע כחול המעיד על דיכוי התדרים הנמוכים, בעוד שבקצוות, שם התדרים גבוהים יותר, נראים צבעים בהירים כמו צהוב, המעידים על הדגשת התדרים הגבוהים.

לעומת זאת, בהתבוננות החלק הימני, בו מוצג מסנן מיצוע, ניתן לראות כי מסנן זה הינו מסנן LPF. ניתן להבחין כי נראית מגמה הפוכה. התדרים הנמוכים מודגשים בצהוב, משמע הדגשת התדרים הנמוכים ובפרט תדר ה-DC (מיצוע), והתדרים הגבוהים מופיעים בכחול, מה שמעיד על דיכוי התדרים הגבוהים. בנוסף, היות והמסנן LPF הינו מסנן ריבועי במרחב, התצוגה התדרית שלו היא sinc דו-מימדי.

הבחירה בין המסננים תלויה במטרת העיבוד :

- במקרה בו נרצה לשפר תמונה מטושטשת, נבחר להשתמש במסנן לפלסיאן. נסביר זאת ; שינויים מהירים בערכי פיקסלים בתמונה נחשבים לתדרים גבוהים, ושינויים איטיים נחשבים לתדרים נמוכים. תמונה מטושטשת מתאפיינת בכך שהשינויים בין ערכי הפיקסלים הם איטיים ולכן יש פחות תדרים גבוהים. כאשר מעוניינים לחדד תמונה, מטרת החידוד היא להבליט את השינויים המהירים הללו, כלומר להדגיש את הקצוות והפרטים בתמונה, שהם מקורות לתדרים גבוהים. לכן מסנן לפלסיאן, שהוא מסנן חדות המדגיש תדרים גבוהים יתאים יותר למקרה זה.
- במקרה בו נרצה לטשטש תמונה, כלומר שהשינויים בין ערכי פיקסלים בתמונה יהיו איטיים יותר, נבחר להשתמש במסנן מיצוע. מסנן זה ממצע את הערכים של הפיקסלים ובכך מטשטש פרטים.
- במקרה בו נרצה לזהות גבולות בתמונה, נבחר להשתמש במסנן לפלסיאן, שכן הוא מגביר תדרים גבוהים ומדגיש שינויים חדים בעוצמת האור, שהם אופייניים לגבולות התמונה.

2.4. שאלה 4

על מנת ליצור מסנן תדרי מהמסכות מן הסעיף הקודם עבור תמונה בגודל 512×512, נתחשב בגודל התמונה וברזולוציה שלה. ראשית, ניצור מטריצת קרנל שמייצגת את המסנן הרצוי (לפלסיאן/מיצוע). לאחר מכן, יש להפעיל את המסנן על התמונה באמצעות פעולת קונבולוציה. השלבים להפעלת המסנן על התמונה הם : המרת התמונה למישור התדר על ידי שימוש בהתמרת פורייה (FFT), יצירת מסנן תדרי באותו גודל כמו התמונה, כאשר המסנן מורכב מהמסכות שנבחרו, כפל אלמנט-אלמנט של התמונה במישור התדר עם המסנן התדרי והחזרת התמונה למישור המרחבי על ידי התמרת פורייה הפוכה (IFFT) כך שנקבל את התמונה החדשה לאחר סינון.

2.5. שאלה 5

ראשית, על מנת לטשטש תמונה נרצה להשתמש במסנן מיצוע, כפי שהסברנו בסעיף 2.3. נטשטש את התמונה בעזרת מסנני מיצוע בגדלים שונים : 5×5 , 4×4 , 3×3.

איור 12 : התמונה המקורית (שמאל עליון) והתמונות המטושטשות בעזרת מסנני מיצוע בגדלים שונים

באיור זה ניתן לראות את התמונה המקורית של lenna (שמאל עליון) ואת התמונות המטושטשות שהתקבלו לאחר שימוש במסנני מיצוע בגדלים שונים (5×5 , 4×4 , 3×3). נבחין כי גודל המסנן משפיע על רמת הטשטוש הנוצרת בתמונה, כך שמסנן גדול יותר מבצע מיצוע על טווח גדול יותר של פיקסלים בכל פעם ולכן התמונה שמתקבלת מטושטשת יותר.

נשפר את התמונה בדרך שהצענו בסעיף 2.4 באמצעות שימוש במטריצת לפלסיאן 3X3 :

איור 13 : התמונות שהתקבלו לאחר שימוש במסנן לפלסיאן

ניתן לראות כי התמונות שהתקבלו אכן חדות יותר, כלומר שיפרנו את איכות התמונה.

כעת נשפר את התמונה בעזרת שימוש בפונקציית fspecial המובנית ב-MATLAB.

איור 14 : התמונות שהתקבלו לאחר שימוש בפונקציית fspecial

ניתן לראות כי גם כאשר השתמשנו בפונקציית fspecial התמונות שהתקבלו ברורות יותר, כלומר שיפרנו את איכות התמונה. אולם, מהתבוננות בתמונות שהתקבלו לאחר שימוש בשתי השיטות, נראה כי קיים שוני בין התמונות ונראה כי קיבלנו תמונות חדות יותר בשימוש בלפלסיאן כפי שתיארנו בסעיף 2.4, כאשר קווי המתאר והקצוות בתמונה בולטים יותר לעין.

נציג את ערכי ה-MSE שחישבנו בעזרת ה-MATLAB :

טבלה 1 : השגיאות שהתקבלו משימוש ב-2 השיטות המתוארות

גודל המסנן	שגיאה עבור Laplacian filter	שגיאה עבור fspecial function
3×3	$18.7 \cdot 10^{-3}$	$2 \cdot 10^{-3}$
4×4	$14.8 \cdot 10^{-3}$	$2.7 \cdot 10^{-3}$
5×5	$9.4 \cdot 10^{-3}$	$2.3 \cdot 10^{-3}$

ניתן לראות כי שגיאות הסינון שהתקבלו לאחר שימוש בפונקציה fspecial המובנית ב-MATLAB קטנות מהשגיאות שהתקבלו לאחר שימוש במסנן לפלסיאן, עבור כל גדלי המסנן. בהתבסס על תוצאות אלו נאמר כי שימוש ב-fspecial function הינה הדרך הטובה יותר.

