**דו"ח מכין – מעבדה בענ"ת**

תרגיל הכנה – הצגת תמונה במטלב:

תמונה במטלב מיוצגת ע"י מטריצה אשר כל איבר בה מייצג פיקסל ומספק מידע לגביו. חלק מהתמונות מיוצגות ע"י מטריצה תלת-מימדית בה כל מימד מייצג אחד מצבעי RGB.

סוגי התמונות:

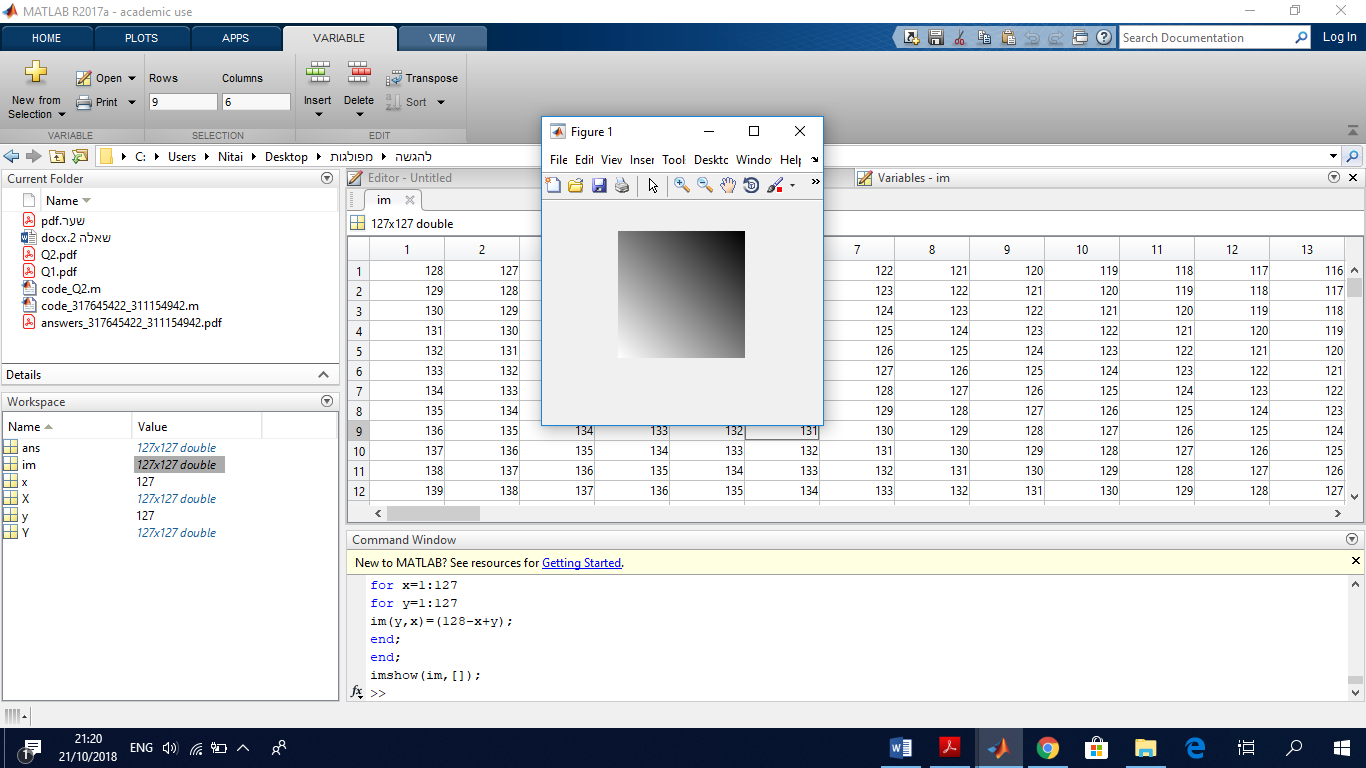
binary – כל פיקסל במערך הדו מימדי מיוצג ע"י 1 או 0 בתור לבן או שחור בהתאמה.

