# עיבוד תמונה – ת"ב מספר 1

## מגישים:

* ואדים חכם, ת"ז 311890156
* גלעד עיני, ת"ז 034744920

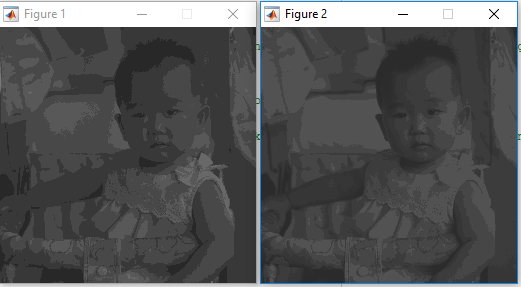
## תלויות של הפרויקט:

* שתי תמונות מה IMAGES של חגית שמצורפות להגשה על כל מקרה:
  + darkimage.tif
  + lakeScene.tif
* פונקציות מהTOOLBOX של חגית (לא מצורף):
  + readImage.m
  + showImage.m

## תשובות ותמונות לסקריפט 7

### 7א. השוואה בין כימות יוניפורמי לבין כימות אופטימלי (N=4)

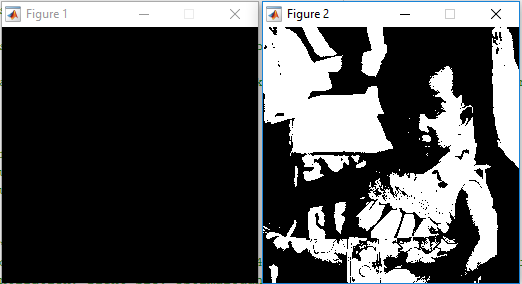
* + תמונה מקורית מה IMAGES: darkimage.tif



* + ניתן לראות משמאל את הכימות היוניפורמי.
  + מימין את הכימות האופטימלי
  + האופטימלי יותר טוב.
    - ביוניפורמי, היד הימנית נראית כחלק מהרקע. כמו כן, קשה להבדיל מתי המצח נגמר ומתי מתחיל השיער. כמו כן, רואים false contour בכל הרקע, במיוחד בפינה שמאלית עליונה.
    - באופטימלי, רואים את המצח, השיער והיד הרבה יותר טוב.
  + בחרנו בתמונה זו כי ידענו שהתמונה עברה השחרה(ע"י חגית), ושהיוניפורמי "יבזבז" צבעים(QI) סתם כי הוא יוניפורמי, להבדיל מהאופטימל שרוב הצבעים קרובים לתמונה.

### 7ב.השוואה בין BINARIZE עם T=128 וBINARIZEOPT (כלומר T אופטימלי)

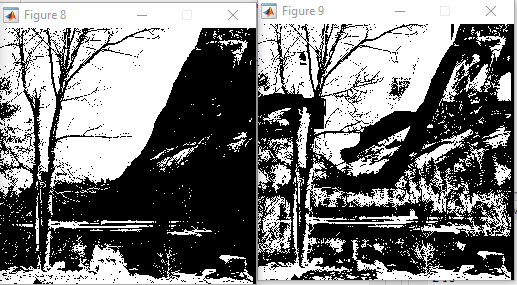
* + תמונה מקורית מה IMAGES: darkimage.tif



* + ניתן לראות משמאל את binarize('darkimage.tif',128).
  + מימין את binarizeOpt('darkimage.tif')
  + האופטימלי יותר טוב.
  + לא יכלה להיות תמונה יותר טובה להמחיש את ההבדל.
  + אנו מניחים ,בלי להסתכל בערכי הצבעים של התמונה, שרובה אם לא כולה, ערכים מתחת ל128. לכן, שקראנו עם T=128, יצאה תמונה שחורה.
  + זהו יתרון ענק של הT אופטימלי, הוא יכול להתמודד עם תמונות בהירות ותמונות שחורות.