2.6. שאלה 6

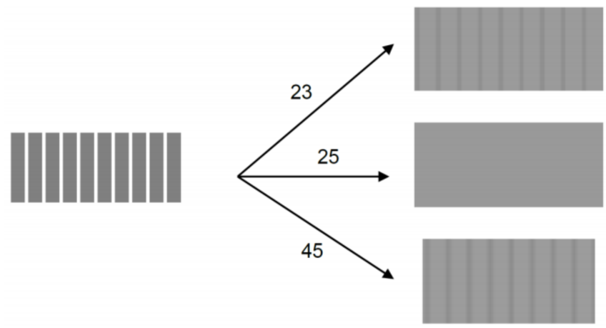
בסעיף זה התבקשנו לבטל אחוז מסוים (10,50,90) מרכיבי התדר הנמוכים בתמונה (לפי אנרגיה כללית), כלומר לבצע HPF עבור רכיבי התדר. בנינו מסנן HPF על ידי חיסור של מסנן LPF ממטריצת אחדות. את מסנן ה-LPF בנינו בעזרת מסנן גאוסיאן שהתקבל משימוש בפונקציית fspecial. מסנן הגאוסיאן המתאים לצורך המימוש נבחר כך שסטיית התקן שנבחרה היא זו שהתקבלה כאשר האנרגיה שקיבלנו הייתה שווה או קטנה מהאחוז הרצוי מתוך האנרגיה הכוללת. לאחר מכן, ביצענו התמרה הפוכה לקבלת התמונה. חזרנו על תהליך זה עבור איפוס 10%, 50% ו-90% מהתדרים. להלן התוצאות שהתקבלו :

איור 15 : התמונות שהתקבלו לאחר ביטול רכיבי התדר הנמוכים באחוזים שונים

באיור זה ניתן לראות כי ככל שאחוז הסינון עולה, כלומר כאשר האנרגיה הכוללת בעלת ערך קטן יותר, כך התמונה בגוונים כהים יותר. בנוסף, אפשר לזהות ירידה בזיהוי פרטים כלליים בתמונה. נשים לב כי כיוון שהתדרים הגבוהים נשמרו, אזי קווי המתאר וגבולות התמונה נשארים ברורים וחדים. דבר זה תואם לציפיותינו, כיוון שעל פי הידוע לנו התדרים הגבוהים הם אלו האחראיים לשינויים החדים ולגבולות בתמונה.

2.7. שאלה 7

בשאלה זו נתונה לנו תמונה אשר עברה סינון מרחבי ע״י 3 מסננים ממצעים באורך 23,25 ו-45 פיקסלים. התהליך ותוצאת הסינון מופיעים באיור הבא:



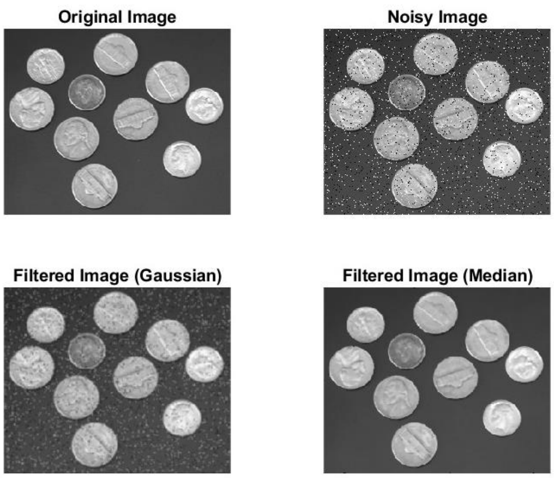
איור 16: התמונה הנתונה (משמאל) והתוצאות שהתקבלו עבור המסננים השונים (מימין)

מאיור זה ניתן לראות שבתוצאות הסינון עבור המסננים באורך 23 ו-45 פיקסלים עדיין ניתן להבחין בחלק מהפסים המקוריים, בעוד שעבור המסנן באורך 25 פיקסלים התמונה נראית אחידה וללא הבדלים בין הפיקסלים. תופעה זו נובעת מהתאמה בין אורך המסנן לתדירות הפסים בתמונה המקורית. כאשר אורך המסנן מתאים בדיוק לרוחב של מספר שלם של פסים, המסנן ממצע את הערכים באופן שמאחד את ההבדלים בין הפסים לרקע, ובכך יוצר תמונה אחידה ללא הבדלים ברורים. נראה שהמסנן באורך 25 פיקסלים, מתאים בדיוק למרווחים בין הפסים, ולכן התמונה המסוננת נראית אחידה. לעומת זאת, המסננים באורך 23 ו-45 פיקסלים אינם מתאימים באותה מידה לתדירות הפסים, לא מחליקים את התמונה באותה המידה, וכתוצאה מכך ניתן להבחין בפסים לאחר הסינון.

2.8. שאלה 8

נממש את הפונקציה CleanSP הבאה:

```
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
```



איור 17: תוצאות הסינון שהתקבלו לאחר שימוש במסננים השונים- גאוסיאן (שמאל תחתון) וחציון (ימין תחתון)

באיור זה ניתן לראות כי לאחר שימוש במסנן חציון התוצאה שהתקבלה טובה יותר. תוצאה זו מתיישבת עם הידוע לנו על אופי המסננים ואופי הרעש שהתווסף. רעש מסוג salt&pepper מאופיין בהפרעות חדות ופתאומיות בתמונה, המופיעות כפיקסלים לבנים ושחורים המופיעים באופן אקראי, כפי שניתן לראות בתמונה המורעשת המופיעה באיור לעיל (ימין עליון). לכן, מסנן החציון יעיל יותר לסינון רעש מסוג זה, זאת מכיוון שמסנן זה מחליף כל ערך פיקסל בתמונה בערך החציוני של הפיקסלים השכנים. החציון אינו רגיש לערכים קיצוניים אשר אופייניים לסוג רעש זה ולכן מצליח להסירם מבלי להשפיע באופן משמעותי על הפיקסלים הסובבים וכך לשמר את נראות התמונה. לעומת זאת, מסנן גאוסי מבצע מיצוע משוקלל של ערכי הפיקסלים בעזרת גרעין גאוסי, הוא מתחשב בערכים שכנים לערך הרעש ולכן אינו יעיל בסינון רעשים המאופיינים בערכים קיצוניים המאפיינים רעש מסוג salt&pepper [1].

