INTRODUCTION TO DIGITAL IMAGE PROCESSING

361.1.4751

EXERCISE 2 - COLOR SPACES

Submission Date: 30.11.2022

Noam Atias – 311394357

Chanel Michaeli - 208491787

תוכן עניינים

3..... RGB and Grayscale.1

7.....Additive vs Subtractive Color space .2

10.....HSV .3

*16.....L*a*b .4*

20.....Color Segmentation .5

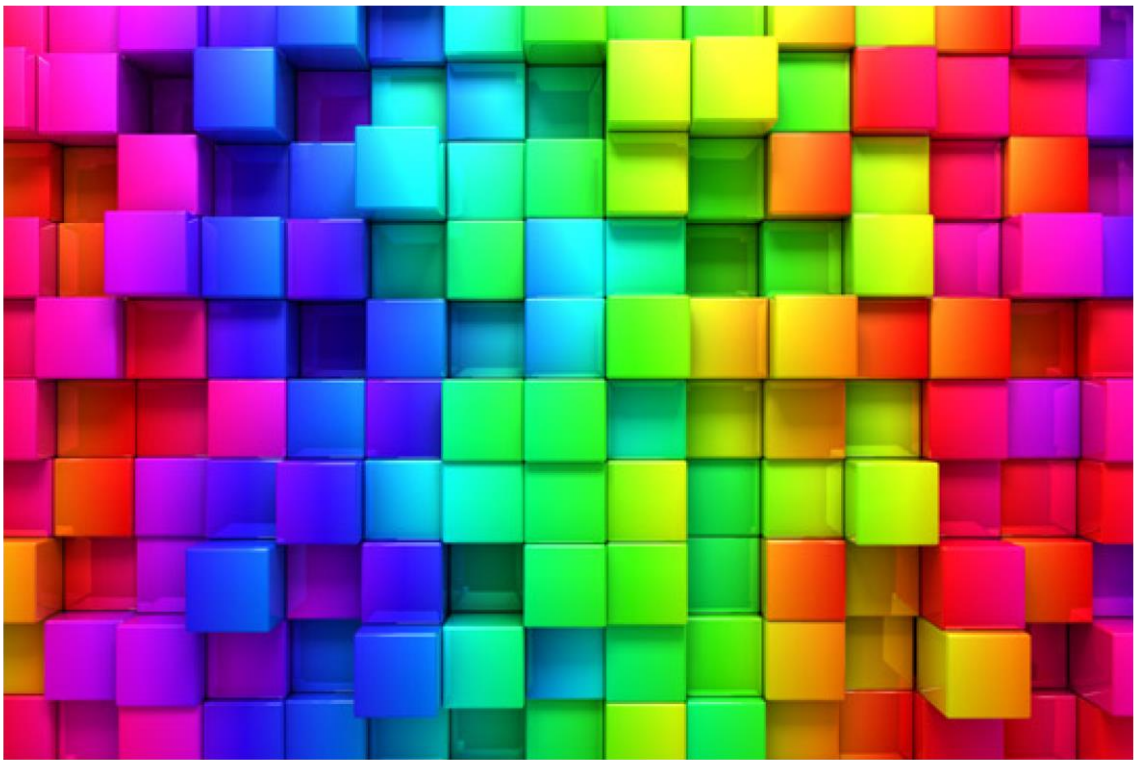
1. RGB and Grayscale

1. ראשית נקרא את התמונה: *'colours.jpeg'* , ולאחר נמיר את התמונה ל – *type double*.

   לאחר מכן נרמלנו את התמונה כך שערכי התמונה יהיו: [0 , 1].

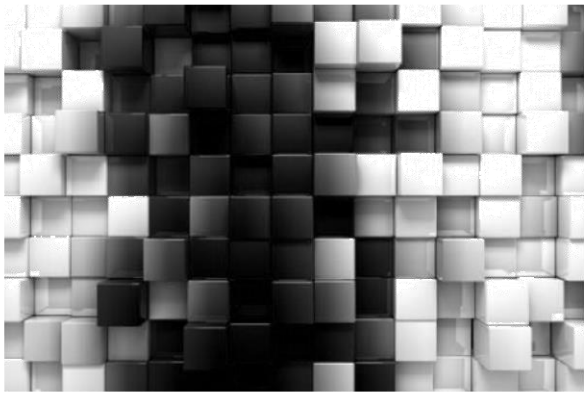
2. נציג את התמונה שקיבלנו על ידי הפונקציה *imshow()*. התמונה שקיבלנו:

Normalized colors image

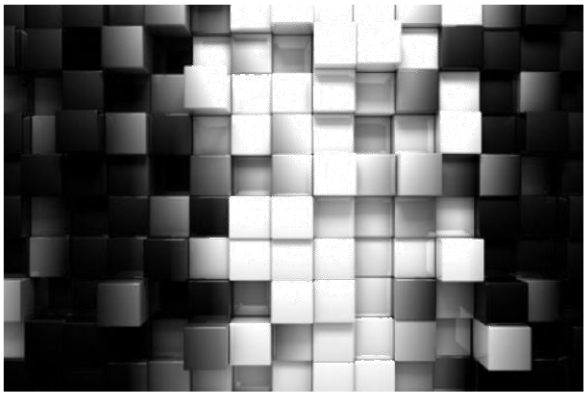


3. נציג כל ערוץ בנפרד בצורה הבאה:

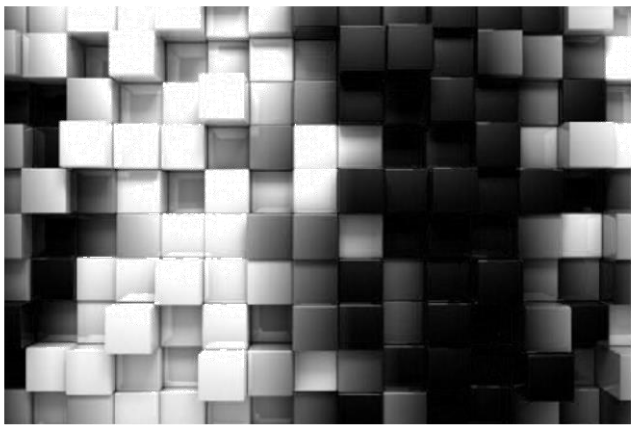
Red Channel



Green Channel



Blue Channel



4. נבנה פונקציה משלנו בשם: *'dip _rgb2gray()'* .