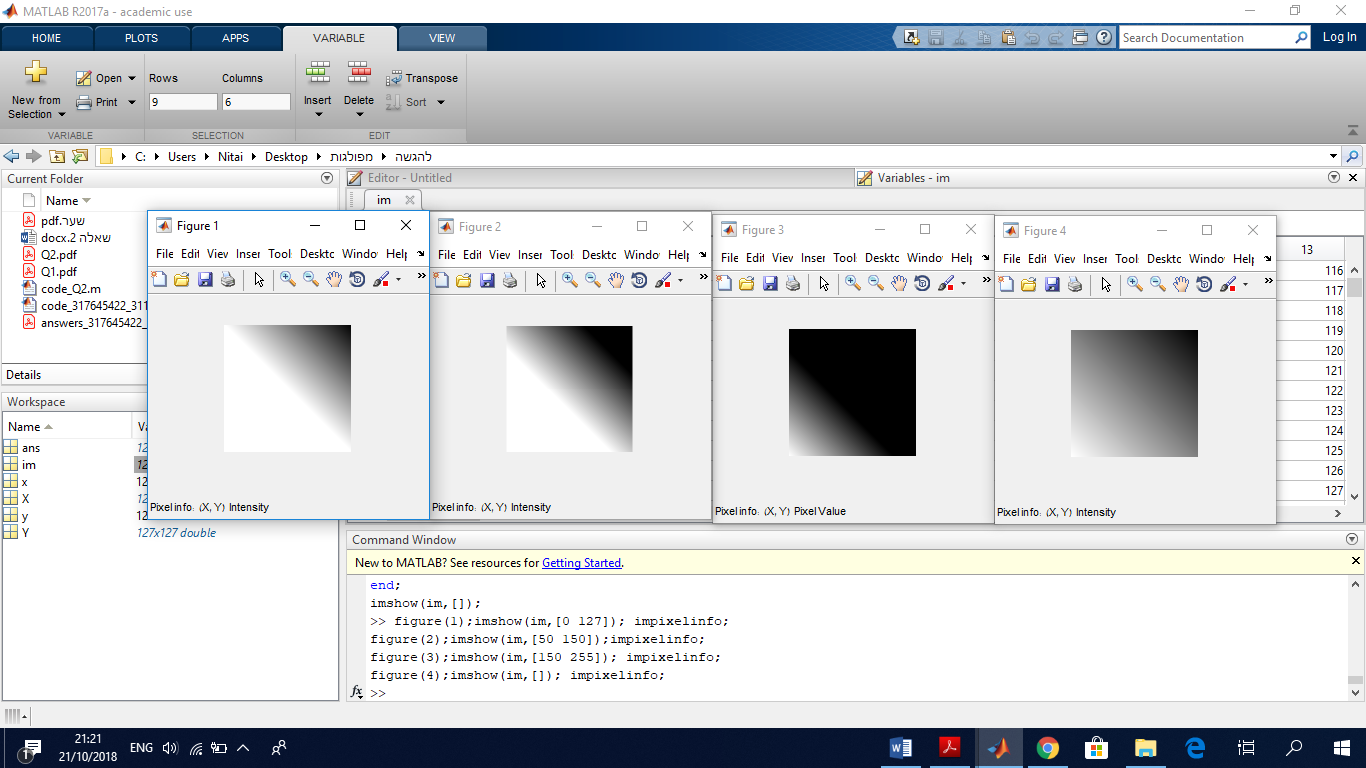
indexed - כל פיקסל במטריצה מיוצג ע"י אינדקס ממפת צבע.

grayscale – כל פיקסל במטריצה מיוצג ע"י ערך העוצמה שלו.

colored – זוהי מטריצה תלת מימדית שבה כל פיקסל מיוצג ע"י איבר המייצג את ערכי העוצמה של כל אחד מצבעי אדום, ירוק וכחול.

עבור התמונה ראשונה נקבל את הערכים הבאים (גם באמצעות לולאה וגם באמצעות meshgrid):



וארבעת התמונות הבאות:

כעת התמונות שונות בערכי הפיקסלים ע"י כך שקבענו את התחום הדינאמי שבהן בפרמטר השני של הפונקציה imshow, כך שהערך הנמוך בתחום מיוצג ע"י הצבע השחור והערך הגבוה מיוצג ע"י הלבן. כאשר הסוגריים ריקות, התחום הדינאמי הנקבע הוא הטווח שבין הפיקסל המינימלי שבתמונה למקסימלי שלה. כך ניתן לראות שבתמונה הראשונה משמאל, שהתחום הדינאמי שלה הוא 0 עד 127, כל הערכים מעל 127 הם לבנים, ולכן רוב התמונה לבנה, לעומת התמונה השלישית, שבה התחום הוא בין 150 ל-255, כלומר כל ערך שקטן מ-150 מוגדר כשחור, ולכן רוב התמונה מוגדרת כשחורה. בתמונה 4 שבה התחום הדינאמי הוא בין הערך המינימאלי של הפיקסלים בתמונה למקסימאלי ניתן לראות בדיוק את השתנות עוצמת הגוון האפור לאורך התמונה.