3. פעולות מורפולוגיות (בתמונות שחור לבן):

3.1 שאלה 1

אלגוריתם "Hit-or-Miss" הינו טרנספורמציה מורפולוגית עבור זיהוי תבניות וצורות בעיבוד תמונה בינארית [2]. תמונה בינארית הינה תמונה המורכבת מפיקסלים אשר ערכם יכול להיות 1, 0 או don't care. השימוש בטרנספורמציה זו נועד בעיקר לזיהוי קבוצות פיקסלים קטנה מסוימות המבודדת משאר חזית התמונה. התבנית שנרצה לזהות תהיה בתצורת שכנות אשר מורכבת מפיקסלים שהם חלק מהחזית של התמונה ופיקסלים שהם חלק מהרקע. ניתן להתייחס אליהם כאל שתי קבוצות של פיקסלים, החזית והרקע שהן לא חופפות אחת עם השנייה. לשתי קבוצות אלו תהיה נקודת ראשית אחת. מריצים את התבנית על פני כל התמונה ובודקים האם הקבוצה של החזית תואמת למערך הפיקסלים שמוכל בחזית התמונה הכוללת בעוד שקבוצת הרקע אינה תואמת כלל לקבוצת חזית התמונה. על כן שם הטרנספורמציה הוא "Hit-or-Miss", כאשר קבוצה אחת "פוגעת" בחזית והקבוצה השנייה "מפספסת" את חזית התמונה.
כאשר אנו מוצאים מערך פיקסלים אשר תואם למערך התבנית אותה נרצה למצוא, הטרנספורמציה תהפוך את הפיקסל בראשית התבנית להיות 1 ואת שאר הפיקסלים תסמן כ-0.
נוסחת הטרנספורמציה:

$$HMT_B(X) = \{x|(B_1)_x \subseteq X\ ,(B_2)_x \subseteq X^c \tag{3}$$

כאשר X הינה קבוצת החזית של כלל התמונה, B קבוצת התבנית הרצויה בעל נקודת ראשית אחת אשר מורכבת משתי תתי קבוצות $B_1 \coloneqq foreground\ ,B_2 \coloneqq background$.
באמצעות נוסחה זו, לוקחים את כל הנקודות אשר מקיימות כי קבוצת חזית התבנית מוכלת בקבוצת החזית הכוללת וגם קבוצת רקע התבנית מוכלת בקבוצת החזית המשלימה (קבוצת רקע התמונה).

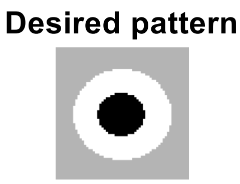
3.2 שאלה 2

נבנה פונקציה המאתרת תבניות רצויות בתוך תמונה:

```
function [n,Coordinates]=LocateCirc(IMG)
% Inputs:
% IMG - Gray levels image
%
% Outputs:
% n - number of identified patterns.
% Coordinates - a vector with the XY coordinates of the
% required pattern [ n * 2 (X Y)]

% Extracting the pattern by eye
origin=[48 153];
bg=30;
pattern=IMG(origin(1)-bg:origin(1)+bg,origin(2)-bg:origin(2)+bg);
% Showing the pattern
figure
```

```
imshow(pattern)
title('Desired pattern')

% Finding all reacurent patterns in the image
hit=false(size(IMG));
% Loop of all pixels in the image (within a frame of 30 pixels high and
% wide)
for x=bg+1:size(IMG,1)-bg
    for y=bg+1:size(IMG,2)-bg
        % Comparisson of the pattern and the appropriate size array of a
        % certain part of the image. Calculation of the number of matching
        % pixels and comparisson to the total number of pixels in the
        % pattern. If matching then True, else False
        hit(x,y)=nnz(IMG(y-bg:y+bg,x-bg:x+bg)==pattern)==(2*bg+1)^2;
    end
end
% Calculating the number of matches
n=nnz(hit);
% Creating coordinates vector
[X,Y]=find(hit);
Coordinates=[X;Y];

% Showing the original image and the identified patterns marked as red X
figure
imshow(IMG)
hold on
scatter(X,Y,[],"red",Marker="x")
title('Original image and identified patterns')
legend('Patterns')
end
```

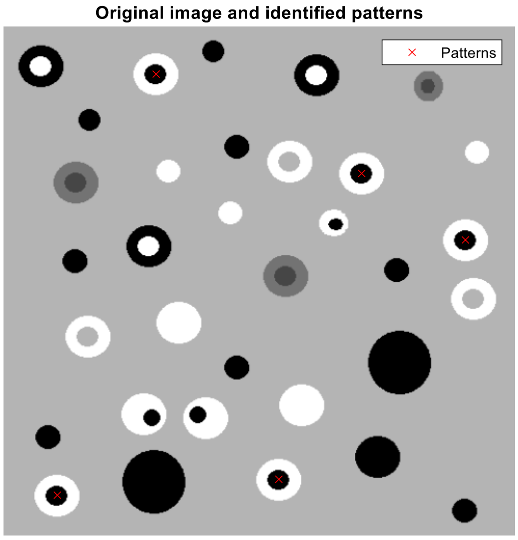
פונקציה זו מקבלת תמונה בגווני אפור ופולטת את מספר הפעמים בהן התבניות נמצאו בתמונה ואת מיקומם בווקטור קואורדינטות. תחילה, הפונקציה מייצרת את התבנית אותה נרצה למצוא:



איור 18: התבנית הרצויה

ניצור לולאה אשר רצה לאורך ולרוחב התמונה, כאשר ישנו מרווח של 30 פיקסלים מהקצוות על מנת לאפשר השוואה בין התבנית בעלת 61X61 פיקסלים למקטע התמונה שניקח. עבור כל ערך בלולאה, נשווה בין המקטע הספציפי לתבנית, נספור את מספר הפיקסלים התואמים לתבנית ולבסוף נשווה למספר הפיקסלים הכולל של התבנית (3721). נעשה זאת על מנת לוודא כל כלל הפיקסלים במקטע שנלקח יהיו חופפים לתבנית הרצויה.