הפונקציה מקבלת תמונה עם שלושה ערוצים (RGB) וממירה אותה לתמונת Grayscale באופן הבא:

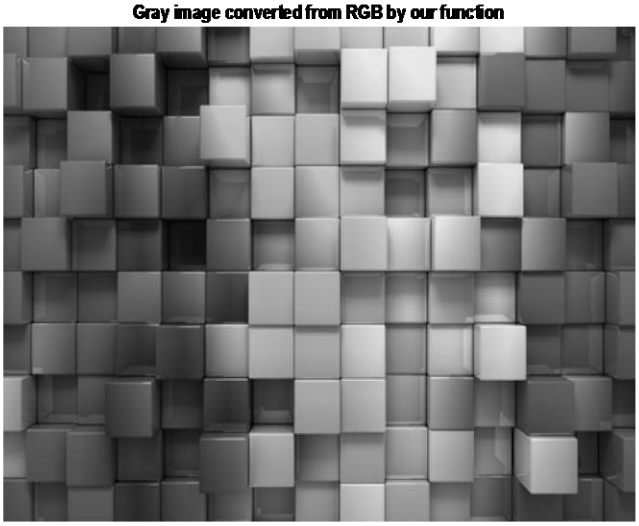
$$Gray = 0.2989 \cdot R + 0.5870 \cdot G + 0.1140 \cdot B$$

<u>כאשר:</u>

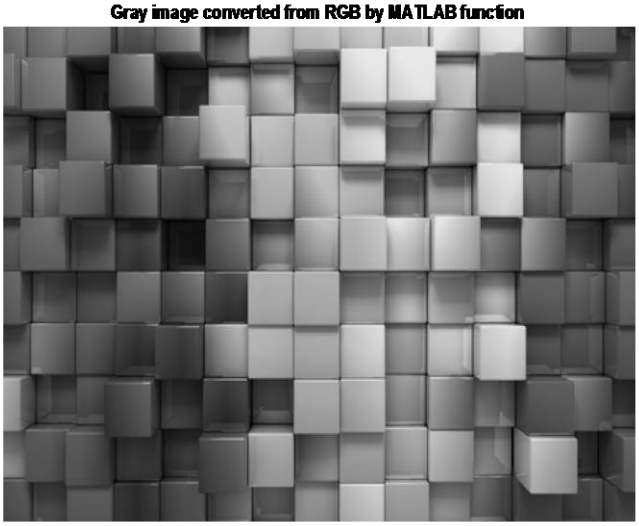
- R – הוא הערוץ הראשון.
- G – הוא הערוץ השני.
- B – הוא הערוץ השלישי.

5. נשווה בין התמונה שקיבלנו לבין התמונה המתקבלת על ידי שימוש בפונקציה המובנית של MATLAB : *'rgb2gray()'*. נציג את שתי התמונות שמתקבלות ונראה מדידות המעידות על הדמיון בין התמונות:

Gray image converted from RGB by our function



Gray image converted from RGB by MATLAB function



ניתן לראות כי בהבחנה ראשונית התמונות נראות זהות. כדי לבחון את רמת הדמיון בין התמונות השתמשנו במדד MSE (שגיאה ריבועית ממוצעת), וקיבלנו שגיאה שקרובה מאוד ל – 0.

בנוסף השתמשנו במדד שנקרא Structural similarity (SSIM) index, המודד את מידת הדמיון

בין שני תמונות Grayscale, שבשונה מ – MSE, מתחשב בכך שלפיקסלים יש תלות הדדית חזקה במיוחד כשהם קרובים מבחינה מרחבית. ככל שהאינדקס המתקבל קרוב יותר ל-1 כך הדמיון בין התמונות חזק יותר. במקרה שלנו קיבלנו כי האינדקס שווה ל-1 בדיוק:

```
Noam Atias 311394357, Chanel Michaeli 208491787

D_H_trace =

   6.4996e-13


D_S_trace =

     0


D_V_trace =

     0
```

6. נפעיל מניפולציות על תמונת ה – RGB המנורמלת בכל ערוץ בנפרד באופן הבא:

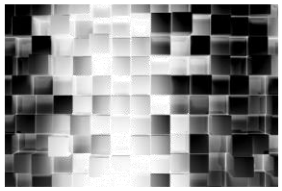
- על הערוץ הראשון (I_1) נפעיל פונקציה לינארית: $F(I_1) = -I_1 + 1$.

מניפולציה זו תגרום לכך שהצבעים יתהפכו, כלומר ערכים נמוכים שהיו קרובים ל – 0 יהפכו להיות גבוהים מאוד, כלומר קרובים ל – 1, ולהיפך. כך נקבל שצבעים בהירים יהפכו להיות צבעים כהים והפוך.

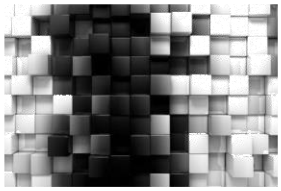
- נחליף בין הערוץ השני לבין הערוץ השלישי. כעת, הערוץ השני יהיה בעל תכונות הצבעים של ערוץ Blue .
- על הערוץ השלישי נפעיל פילטר גאוסיאן (השתמשנו במימוש של הפילטר שפיתחנו בעבודה 1), כתוצאה מכך נסיר את כל התדרים הגבוהים של הערוץ, כלומר נגרום לשפות של הערוץ להיות פחות בולטות וכך ניצור מריחה או טשטוש של השפות בתמונה.