ניסוי 1 – אינפורמציה על תמונה

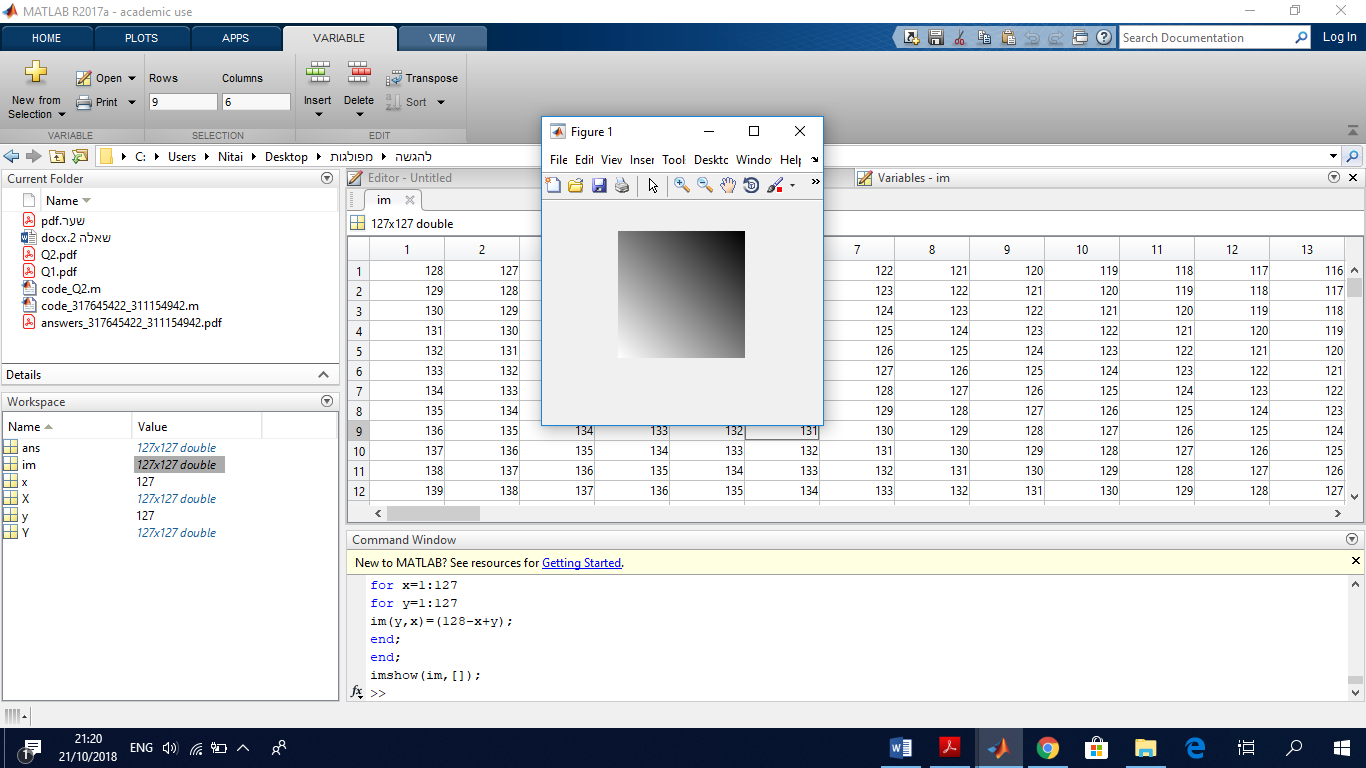
הפונקציה impixelinfo מאפשרת לשים את הסמן על התמונה, לקבל את הקואורדינטה הדו-מימדית שלה ואת ערך הפיקסל שלה.

הפונקציה histogram מספקת לנו את מספר הערכים של כל איבר במערך. כך, למשל, עבור תמונה, נוכל לקבל את היסטוגרמת הפיקסלים שלה, וכך ניתן לדעת את התפלגות הגוונים והצבעים השונים, לראות האם קיים פיקסל מסוים שבולט על פני אחרים וכד'.