הפונקציה מחשבת את כמות התבניות שנמצאו ואת ווקטור הקואורדינטות ומציגה את התמונה המקורית ועליה סימון התבניות שנמצאו:

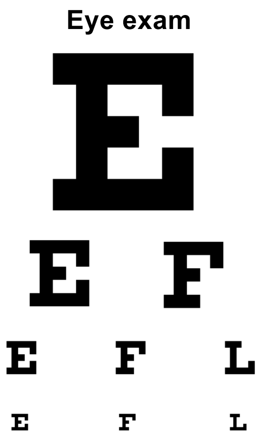


איור 19: התמונה המקורית והתבניות שנמצאו

באיור זה ניתן לראות את התמונה המקורית שעליה מסומנות התבניות שנמצאו על ידי סימון איקס אדום. נבחין כי זיהינו את כלל המקטעים התואמים לתבנית באיור (18) ולא קיים זיהוי מוטעה.

3.3. שאלה 3

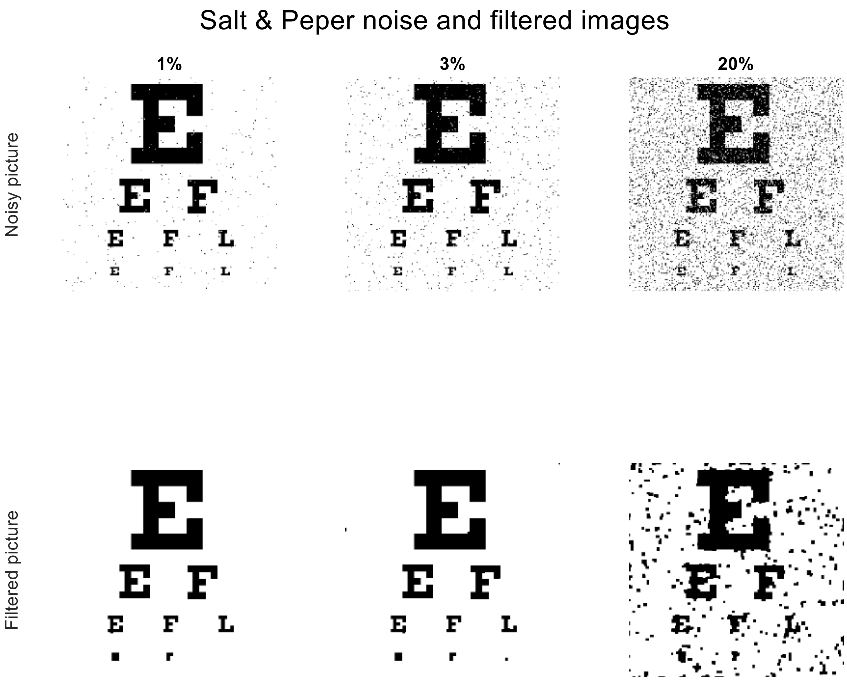
נציג תחילה את התמונה הנקייה:



איור 20: התמונה הנתונה לפני הוספת הרעש הסינטטי

באיור זה ניתן לראות את תמונת בדיקת הראיה הנקייה. נבחין כרואי 6/6 כי כל האותיות בכל רמה ברורות וחדות.

הוספנו בצורה סינטטית רעש "Salt and Peper" לתמונה באחוזי צפיפות שונים ולאחר מכן סיננו את הרעש באמצעות פעולות מורפולוגיות של פתיחה (dilation) המלווה בסגירה (erosion) עם אלמנט של מטריצת אחדות בגודל 3X3. פעולות מורפולוגיות אלה משומשות בעיקר לסינון תמונות כאשר פתיחה גורמת להחלקת התמונה ולהסרה של גורמים קטנים בעוד שסגירה גורמת למילוי חורים קטנים בתמונה.



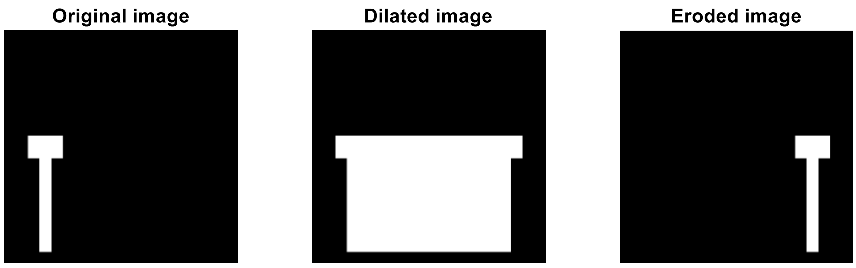
איור 21 : תמונות בדיקת הראיה המורעשות ותוצאות הסינון

באיור זה ניתן לראות את התמונות המורעשות ברעש ה-salt&paper בשורה העליונה ואת סינונם בשורה התחתונה בהתאמה. נבחין כי לאחר הפתיחה המלווה בסגירה, התמונות עם האחוזים הנמוכים של הרעש (שמאל ואמצע) סוננו באופן יעיל. האותיות הגדולות סוננו בהצלחה עד כדי הוספת פיקסלים שחורים בודדים לכל אות, אך למרות זאת האותיות הקטנות אכן סוננו מהרעש אך צורתם השתנתה בצורה דרמטית. לדוגמא האות E התמלאה וכעת נראית כריבוע שחור והאות L בשורה התחתונה נעלמה כליל. זאת מכיוון שהאלמנט שבחרנו, מטריצת אחדות בגודל 3X3, אינו מתאים בדיוק לצורת האות ולכן האותיות הקטנות סוננו בצורה חלקית. עבור אחוז הרעש הגבוה (ימין) נשים לב כי הסינון לא עבר בהצלחה. מכיוון שאחוז הרעש היה גדול מאוד, יותר פיקסלי רעש היו מקובצים, ולכן האלמנט בחרנו התאים גם לאזורי הרעש, כלומר הרעש לא הונחת בסינון.

על מנת להנחית את אחוזי הרעש הגבוהים בצורה מיטבית יותר, נצטרך להשתמש באלמנט ספציפי ויעודי לתמונה כך שיתאים אך ורק לאזורי התמונה המקורית ויהיה פחות סיכוי שיתאים לאזורי הרעש. דרך זו תסנן את הרעש בצורה יעילה, אך מכיוון שהאלמנט יהיה ספציפי, במקרה זה עלולה להעביר רק סוג וגודל מסוים של אותיות.

3.4. שאלה 4

בשאלה זו יצרנו תמונה בינארית של stick-man הנמצא בצד שמאל של התמונה ובעזרת פעולות מורפולוגיות נזיז אותו לצד ימין של התמונה.