נציג את הערוצים לפני ואחרי הפעלת המניפולציות :

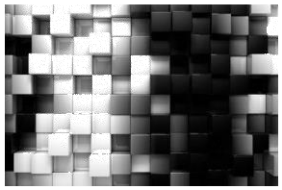
Manipulated Red Channel



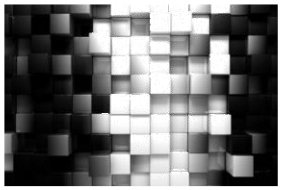
Red Channel



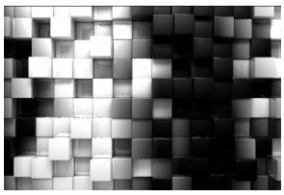
Manipulated Green Channel



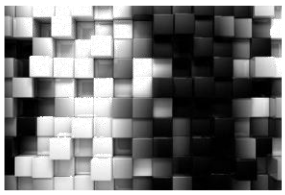
Green Channel



Manipulated Blue Channel

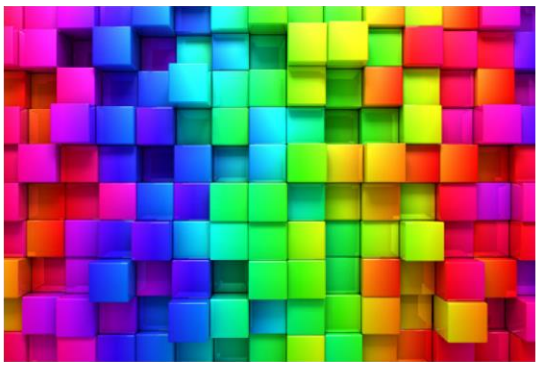


Blue Channel

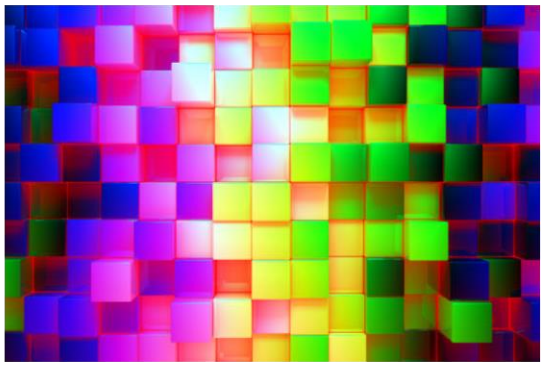


נציג את הערוצים לפני ואחרי הפעלת המניפולציות בצבעים:

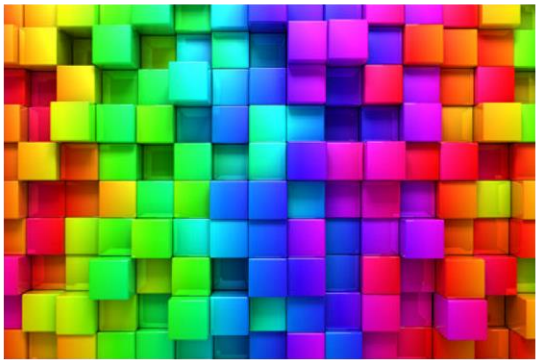
Original image



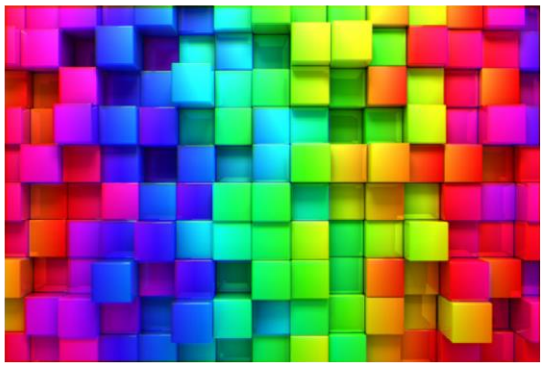
Manipulated Red Channel



Manipulated Green Channel



Manipulated Blue Channel



ניתן הסבר לכל מניפולציה:

- נבחין כי עבור הערוץ Red, קיבלנו היפוך בין הערכים הגבוהים לערכים הנמוכים ובערוץ הבודד נראה היפוך בין האזורים הכהים לבהירים. בתמונה הכוללת את שלושת הערוצים ,נראה כי אזורים שהיו בצבע אדום בתמונה המקורית הפכו לצבע כחול וזאת מאחר שבאזורים אלה לאחר המניפולציה רק הערוץ הכחול מכיל ערכים גבוהים. בנוסף, נראה כי הצבע הירוק הפך לצבע צהוב ולהפך מכיוון שצבע צהוב נוצר משילוב של אדום וירוק ולכן נקבל שבהעדר הצבע האדום, הצבע הצהוב ייהפך לירוק. הבחנה יותר מעמיקה היא שהשפות של הערוץ שלפני כן היו כהות, הפכו להיות בהירות, כלומר בתמונה הכוללת יהיו אדומות וזאת מאחר שלפני המניפולציה השפות בערוץ האדום היו בעלות ערך נמוך ולכן לאחר המניפולציה קיבלו ערך גבוה ולכן מוצגות בצבע אדום.
- עבור ערוץ Green קיבלנו את הצבעים של הערוץ Blue, כלומר קיבלנו שהצבע הירוק הפך לצבע כחול ולהפך, בעוד הצבע האדום נותר ללא שינוי.
- עבור ערוץ Blue, נקבל כי השפות של הצבע הכחול נעלמו עד כדי טשטוש בעקבות השימוש במסנן גאוסיאן.

2. Additive vs Subtractive Color space

1. מרחב הצבעים CMYK הוא מרחב צבעים חיסורי המשמש להדפסות, ומתייחס לארבעת צבעי הדיו: cyan, magenta, yellow and key (black) .
המרחב מצריך את קיום הצבע השחור, מאחר לא ניתן לייצר את הצבע השחור מבלי להגדיר אותו.
מרחב זה עובד על ידי מיסוך חלקי או מלא של צבעים על רקע בהיר יותר, בדרך כלל לבן.
מרחב זה נקרא חיסורי מאחר שצבע הדיו "מפחית" את הצבעים אדום, כחול וירוק מהרקע הלבן.
אם מחסירים מהרקע הלבן את הצבע האדום נשארים עם הצבע ציאן (cyan) .
אם מחסירים מהרקע הלבן את הצבע הירוק נשארים עם הצבע מג'נטה (magenta).
אם מחסירים מהרקע הלבן את הצבע הכחול נשארים עם הצבע צהוב.
בשונה ממרחבים חיבוריים, כמו מרחב RGB, שבו הצבע הלבן נוצר על ידי חיבור שלושת הצבעים אדום כחול וירוק והצבע השחור הוא המחסור בצבע, במרחב CMYK הצבע הלבן הוא הרקע והצבע השחור נוצר משילוב שלושת הצבעים ציאן, מג'נטה וצהוב, אך כדי להסוך בעלויות דיו וכדי לייצר צבע שחור עמוק יותר משתמשים בצבע דיו שחור.