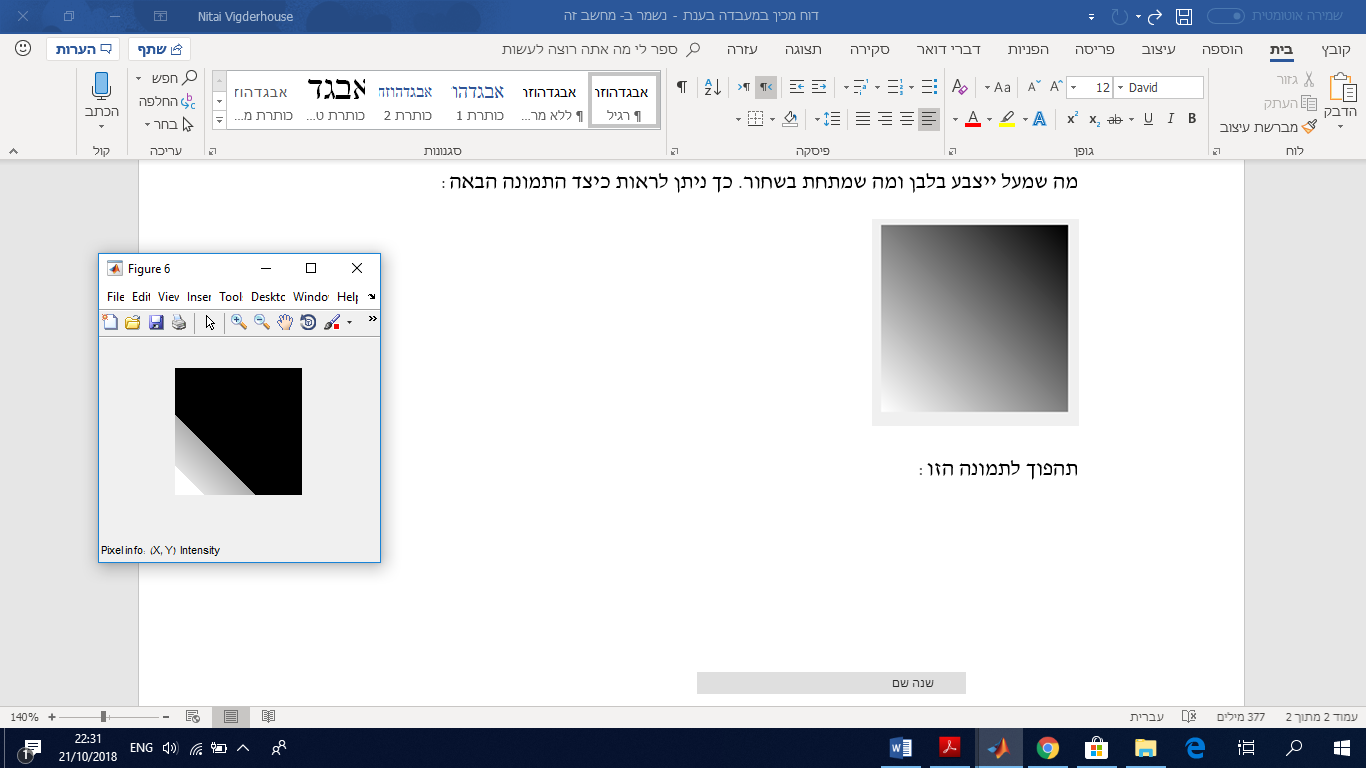
ניסוי 2 – מתיחת היסטוגרמה, שינוי בהירות וניגודיות

ניתן לבצע מתיחת היסטוגרמה על מנת להבין את התפלגות הגוונים בתמונה עבור תחום מסוים. למשל אם נקבע פיקסל מסוים וחלון מסוים שנמצא בסביבת הערך שלו, נוכל לראות כמה מהתמונה נמצא בתחום זה, כמה ממנו גדול מהתחום (אז ייצבע בלבן) וכמה קטן ממנו (אז ייצבע בשחור), וכך נוכל להבין איזה חלק בתמונה מכיל את התחום שאנו מתעניינים בו. נוכל לבודד את תחום זה ולבצע עליו שינויים בהתאם לרוחנו.

למשל, עבור פיקסל שערכו 200 וחלון בגודל 50 נוכל לראות איזה חלק בתמונה נמצא בתחום 175-225, מה שמעל ייצבע בלבן ומה שמתחת בשחור. כך ניתן לראות כיצד התמונה הבאה:

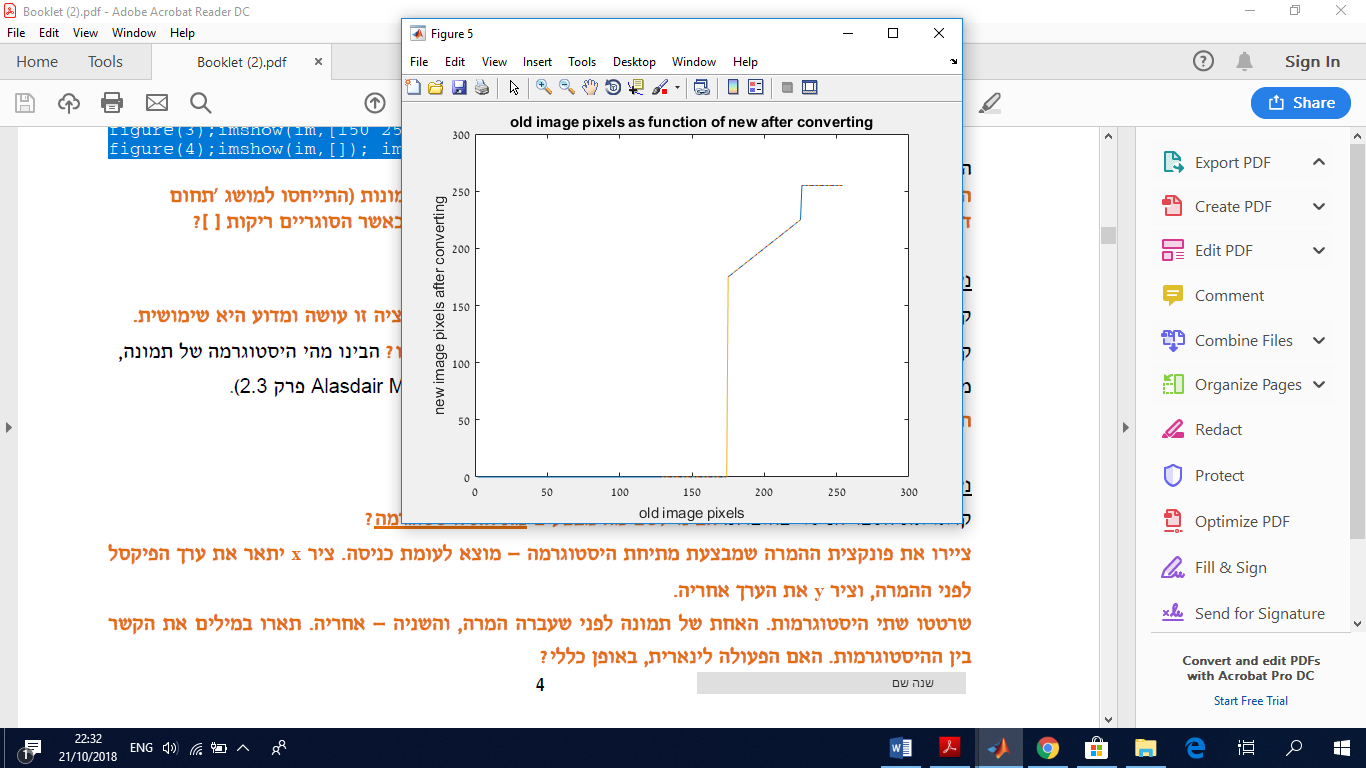


תהפוך לתמונה הזו:



וניתן לראות כי החלק שבתחום 175-225 מפולג באופן לינארי, זה שמעליו הפך לבן ושמתחתיו שחור.

פונקציית ההמרה שקיבלנו:



באופן כללי הפעולה אינה לינארית, כיוון שעבור התחום הרלוונטי נקבל את הפונקציה y=x, אך עבור x<level-windows/2 נקבל 0 ועבור x>level+windows/2 נקבל 255. באופן כללי ניתן לראות כי הפעולה אינה לינארית, כיוון שעבור ערכים מסוימים היא מקבעת אותם לערך קבוע מסוים ועבור ערכים אחרים (בחלון) היא לינארית.

ניסוי 3 – סגמנטציה של תמונה

תפקיד הפונקציה bwconncomp הוא למצוא רכיבים מקושרים בתמונה בינארית. כפרמטרים מובאת התמונה הבינארית וסוג הקישוריות המבוקשת שתקבע האם רכיב הוא מקושר או לא. במוצא נקבל את השדות של המבנה: סוג הקישוריות שנקבעה, גודל התמונה, מספר הרכיבים המקושרים והאינדקסים של כל אחד מהם.

קישוריות 4 קובעת כי אם הפיקסלים מקושרים באחד מ-4 הקצוות שלהם (האנכיים והמישוריים), כלומר סמוכים אחד לשני ובעלי אותו ערך בפיקסל, אז הם חלק מאותו רכיב מקושר. לעומת זאת, עבור קישוריות 8, אם הפיקסלים סמוכים אחד לשני באנך, במישור ובאלכסון, אז הם חלק מאותו רכיב קישורי.

עבור קישוריות 4 אלו הכיוונים הרלוונטים כדי ששני פיקסלים סמוכים יהיו באותו רכיב מקושר:



ועבור קישוריות 8:



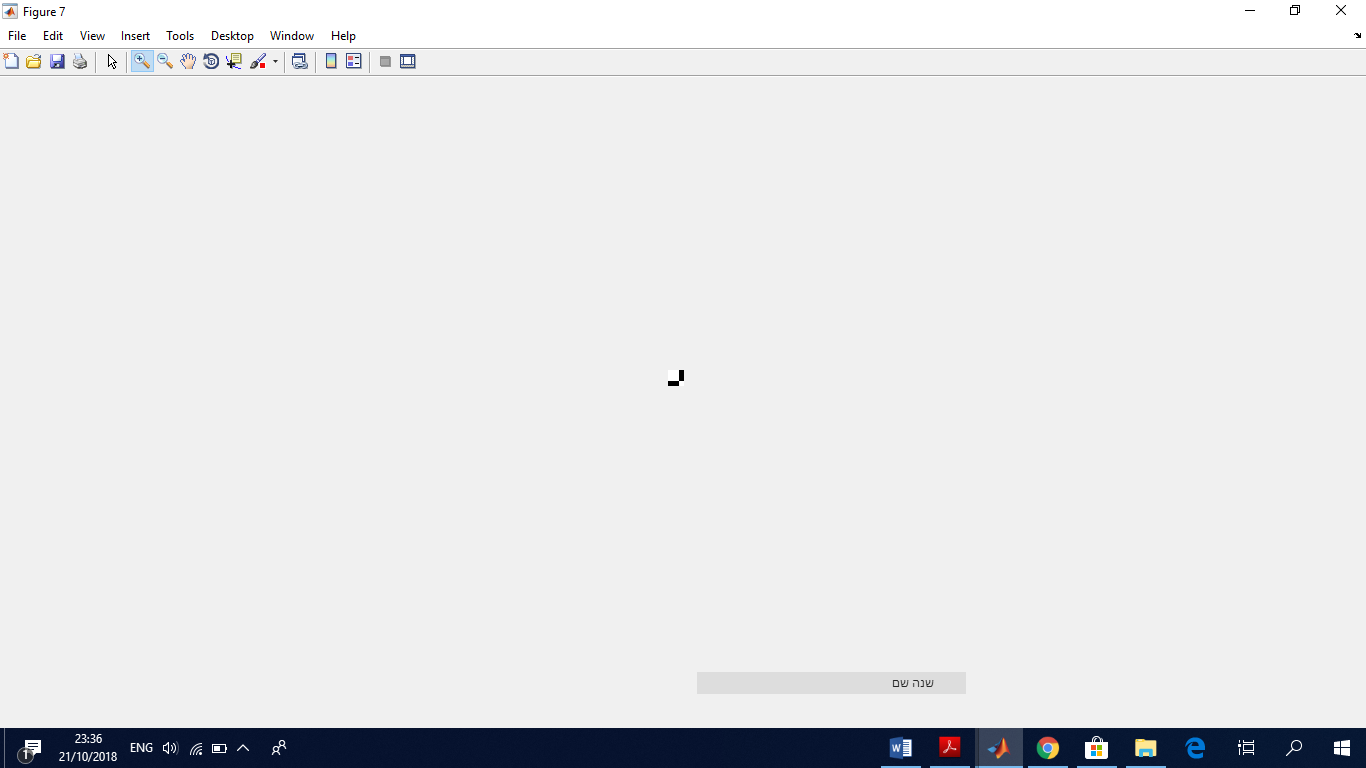
ניקח למשל את המטריצה הבאה:

1 1 0

1 1 0

0 1 1

שנראית כך:



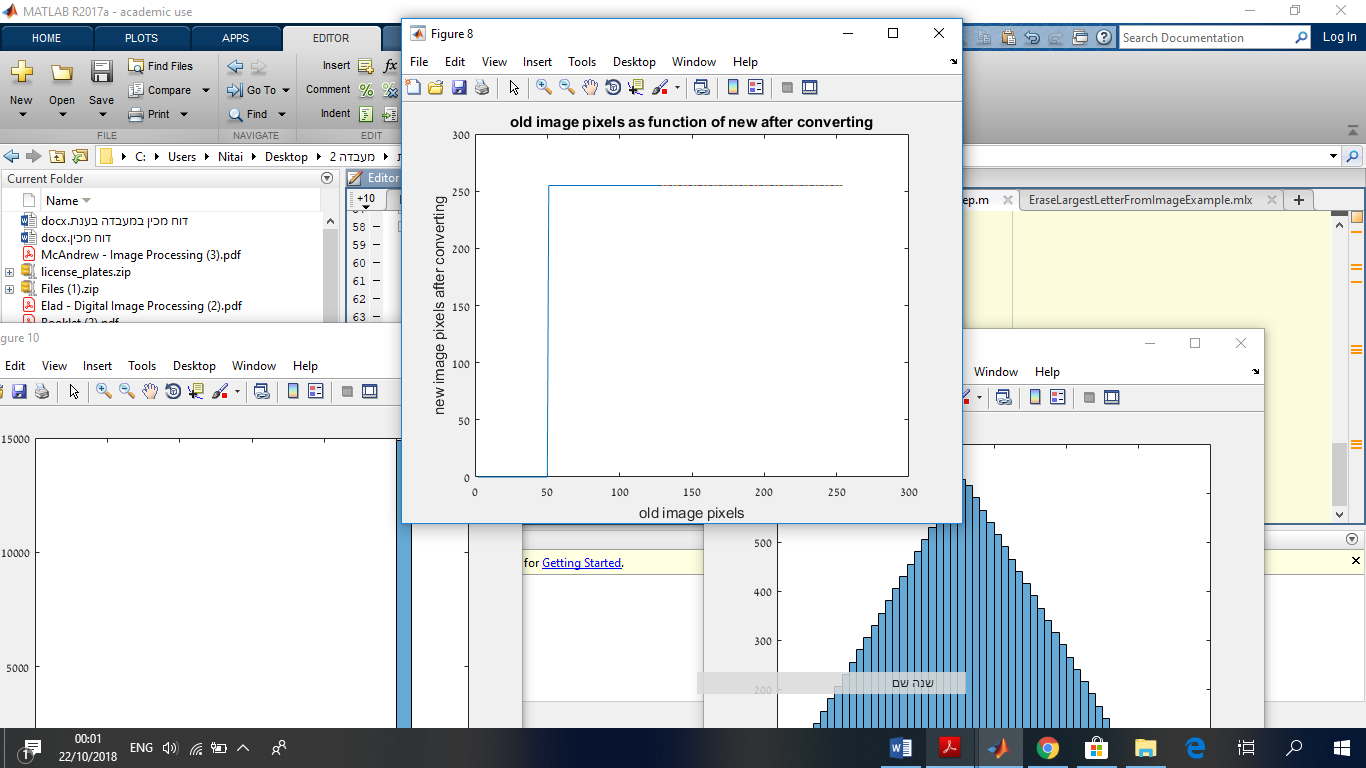
t

לאחר שנריץ עליה את הפונקציה בקישוריות 4, נקבל כי קיימים 2 רכיבים מקושרים ואילו עבור קישוריות 8 נקבל רכיב מקושר אחד, כיוון שהוא מחשיב שני פיקסלים באותו איבר גם אם הם באלכסון (בניגוד לקישוריות 4).

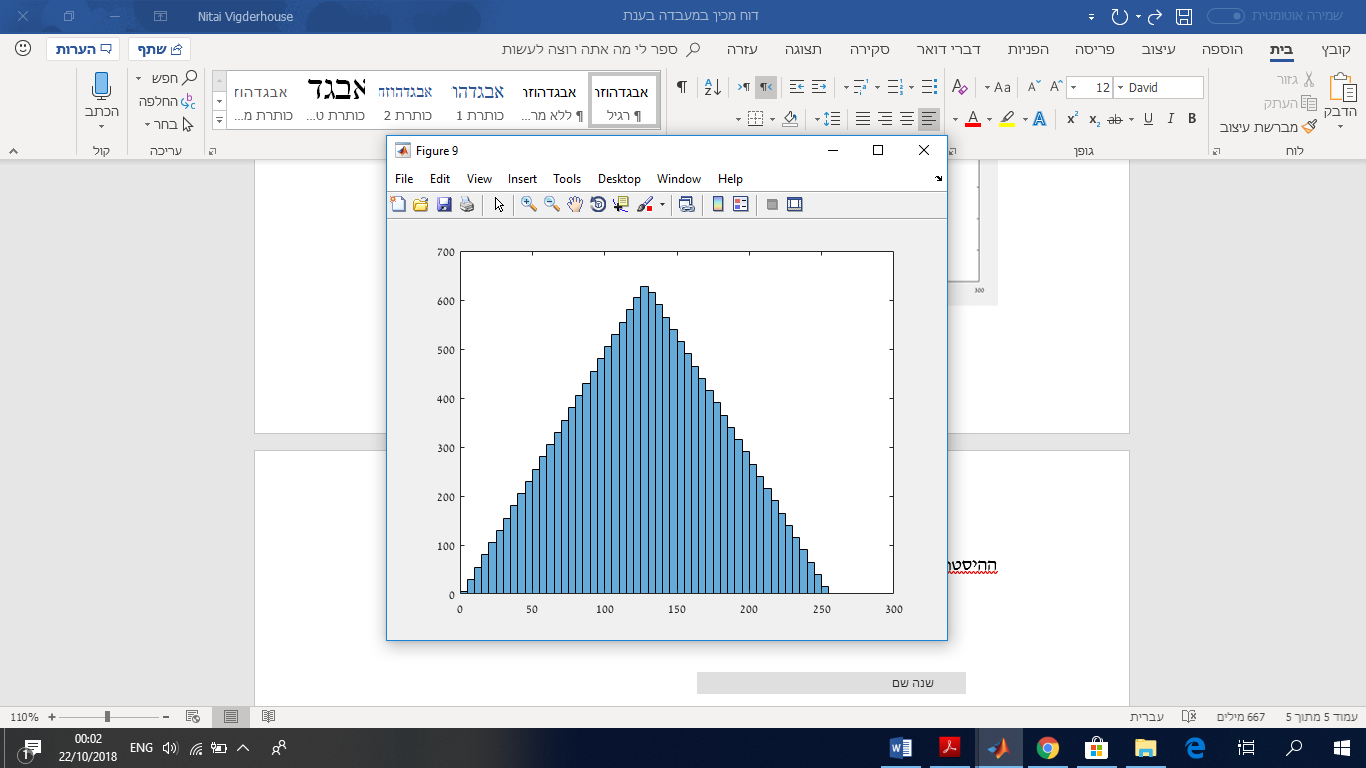
הפונקציה labelmatrix מאפשרת לנו לקבל מטריצה עבורה פיקסלים עם ערך 0 הם הרקע, ועבור כל ערך השונה מ-0, קבוצת הפיקסלים שערכה ערך מסויים היא חלק מאותו רכיב מקושר. דבר זה מאפשר לנו לעשות סגמנטציה כי הוא מבודד ומייחד אובייקטים בתמונה ומאפשר לנו לבצע עליהם סגמנטציה.