איור 22 : תמונת ה-stick-man שיצרנו ותוצאות הפעולות המורפולוגיות

באיור זה ניתן לראות את התמונה המקורית שיצרנו (שמאל), לאחר dilation (אמצע) ולאחר erosion (ימין). בחרנו באלמנט ה-dilation להיות מטריצה 3X3: $\begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{pmatrix}$ על מנת ליצור הרחבה לימין ואלמנט ה-erosion להיות מטריצה 3X3: $\begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$ על מנת ליצור צמצום משמאל. תחילה הפעלנו את פעולת ה-dilation 13 פעמים על מנת להרחיב את הדמות עד צד ימין ולאחר מכן הפעלנו את פעולת ה-erosion 13 פעמים על מנת לצמצם את הדמות להיות דומה לדמות המקורית.

3.5. שאלה 5

נכתוב פונקציה התאפשר לייצר סדרת תמונות בתנועה של stick-man:

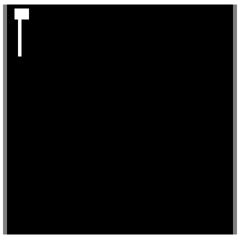
```
function [M]=IMove(Track)
% Input:
% Track - a vector [n x 2] contains movement instruction for
% stick-man. the first column contains axis:
% 1=movement in the X-axis
% 2=movement in the Y-axis
% the second column contains number of steps in the same
% direction. %
% Output:
% M - a Matrix [50 x 50 x (n*steps)] contains the movements
% step-by-step.

% Creating the stick-man image
M=zeros(50);
M(2:11,4)=1;
M([2 3],[3 5])=1;

% Plotting the steps (f1) and an animated gif (f2)
f1=figure;
f2=figure;
imshow(M)
exportgraphics(f2,'SM50.gif','Append',true);

% Loops to run the dilation and erosion according to track
for i=1:length(Track)
    if Track(i,1)==1
        % X-axis movement
        SE=[0 0 0;0 1 1;0 0 0];
        if Track(i,2)<0
            % If the movement number is negative, flip the SE matrix
            SE=fliplr(SE);
        end
    else
        % Y-axis movement
        SE=[0 0 0;0 1 0;0 1 0];
        if Track(i,2)<0
            % If the movement number is negative, flip the SE matrix
            SE=flipud(SE);
        end
    end
    % plot the steps
    set(0,'CurrentFigure',f1)
    subplot(2,ceil(length(Track)/2),i)
    imshow(M)
```

```
    title([num2str(i) '/10'])

    % Dilation and erosion on the image and plotting the gif
    set(0,'CurrentFigure',f2)
    for j=1:abs(Track(i,2))
        M=imdilate(M,SE);
        imshow(M)
        exportgraphics(f2,'SM50.gif','Append',true);
    end
    % Erosion while SE is flipped 180 degrees
    for j=1:abs(Track(i,2))
        M=imerode(M,rot90(rot90(SE)));
        imshow(M)
        exportgraphics(f2,'SM50.gif','Append',true);
    end
end
% Plot the last step
set(0,'CurrentFigure',f1)
subplot(2,ceil(length(Track)/2),10)
imshow(M)
title('10/10')
f1.WindowState="fullscreen";
end
```

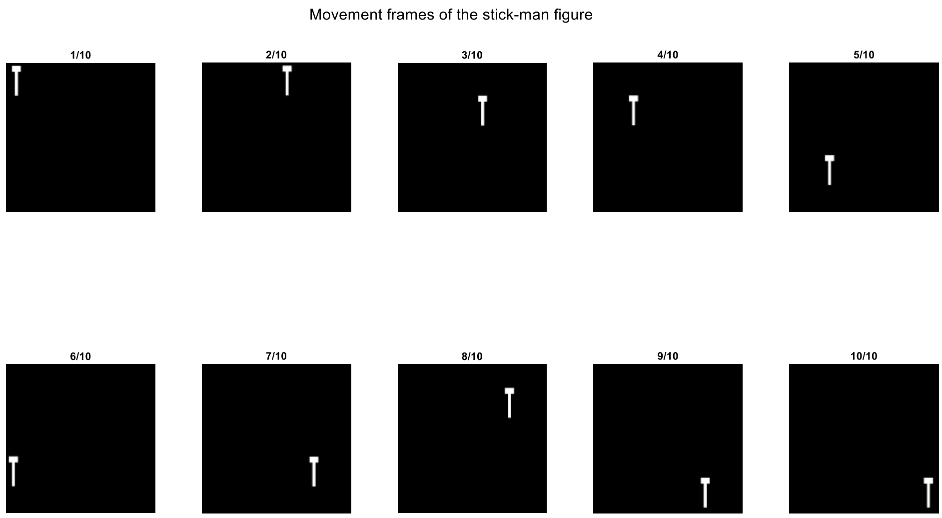
פונקציה זו מקבלת מטריצת שלבי תזוזה של דמות ה-stick-man שיצרנו בתחילת הקוד ופולטת את התמונה הסופית לאחר כל ההזזות. בדומה לסעיף קודם, נבצע את ההזזות בעזרת dilation ו-erosion על מנת להרחיב את הדמות בכיוון הרצוי ולצמצם את הדמות בכיוון ההפוך בהתאמה. מטריצת אלמנטי ה-SE עבור תזוזת הדמות בכיוונים החיובים של צירי ה-X וה-Y הינם:

$$X_{movement} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{pmatrix}, \qquad Y_{movement} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

במידה ומספר התזוזה הינו שלילי, נזיז את הדמות בכיוון ההפוך על ידי שינוי מטריצת אלמנט ה-SE לכיוון המתאים (נסובב את המטריצה ב-180 מעלות).



איור 23: גיף אנימציה של תזוזת הדמות



איור 24 : תמונות הפריימים של תזוזת הדמות

באיורים לעיל ניתן לראות את תזוזת הדמות. בחרנו להציג את התזוזה בעזרת מטריצת התזוזה הבאה :

$$Track = \begin{pmatrix} 1 & 25 \\ 2 & 10 \\ 2 & -15 \\ 2 & 20 \\ 1 & -11 \\ 1 & 35 \\ 2 & -23 \\ 2 & 30 \\ 1 & 9 \end{pmatrix}$$

באיור (23) ניתן לראות את פעולת ההרחבה והכיווץ שיצרנו על מנת להגיע למיקום הסופי לאחר כל שורה ב-track ובאיור (24) נראה את המיקום הסופי של הדמות לאחר כל שורה.