2. נייצר את ערוצי המרחב CMYK מתוך ערוצי ה- RGB באופן הבא:

$$Black = minimum(1 - Red, 1 - Green, 1 - Blue)$$

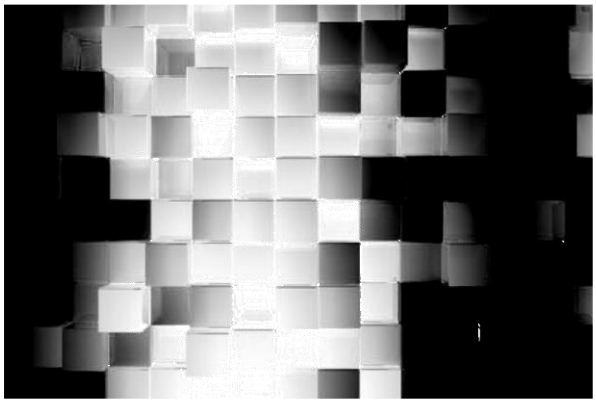
$$Cyan = \frac{1 - Red - Black}{1 - Black}$$

$$Magenta = \frac{1 - Green - Black}{1 - Black}$$

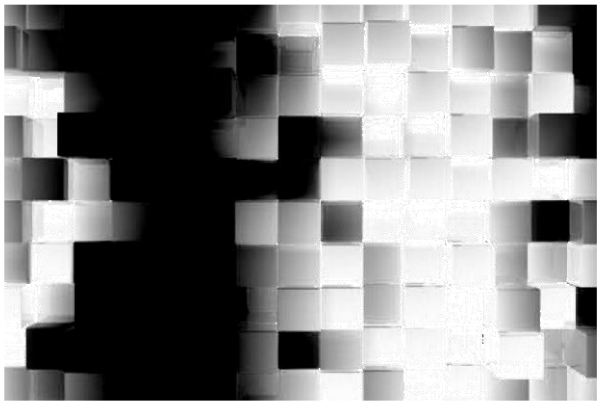
$$Yellow = \frac{1 - Blue - Black}{1 - Black}$$

נציג כל ערוץ בנפרד:

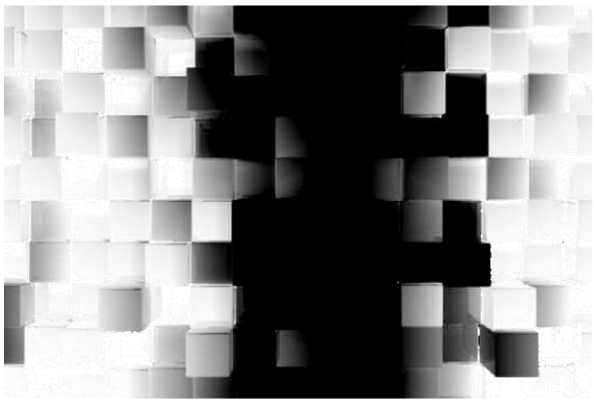
Cayn channel



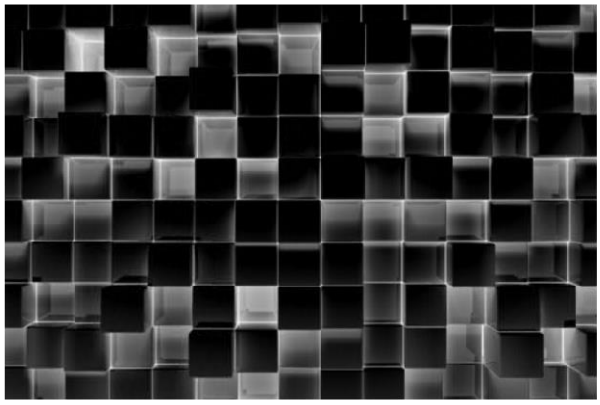
Yellow channel



Magenta channel

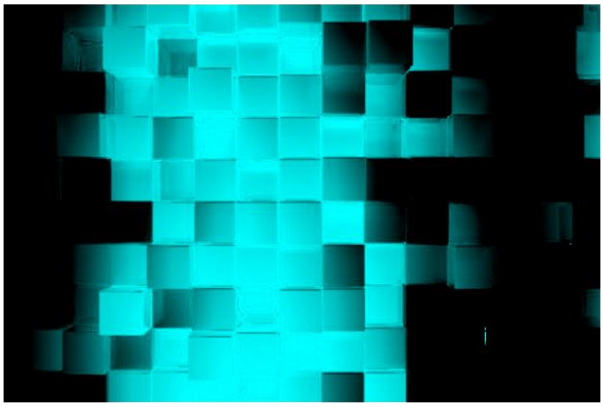


Black channel

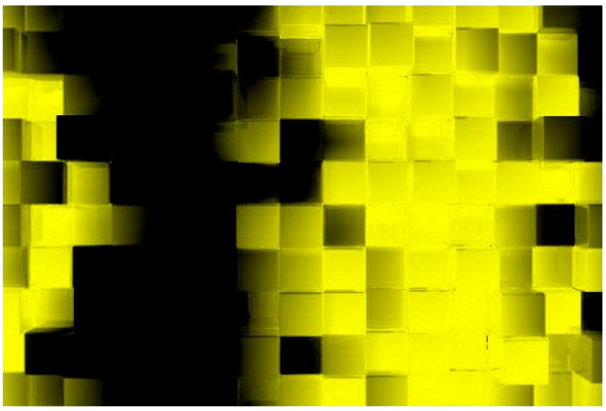


3. נציג את הערוצים הנפרדים על ידי שימוש בפונקציה: *'displayCYMK()'*.