עבור תמונה העוברת חיתוך סף נקבל את הגרפים הבאים:

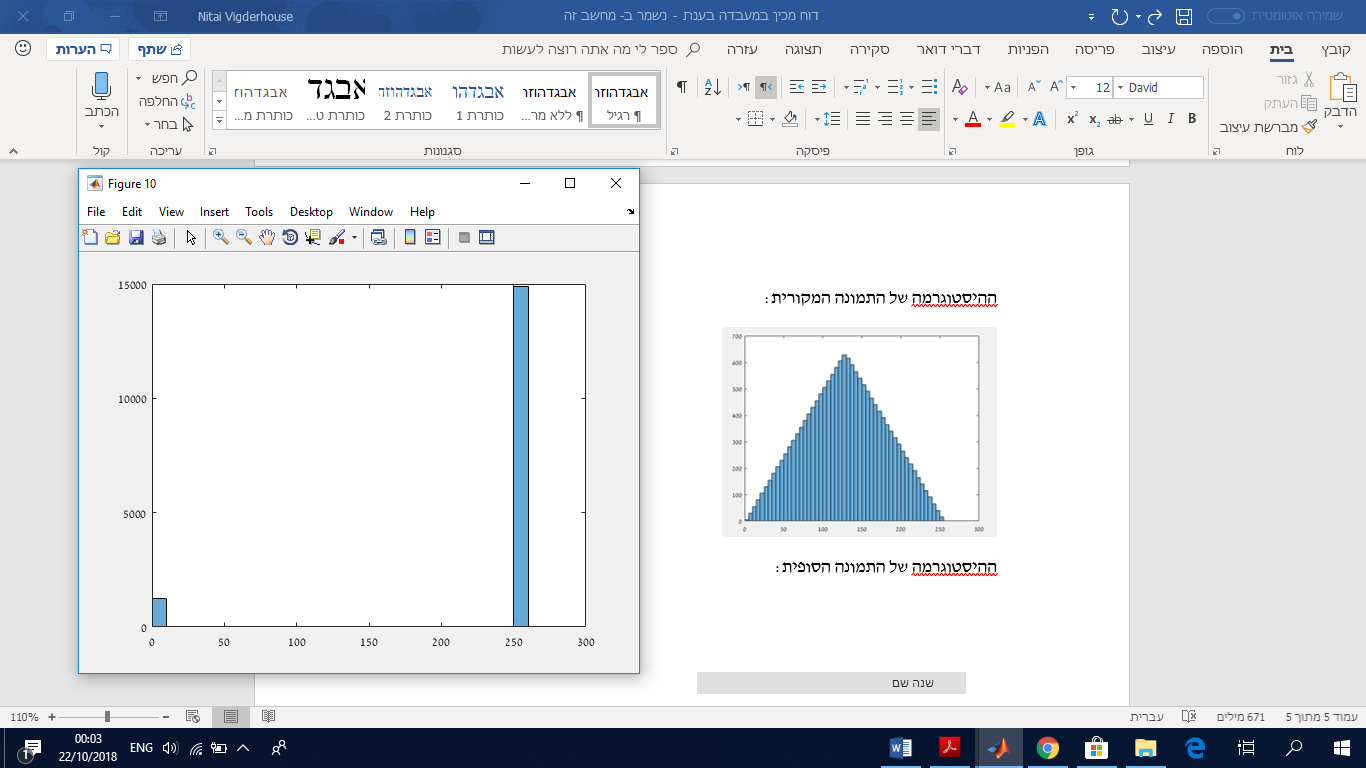
ההארכה ההתחלתית (ציר x) לעומת הסופית (ציר y):



ההיסטוגרמה של התמונה המקורית:



ההיסטוגרמה של התמונה הסופית:



לתמונות רמות אפור המכילות מספר אובייקטים המסתירים זה את זה באופן חלקי יש לבצע סגמנטציה באמצעות קביעת מספר ספים בתמונה על פי חלוקה בתחומים, כך שעבור אובייקטים המסתירים אובייקטיבים אחרים נבצע סגמנטציה על ידי קביעת ערך מסוים עבור תחום ערכי הפיקסלים בהם הם נמצאים וערך אחר עבור האובייקט המוסתר כך שהתחום שלו לא יחפוף עם התחום של האובייקט המסתיר.

נדרש לעשות סגמנטציה בתהליכים שברצוננו לבודד אובייקטים מסוימים מהרקע ולסווגם. למשל, אם נרצה לסווג סוג חיה על רקע נוף, נרצה לבצע סגמנטציה על החיה ולבודד אותה מהנוף, ורק אז לבודד אותה. כמו כן, סגמנטציה מאפשרת לנו באמצעות בידוד האובייקט בתמונה מסוימת לבצע רק עליו שינויים – למשל במקרה שיש לנו מספר חיות על רקע לבן, נוכל לבצע עליהן סגמנטציה ולהחליט שעבור החיה המסווגת כחתול נוכל "למחוק" אותה ע"י הפיכת כל הפיקסלים שלה ל-0.