3.6. שאלה 6

א. במקרה ונרצה לסמן את הפיקסל שהוא או לפחות אחד משני הפיקסלים מימינו שווים ל-1, נשתמש במטריצת אלמנט שהיא $\begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$ כאשר פיקסל הראשית יהיה במקום (3,3).

ב. במקרה אם בתמונה יש ריבוע לבן בגודל 2X2, על מנת ליצור בתמונת המוצא ריבוע לבן בגודל 4X4, נשתמש במטריצת האלמנט הבאה : $\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$ כאשר פיקסל הראשית נמצא במרכז המטריצה במיקום (2,2).

4. זיהוי קצוות:

4.1 שאלה 1

אופרטור sobel הינו אופרטור המיישם מדידת גראדיאנט מרחבי דו ממדי על תמונה [3]. משתמשים באופרטור זה על מנת לזהות קצוות (שינויים גבוהים במגניטודת הגראדיאנט) בתמונה הרצויה. גרדיאנט התמונה מוגדר כך:

$$\nabla f(x,y) = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \tag{4}$$

על מנת לחשב כל אחד מאיברי הגרדיאנט בתמונה הדיגיטלית, נשתמש בזוג גרעיני קונבולוציה:

$$G_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}, \qquad G_y = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$$

המטריצות הן סיבוב אחת של השנייה ב-90 מעלות. כאשר נפעיל קונבולוציה על התמונה עם 2 גרעינים אלו בנפרד, נחשב את איברי הגרדיאנט. כעת באופרטור sobel אנו מחשבים את אמפליטודת הגרדיאנט כך:

$$|G| = \sqrt{G_x^2 + G_y^2} \approx |G_x| + |G_y| \tag{5}$$

בעזרת חישוב זה נקבל ערך ייחודי לכל פיקסל בתמונה. חישוב שיערוך האמפליטודה מוריד קצב חישוב בכך שהינו חישוב פשוט יותר, כלומר שימוש בשערוך זה מתבצע בעיקר בעיבוד תמונות בזמן אמת.

ניתן לחשב גם את זווית הגרדיאנט בצורה הבאה:

$$\theta = \arctan\left(\frac{G_y}{G_x}\right) \tag{6}$$

אך חישוב זה הינו אופציונלי, רוב המידע הנוגע לזיהוי קצוות נמצא באמפליטודת הגרדיאנט. ניתן יהיה להשתמש בזווית הגרדיאנט על מנת למצוא את כיוון השינוי המשמעותי בתמונה.

4.2 שאלה 2

Laplacian of Gaussian Method הינה שיטה לזיהוי קצוות. נשתמש בנוסחת הלפלסיאן המוצגת בנוסחה (2) ואת חישובה בצורה דיסקרטית, כפי שהצגנו בסעיף 2.2, באמצעות קונבולוציה עם אחד מגרעיני הקונבולוציה הבאים:

$$\begin{pmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{pmatrix}, \qquad \begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}$$

גרעינים אלו הם העיקריים ביותר שמשתמשים בהם לחישוב הלפלסיאן.

הלפלסיאן הינו חישוב של נגזרת מרחבית שנייה של תמונה ולכן נותן לנו מידע על שינוי מהיר מאוד בה. לעיתים קרובות משתמשים תחילה בפילטר גאוסיאני על מנת להפחית את הרגישות לרעש של הלפלסיאן. על מנת להפחית סיבוכיות חישובית, ניתן תחילה לבצע קונבולוציה בין הלפלאסיאן לבין פילטר הגאוסיאן, שהן לרוב מטריצות הקטנות בהרבה ממטריצת התמונה, ונקבל פילטר חדש המשלב ביניהם [4].

$$LoG(x,y) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right]e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{7}$$

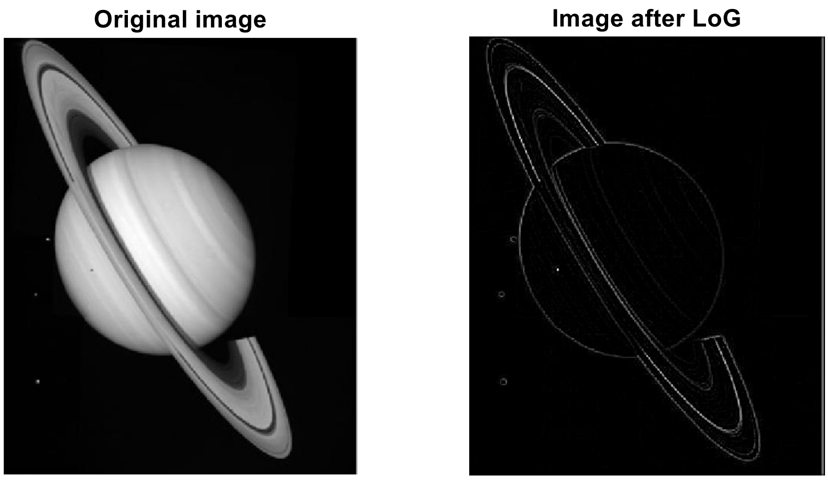
ניתן לשערך משוואה זו באמצעות גרעין בדיד. כאשר סטיית התקן הינה 1.4 :

0	1	1	2	2	2	1	1	0
1	2	4	5	5	5	4	2	1
1	4	5	3	0	3	5	4	1
2	5	3	-12	-24	-12	3	5	2
2	5	0	-24	-40	-24	0	5	2
2	5	3	-12	-24	-12	3	5	2
1	4	5	3	0	3	5	4	1
1	2	4	5	5	5	4	2	1
0	1	1	2	2	2	1	1	0

איור 25 : מטריצת שיערוך Laplacian of Gaussian עם סטיית תקן 1.4

באיור זה ניתן לראות את מטריצת הגרעין בה נשתמש על מנת ליישם את Laplacian of Gaussian Method.

ניישם שיטה זו על תמונה לבחירתנו :



איור 26 : התמונה לפני ולאחר מעבר בפילטר LoG

באיור זה ניתן לראות את המוצא של התמונה לאחר מעבר בפילטר LoG. נשים לב כי קצוות הכוכב שהן בעלות השינויים המקסימליים חודדו ונצבעו בלבן בעוד שהשינויים הנמוכים נצבעו בשחור. במעבר במסנן זה זיהינו את קצוות העצמים בתמונה דבר אשר עלול להיות כלי טוב לזיהוי וסיווגם.