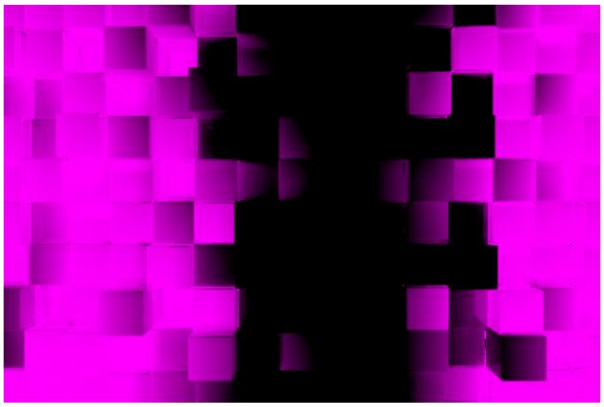
Cyan



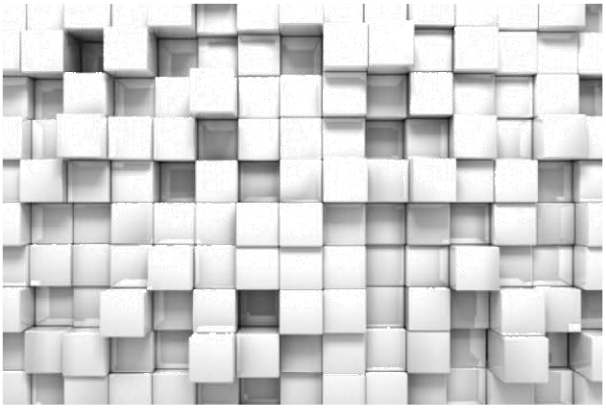
Yellow



Magenta



Black



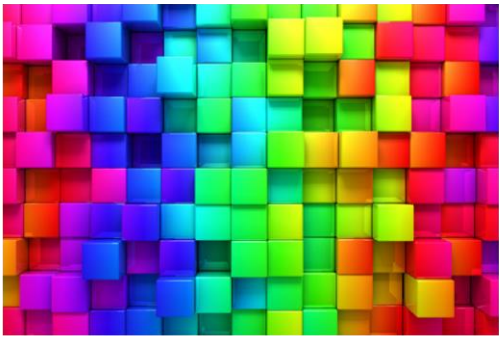
4. נחזור על סעיף 6 משאלה 1:

נפעיל מניפולציות על תמונת ה – CYMK המנורמלת בכל ערוץ בנפרד באופן הבא:

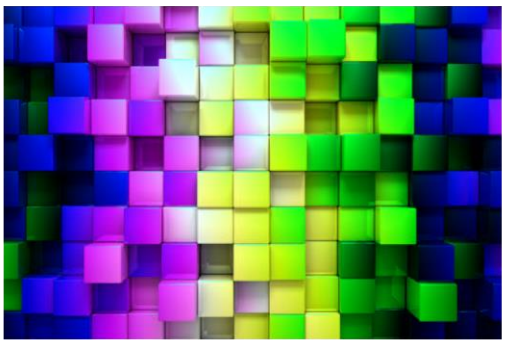
- על הערוץ הראשון (I_1) נפעיל פונקציה לינארית: $F(I_1) = -I_1 + 1$.
- נחליף בין הערוץ הראשון לבין הערוץ השני. כעת, הערוץ השני יהיה בעל תכונות הצבעים של ערוץ Cyan, והערוץ הראשון יהיה בעל תכונות של הערוץ Yellow .
- על הערוץ הרביעי נפעיל פילטר גאוסיאן (השתמשנו במימוש של הפילטר שפיתחנו בעבודה 1), כתוצאה מכך נסיר את כל התדרים הגבוהים של הערוץ, כלומר נגרום לשפות של הערוץ להיות פחות בולטות וכך ניצור מריחה או טשטוש של השפות בתמונה.

נציג את הערוצים לפני ואחרי הפעלת המניפולציות בצבעים:

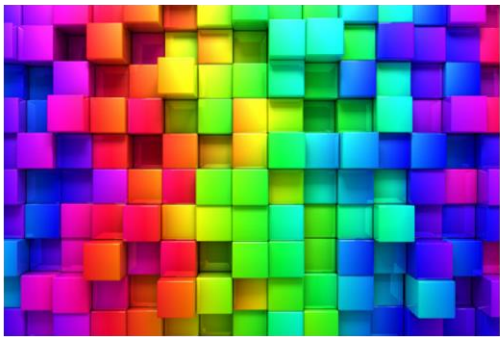
Original image - CYMK



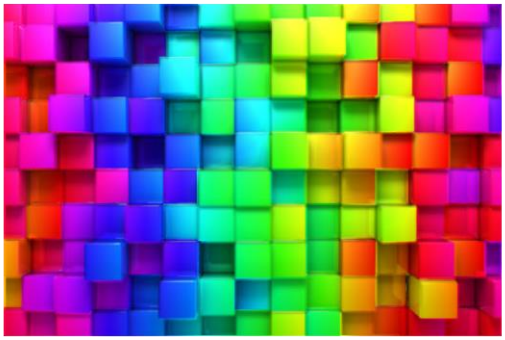
Linear manipulated Cyan Channel



Manipulated swap Channels



Gaussian manipulated black Channel



ניתן הסבר לכל מניפולציה:

- במניפולציה הראשונה ביצענו היפוך של הצבע Cyan, כלומר באזורים בהם היה הצבע Cyan כעת לא יהיה כלל ולהיפך. מכיוון שבמרחב צבעים זה הצבע הירוק נוצר על ידי שילוב של Cyan יחד עם Yellow כעת לאחר מניפולציה זו נקבל שבאזורים בהם היה צבע ירוק כעת יהיה צבע צהוב ולהיפך. בנוסף נוכל לראות שמכיוון שמרחב צבעים זה הצבע הכחול נוצר על ידי Cyan יחד עם Magenta כעת לאחר המניפולציה נקבל שבאזורים בהם היה צבע כחול כעת

יהיה צבע לכיוון מג'נטה וההיפך. באופן דומה נוכל להסיק זאת לגבי יתר שילובי הצבעים שבהיעדר או בתוספת Cyan נקבל היפוך של צבעים מסוימים.

- בדומה להסבר על שילוב הצבעים במניפולציה הראשונה, גם במניפולציה השניה נוכל לראות לאחר החלפת הערוצים Cyan ו – Yellow שכעת נוצר חילוף בין הצבעים האדום והכחול, וזאת מאחר שצבעים אלה נוצרים על ידי שילוב של המג'נטה עם הצבע הצהוב ועם הצבע Cyan בהתאמה.
- במניפולציה השלישית נוכל לראות כי לאחר הפעלת מסנן גאוסיאן על הערוץ הרביעי, נקבל טשטוש של השפות בתמונה כולה וזאת מאחר שהערוץ הרביעי מחזיק מידע על כל הפיקסלים.