### 7ג. השוואה בין binarizeOpt() ל binarizeOptAdaptive()

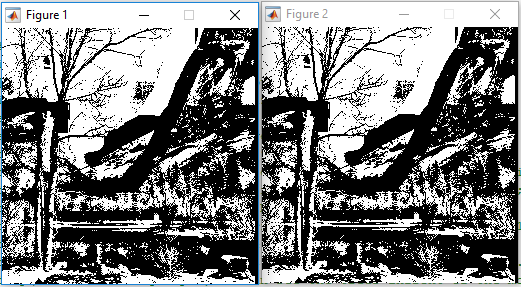
* + תמונה מקורית מה IMAGES: lakeScene.tif



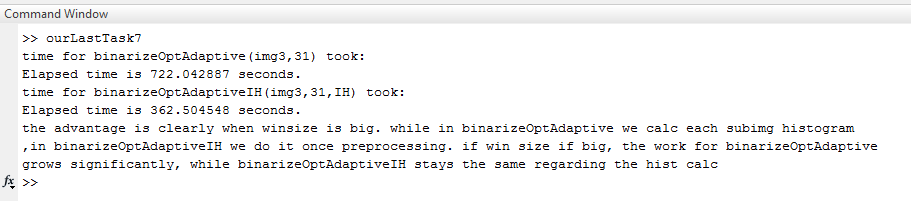
* + משמאל את binarizeOpt('lakeScene.tif ')
  + מימין את binarizeOptAdaptive('lakeScene.tif ',31)
  + בחרנו בתמונה עם שינוי מהיר בפרטים כדי לקבל את המקסימום מהפונקציה.
  + התמונה האדפטיבית יותר טובה מפני שקיבלנו הרבה יותר פרטים. ניתן לשים לב לכל הצמחיה מצד ימין למטה, שנמצאת לפני ההר. ניתן לשים לב גם לצמחיה הפזורה על המים. דבר נוסף, ניתן לשים לב למשהו משמעותי ביותר, רואים את ההרים מאחורי העץ הגדול מצד שמאל. בתמונה השמאלית,נבחר T אחד לכל התמונה. פשוט נעלמו ההרים. הפונקציה לא ידעה להתמודד עם 3 גוונים ממש שונים: צבע ההרים, השמיים והצמחיה. בפונקציה האדפטיבית, זה נעשה פר פיקסל שמוקף בתת תמונה בגודל WINSIZE. לכן היה יותר קל שהגיעו חתיכות של תמונות.

### 7ד. השוואה בין binarizeOptAdaptive () ל binarizeOptAdaptiveIH()

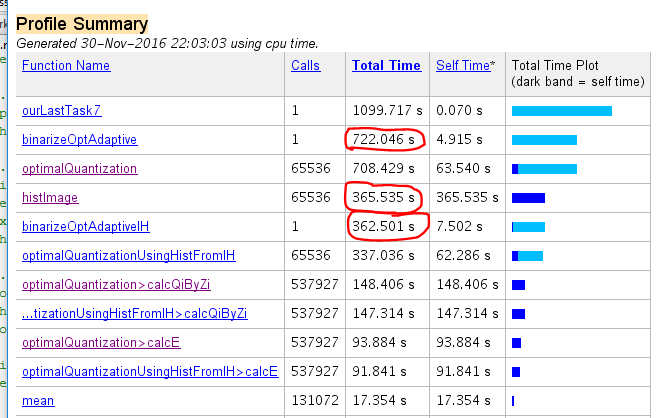
* + תמונה מקורית מה IMAGES: lakeScene.tif



* + כצפוי אותה תוצאה. אבל מה מבחינת יעילות?
  + אז מסתבר שחישוב ההיסטוגרמה בכל איטרציה לקח המון זמן מבחינת סך כל ריצת התוכנית. לשם מדידה השתמשנו ב TIC TOC וגם בפרופיילר



* + זו התוצאה של הרצת הסקריפט (7ד) ורואים שגם בחלון קטן בגודל 31 הזמן משתפר.
    - בלי IH: 722 שניות
    - עם IH: 362 שניות
    - ההפרש: 360 שניות.
  + כל ההבדל בקוד הוא קריאה לפונקציה optimalQuantization עם ובלי ההיסטוגרמה של התמונה שמועברת.



* + תמונה של זמני הפונקציות.
  + ניתן לראות שבלי IH לקח 722 שניות שמתוכן 365 שניות בוזבזו על histImage.
  + עם IH לקח 362 שניות. אם נסכום את הזמן שלקח ל histImageנקבל 727 שניות, כלומר כמעט את הזמן שלקח בלי IH(החמש שניות זה החישוב של ההיסטוגרמה מתוך IH + זמן העברת ההיסטוגרמה לכימות האופטימלי).