נספחים:

קוד המטלב

```matlab
%% 1.2
% Load the picture
RawIMG=imread('IMG_0989.jpg');
% Convert to grayscale
GrayIMG=rgb2gray(RawIMG);
% Normalize
NormIMG=mat2gray(GrayIMG);
TransIMG=NormIMG;

% Transformation visualization
transformation=ones(1,1001);
transformation([1:300 601:1001])=0;
figure
plot(0:0.001:1,transformation)
title('Transformation visualization')
xlabel('Gray level')
ylabel('Multiplication factor')
ylim padded

% Show img
figure
subplot(121)
imshow(NormIMG)
title('Original image')

% Transformation
TransIMG(NormIMG<=0.3|NormIMG>=0.6)=0;
% Show img
subplot(122)
imshow(TransIMG)
title('Transformed image')

% Histograms
f=figure(WindowState="maximized");
subplot(121)
imhist(NormIMG)
title('Original image',FontSize=18)
xlabel({'','','Gray level'},FontSize=16)
ylabel('Count',FontSize=16)
ylim([0 2.5e4])

subplot(122)
imhist(TransIMG)
title('Transformed image',FontSize=18)
xlabel({'','','Gray level'},FontSize=16)
ylabel('Count',FontSize=16)
ylim([0 2.5e4])

%% 1.4
% Function activation
NegIMG=Negative(NormIMG);
% Plot Negative image
figure
subplot(121)
imshow(NormIMG)
```

```
title('Original image')

subplot(122)
imshow(NegIMG)
title('Negative image')

%% 1.5
% Partialy linear inversible transformation
f=0:0.01:1;
g=zeros(size(f));

ind=f<=0.75;
g(ind)=f(ind)/3;
g(~ind)=3*f(~ind)-2;
% Plotting the transformation
figure
subplot(121)
plot(f,g)
title('Transformation T_1\{f[m,n]\}')
xlabel('f[m,n] values')
ylabel('g[m,n] values')
grid on

subplot(122)
plot(g,f)
title('Inverse transformation T_1\{g[m,n]\}')
xlabel('g[m,n] values')
ylabel('f[m,n] values')
grid on

% Partialy linear non-inversible transformation
f=0:0.01:1;
g=zeros(size(f));

ind=f<0.5;
g(ind)=f(ind);
g(~ind)=1-f(~ind);
% Plotting the transformation
figure
plot(f,g)
title('Transformation T_2\{f[m,n]\}')
xlabel('f[m,n] values')
ylabel('g[m,n] values')
ylim([0 1])
grid on

%% 1.8
% Load the picture
RawIMG=imread('IMG20240117191010~7.jpg');
% Convert to grayscale
GrayIMG=rgb2gray(RawIMG);
% Normalize
mandiIMG=mat2gray(GrayIMG);

% Function activation
UsAndMandi=AddToMandi(mandiIMG);

% Plotting the result
figure
```

```
imshow(UsAndMandi)
title('Stinky Mandi and us at the museum')

%% 2.3
% Define Matrices for the filters
Lap_filter = [-1 -1 -1; -1 8 -1; -1 -1 -1];
avg_filter = (1/9) * ones(3, 3);

% Correct the padding to create a 30x30 matrix
padded_Lap = padarray(Lap_filter, [13 13], 'post');
padded_avg = padarray(avg_filter, [13 13], 'post');

padded_Lap = padarray(padded_Lap, [14 14], 'pre');
padded_avg = padarray(padded_avg, [14 14], 'pre');

% FFT on the padded filters
fft_laplacian = fft2(padded_Lap);
fft_average = fft2(padded_avg);

% Display the magnitude using surface
figure(WindowState="maximized");
subplot(121)
surface(-15:14,-15:14, abs(fftshift(fft_laplacian)));
title('Laplacian Filter FFT');
colormap;
colorbar;

subplot(122)
surface(-15:14,-15:14, abs(fftshift(fft_average)));
title('Average Filter FFT');
colormap;
colorbar;

%% 2.5
original=double(imread('images\lenna.jpg'))/255;
%imshow(original)

% Creating the average filters of different sizes by fspecial function
H3=fspecial('average');
H4=fspecial('average',[4 4]);
H5=fspecial('average',[5 5]);

% Blurring the image - apply the filter with zero padding
blurred3 = imfilter(original, H3, 'same');
blurred4 = imfilter(original, H4, 'same');
blurred5 = imfilter(original, H5, 'same');

%plotting original and blurred pictures
figure
subplot(2,2,1)
imshow(original);
title('Original image')
subplot(2,2,2)
imshow(blurred3);
title('Blurred lenna using 3x3 average filter')
subplot(2,2,3)
imshow(blurred4);
title('Blurred lenna using 4x4 average filter')
subplot(2,2,4)
```

```
imshow(blurred5);
title('Blurred lenna using 5x5 average filter')

% Using laplacian filter to improve the picture quality
% Convert the image to the frequency domain
F_blu3 = fft2(double(blurred3));
F_blu4 = fft2(double(blurred4));
F_blu5 = fft2(double(blurred5));
% Create the frequency domain filter (same size as the image)
F_filter = fft2(double(Lap_filter),size(original,1),size(original,2));
% Apply the filter in the frequency domain
F_filtered3 = F_blu3 .* F_filter;
F_filtered4 = F_blu4 .* F_filter;
F_filtered5 = F_blu5 .* F_filter;
% Convert back to the spatial domain
filtered_imageB3 = ifft2(F_filtered3);
filtered_imageB4 = ifft2(F_filtered4);
filtered_imageB5 = ifft2(F_filtered5);

%plotting original and the pictures after improving (using laplacian
%filter)
figure
subplot(2,2,1)
imshow(original);
title('Original image')
subplot(2,2,2)
imshow(filtered_imageB3+original);
title('Filtered lenna for Blurred3')
subplot(2,2,3)
imshow(filtered_imageB4+original);
title('Filtered lenna for Blurred4')
subplot(2,2,4)
imshow(filtered_imageB5+original);
title('Filtered lenna for Blurred5')

% improving the picture using fspecial function
LapF=fspecial('laplacian');
% improve the image - apply the filter
fil3 = imfilter(blurred3, LapF);
fil4 = imfilter(blurred4, LapF);
fil5 = imfilter(blurred5, LapF);

%plotting original and the pictures after improving (using laplacian
%filter by fspecial function)
figure
subplot(2,2,1)
imshow(original);
title('Original image')
subplot(2,2,2)
imshow(fil3+original);
title('Filtered lenna for Blurred3')
subplot(2,2,3)
imshow(fil4+original);
title('Filtered lenna for Blurred4')
subplot(2,2,4)
imshow(fil5+original);
title('Filtered lenna for Blurred5')

% MSE calculation
```

```
% for laplacian
err1 = immse(blurred3,filtered_imageB3+original);
err2 = immse(blurred4,filtered_imageB4+original);
err3 = immse(blurred5,filtered_imageB5+original);
% using fspecial
err4 = immse(blurred3,fil3+original);
err5 = immse(blurred4,fil4+original);
err6 = immse(blurred5,fil5+original);

%% 2.6
% upload image
pic = imread('images/lenna.jpg');
pic = im2double(pic);
%pic = rgb2gray(pic); is already gray

P = [0.1 0.5 0.9];

%plotting
figure
subplot(2,2,1)
imshow(pic)
title('original')
for i=1:3
 imf = FreqCanceling(pic,1-P(i));
 subplot(2,2,i+1)
 imshow(imf)
 title([num2str(P(i) * 100), '% filtered']);
end

%% 2.8
IM = double(imread('coins.png'))/255;
noisy_I = imnoise(IM, 'salt & pepper');
filtered_I_gaussian = CleanSP(noisy_I, 'Gaussian', 3, 3);
filtered_I_median = CleanSP(noisy_I, 'Median', 3, 3);

figure
subplot(2,2,1)
imshow(IM)
title('Original Image');
subplot(2,2,2)
imshow(noisy_I)
title('Noisy Image');
subplot(2,2,3)
imshow(filtered_I_gaussian)
title('Filtered Image (Gaussian)');
subplot(2,2,4)
imshow(filtered_I_median)
title('Filtered Image (Median)');

%% 3.2
% Loading circles picture
circles=imread("images\circles.png");
% Function activation
[n,coordinates]=LocateCirc(circles);

%% 3.3
% Loading the image
Iexam=imread('images\Iexam.tif');
% Normalizing the matrix
```

```
Iexam=mat2gray(Iexam);

% Showing the image
figure
imshow(Iexam)
title('Eye exam')

% Cell vector of the noisy images
IexamSP={imnoise(Iexam,'salt & pepper',0.01),imnoise(Iexam,'salt & pepper',0.03),imnoise(Iexam,'salt & pepper',0.2)};
% Allocating memory
IexamFiltered=IexamSP;
percentages=[1,3,20];
% Loop for filtering and showing the results
figure
for i=1:3
    % Filtering using opening and then closing of the noisy images
    IexamFiltered{i}=imclose(imopen(IexamSP{i},ones(3)),ones(3));

    % Plotting the noisy images
    subplot(2,3,i)
    imshow(IexamSP{i})
    title([num2str(percentages(i)) '%'])
    if i==1
        ylabel('Noisy picture')
    end
    % Plotting the filtered images
    subplot(2,3,i+3)
    imshow(IexamFiltered{i})
    if i==1
        ylabel('Filtered picture')
    end
end
sgtitle('Salt & Peper noise and filtered images')

%% 3.4
% Creating the stick-man
smIMG=zeros(20);
smIMG(10:19,4)=1;
smIMG([10 11],[3 5])=1;

% Plotting the stick-man
figure
subplot(131)
imshow(smIMG)
title('Original image')
% Moving the stick-man using dilation
SEd=[0 0 0 ;0 1 1;0 0 0];
for i=1:13
    smIMG=imdilate(smIMG,SEd);
end
% Plotting
subplot(132)
imshow(smIMG)
title('Dilated image')

% Moving the stick-man using erosion
SEe=[0 0 0 ;1 1 0;0 0 0];
for i=1:13
```

```
    smIMG=imerode(smIMG,SEe);
end
% Plotting
subplot(133)
imshow(smIMG)
title('Eroded image')

%% 3.5
% Creating track matrix
track=[1 25;2 10;1 -15;2 20;1 -11;1 35;2 -23;2 30;1 9];
% Function activation
M=IMove(track);

%% 4.2
% Loading the image
saturn=imread('saturn.jpg');
% Converting to gray-scale
saturn=rgb2gray(saturn);
% Normalizing the image
saturn=mat2gray(saturn);

% Creating the element for LoG filter
h=fspecial("log");

% Plotting the results
figure
subplot(121)
imshow(saturn)
title('Original image')

subplot(122)
imshow(imfilter(saturn,h))
title('Image after LoG')
```