3. HSV

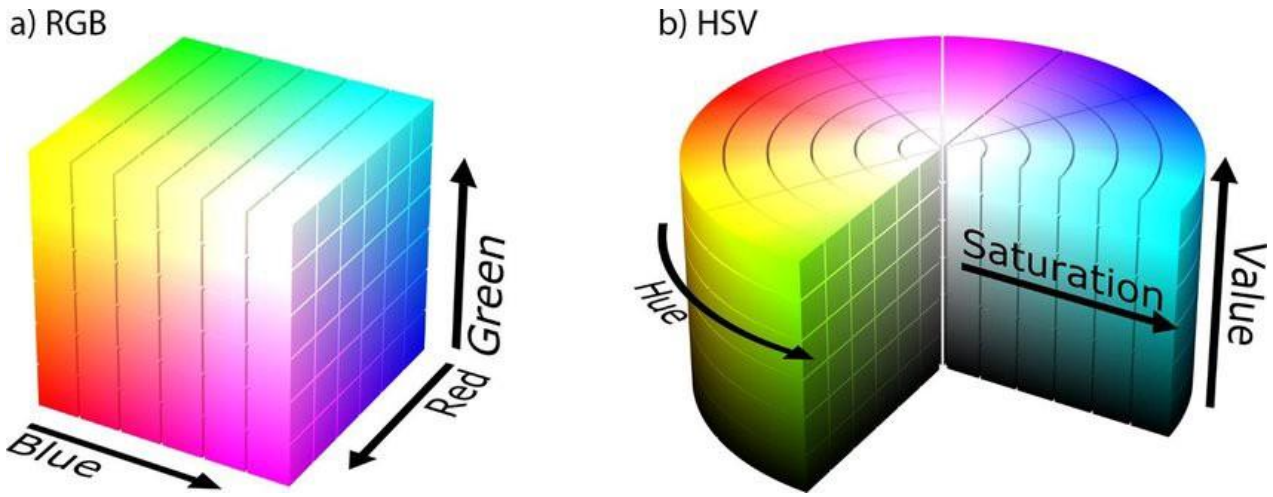
1. בנוסף למודל ה – RGB, HSV הינו מודל נוסף לייצוג צבעים כאשר במודל זה נייצג כל צבע על ידי 3 פרמטרים:

- H (Hue) – הגוון מייצג את הצבעים על גלגל הצבעים, כלומר הגוון הוא חלק הצבע של מודל הצבע, והוא מתבטא כמספר 0 עד 360 מעלות:

צבע:	**מעלות:**
אדום	$0 - 60$
צהוב	$60 - 120$
ירוק	$120 - 180$
טורקיז	$180 - 240$
כחול	$240 - 300$
מג'נטה	$300 - 360$

- S (Saturation) – רוויה הוא מספר המייצג את כמות האפור בצבע, בין 0 ל -100%. אפקט דהוי יכול להתקבל על ידי הפחתת הרוויה לכיוון 0 כדי להציג אפור יותר. עם זאת, הרוויה נתפסת לעתים בטווח של $0 - 1$, כאשר 0 הוא אפור (או צבע ניטרלי, תלוי ב – V), ו- 1 הוא צבע עיקרי (צבע שלא התערבב עם אפור).
- V (Value) – הערך עובד יחד עם הרוויה ומתאר את בהירות או עוצמת הצבע, בין 0 ל -100%, כאשר 0 הוא שחור לחלוטין ו -100 הוא הבהיר ביותר וחושף את הצבע הגדול ביותר (הקרוב ביותר ללבן).

נציג תמונה הממחישה את ייצוג זה בהשוואה ל - RGB:



2. נבנה פונקציה משלנו בשם: *'dip rgb2hsv()'*.

הפונקציה ממירה תמונה בייצוג RGB לייצוג של HSV כך:

- ראשית, ננרמל את ערכי ה – R, G, B כך שיו בטווח ערכים של $0 - 1$, כלומר נחלק כל ערך ב 255:

$$R' = {}^{R}/_{255}$$
$$G' = {}^{G}/_{255}$$
$$B' = {}^{B}/_{255}$$

- נחשב את הערך המקסימלי והמינימלי של ערכי ה – R, G, B ונמצא את ההפרש ביניהם:

$$C_{max} = \max(R', G', B')$$
$$C_{min} = \min(R', G', B')$$
$$C_{\Delta} = C_{max} - C_{min}$$

- כעת נחשב את הגוון כך:

Hue calculation:

$$H = \begin{cases} 0^{\circ} \quad , \quad C_{\Delta} = 0 \\ 60^{\circ} \times \left(\left(\frac{G' - B'}{C_{\Delta}}\right) mod\ 6\right) \quad , C_{max} = R' \\ 60^{\circ} \times \left(\frac{B' - R'}{C_{\Delta}} + 2\right) \quad , \quad C_{max} = G' \\ 60^{\circ} \times \left(\frac{R' - G'}{C_{\Delta}} + 4\right) \quad , \quad C_{max} = B' \end{cases}$$

- כעת נחשב את הרוויה כך:

Saturation calculation:

$$S = \begin{cases} 0 \quad , & C_{max} = 0 \\ \dfrac{C_{\Delta}}{C_{max}} & , C_{max} \neq 0 \end{cases}$$

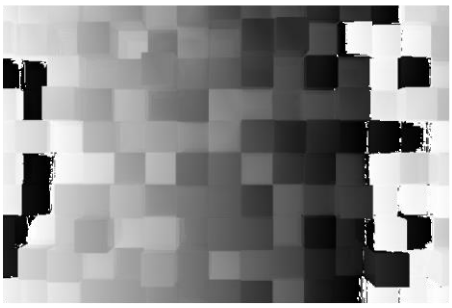
- כעת נחשב את הבהירות כך:

Value calculation:

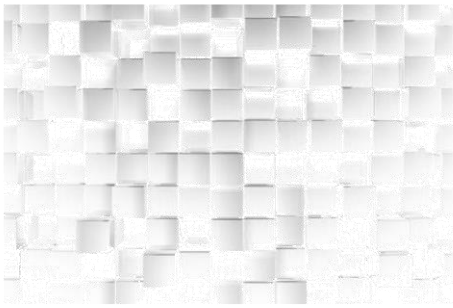
$$V = C_{max}$$

3. נפעיל את הפונקציה שבנינו על התמונה שלנו, ונציג כל ערוץ בנפרד:

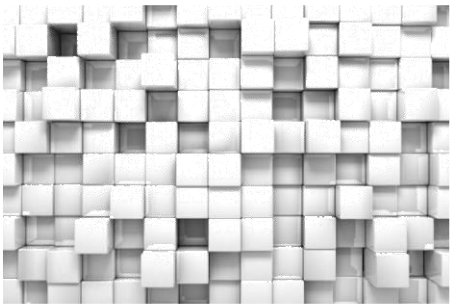
H Channel



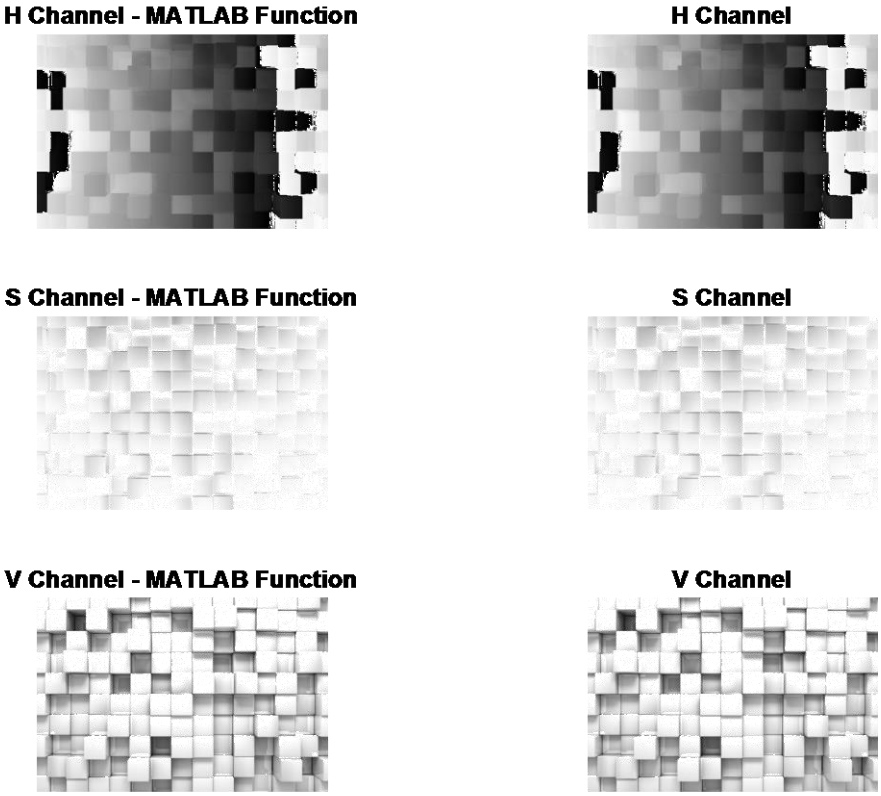
S Channel



V Channel



4. נבצע השוואה בין התמונות שקיבלנו בכל ערוץ עבור הפונקציה שבנינו לבין התמונות שקיבלנו עבור הפונקציה הבנויה של MATLAB, *'rgb2hsv()'*:



במבט ראשוני נוכל לראות את הדמיון בין התמונות. נראה את הדמיון גם בצורה כמותית על ידי חישוב המרחק האוקלידי על ידי שימוש בפונקציה *pdist()*. פונקציה זו מודדת את המרחק בין כל פיקסל בתמונה אחת לשאר הפיקסלים בתמונה האחרת. מכיוון שמעניין אותנו רק המרחק בין שני פיקסלים שנמצאים באותו מיקום נסתכל על האלכסון לאחר הפעלת הפונקציה, ונסכום את השגיאות של כל הפיקסלים כך:

```
Noam Atias 311394357, Chanel Michaeli 208491787
205 D_V_trace = trace(D_V);

D_H_trace =

   6.5319e-13


D_S_trace =

   5.5368e-13


D_V_trace =

     0
```

נוכל לראות שקיבלנו ערכים שקרובים מאוד ל – 0 . כלומר המרחק של כל ערוץ במרחב הערוצים של HSV לאחר המרה ממרחב ערוצים של RGB על ידי הפונקציה שלנו אל מול הפונקציה המובנית של MATLAB, קטן מאוד ולכן התוצאות דומות מאוד.

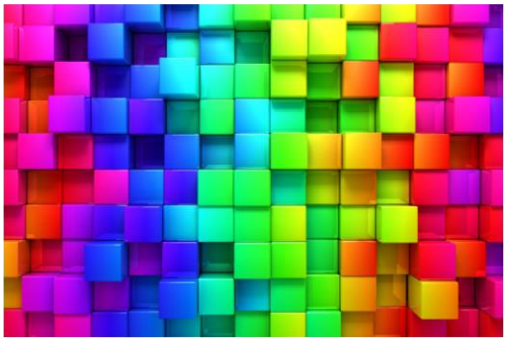
5. נחזור על סעיף 6 משאלה 1:

נפעיל מניפולציות על תמונת ה – HSV המנורמלת בכל ערוץ בנפרד באופן הבא:

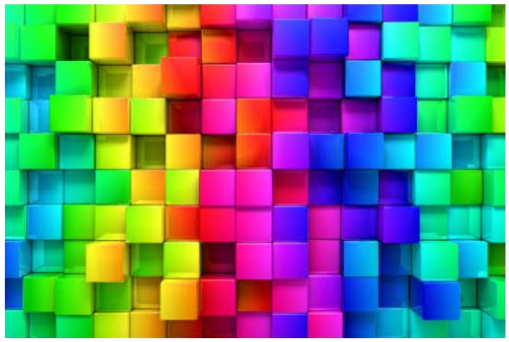
- על הערוץ הראשון (H) נפעיל פונקציה לינארית: $F(H) = H + 180°$.
- בערוץ השלישי נבצע מניפולציה לינארית נוספת כך: $F(V) = -0.5\,V + 1$.
- על הערוץ השלישי נפעיל פילטר גאוסיאן (השתמשנו במימוש של הפילטר שפיתחנו בעבודה 1), כתוצאה מכך נסיר את כל התדרים הגבוהים של הערוץ, כלומר נגרום לשפות של הערוץ להיות פחות בולטות וכך ניצור מריחה או טשטוש של השפות בתמונה.