פונקציית עזר:

```
function filteredIMG = FreqCanceling(IMG,P_energy)
% filter image according to energy threshold of low frequencies

IMGfft = fftshift(fft2(IMG(2:end,2:end)));
s = 0.02:0.1:10;
Energy = sum(sum(abs(IMGfft).^2)); %the total energy
for i=1:length(s)
    % high-pass filter h is created by subtracting a Gaussian low-pass filter from
an all-ones matrix.
    % The fspecial function generates the Gaussian filter with the current
standard deviation s(i).
    h = ones(size(IMGfft)) - fspecial('gaussian',size(IMGfft),s(i));
    filtered = IMGfft.*h; %apply the HPF
    energy = sum(sum(abs(filtered).^2));
    if energy >= P_energy*Energy
        filteredIMG = abs(ifft2(filtered));
        break
    end
end
end
```

ביבליוגרפיה:

[1] מ. אלעד, *עיבוד תמונות*. הפקולטה להנדסת חשמל, הטכניון, 2003.

[2] P. Soille, “Hit-or-miss and Skeletons,” in *Morphological Image Analysis: Principles and Applications*, P. Soille, Ed., Berlin, Heidelberg: Springer, 1999, pp. 129–154. doi: 10.1007/978-3-662-03939-7_5.

[3] “Feature Detectors - Sobel Edge Detector.” Accessed: Jan. 21, 2024. [Online]. Available: https://homepages.inf.ed.ac.uk/rbf/HIPR2/sobel.htm

[4] “Spatial Filters - Laplacian/Laplacian of Gaussian.” Accessed: Jan. 21, 2024. [Online]. Available: https://homepages.inf.ed.ac.uk/rbf/HIPR2/log.htm