נציג את הערוצים לפני ואחרי הפעלת המניפולציות בצבעים:

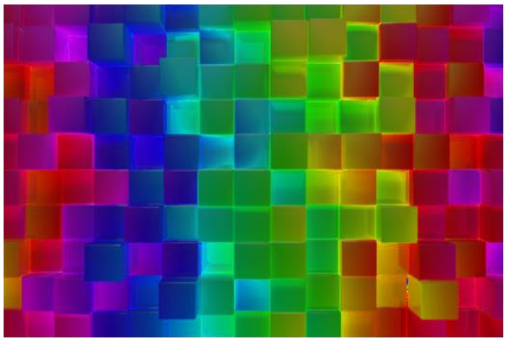
Original image



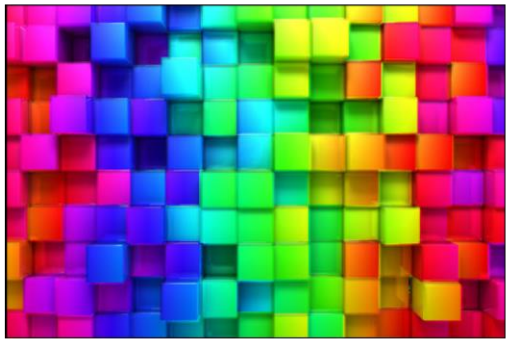
Linear manipulated Hue Channel - add 180 degrees



Linear manipulated Value Channel

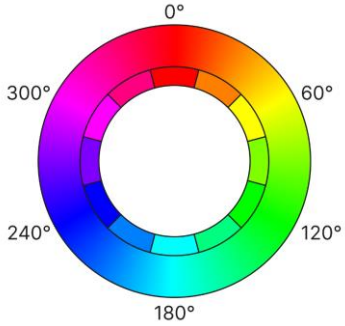


Gaussian manipulated Value Channel



ניתן הסבר לכל מניפולציה:

- במניפולציה הראשונה הוספנו 180 מעלות לערוץ H כך שגרמנו להיפוך של צבעים, כך שצבע וורוד/ סגול הפך להיות צבע ירוק וההפך, צבע כחול הפך לצבע צהוב וההפך, וכן הלאה לפי גלגל הצבאים באיור 1:
- במניפולציה השניה ביצענו היפוך של ערכים בערוץ השלישי, כלומר הערוץ של ה –



איור 1: גלגל צבעים HSV

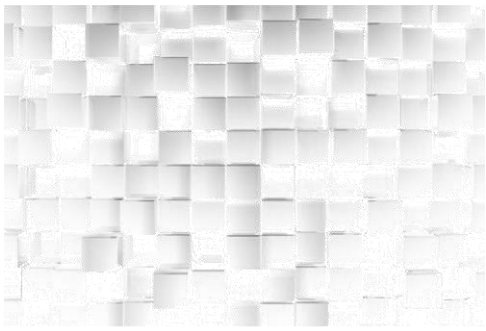
Value. כך מה שהיה בעל ערך גבוה של 1 יהפוך כעת להיות בעל ערך של 0.5 וערך של 0 יהפוך להיות 1. רוב הערכים יקטנו ולכן נקבל ירידה בעוצמת הצבע. בחלק מהמקומות נוכל לראות שינו ממש של הצבע מכיוון שכעת צבע מסוים שהיה דומיננטי קודם כעת הרבה פחות מה ששינה את הרכב הצבעים באותו אזור.

- במניפולציה השלישית נוכל לראות כי לאחר הפעלת מסנן גאוסיאן על הערוץ ה – Value , נקבל טשטוש קל בחדות של התמונה.

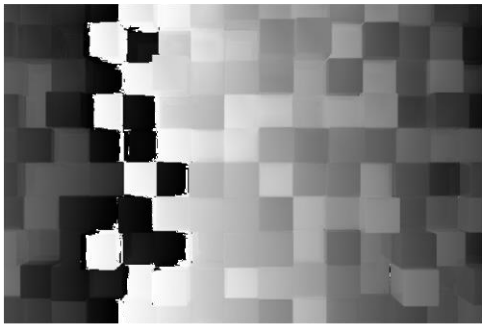
6. ראשית נחליף בין הערוצים של תמונת ה - RGB :
   - הערוץ הראשון הפך להיות B.
   - הערוץ השני הפך להיות R.
   - הערוץ השלישי הופך להיות G.

כעת נבצע המרה למרחב הצבעים של HSV על ידי שימוש בפונקציה שבנינו, ונקבל:

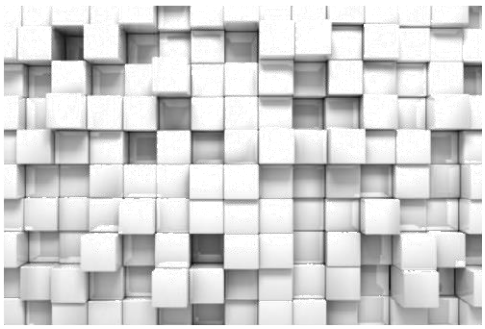
S swap RGB Channel



H swap RGB Channel



V swap RGB Channel



נוכל לראות שהערוץ היחיד שהשתנה הוא הערוץ H, מכיוון שלאחר שביצענו החלפה בין הערוצים הדבר היחידי שהשתנה זה שכל ערוץ קיבל תכונות של הערוץ האחר, כלומר נקבל החלפה בצבעים בדומה למה שהסברנו בסעיף 3.5 במניפולציה הראשונה (עם הגלגל צבעים).

ערוץ הרוויה וערוץ הבהירות לא השתנו מכיוון שלא שינו שום ערך שלהם ולכן זהים למה שקיבלנו גם לפי ההחלפה בין הערוצים.

4. L*a*b

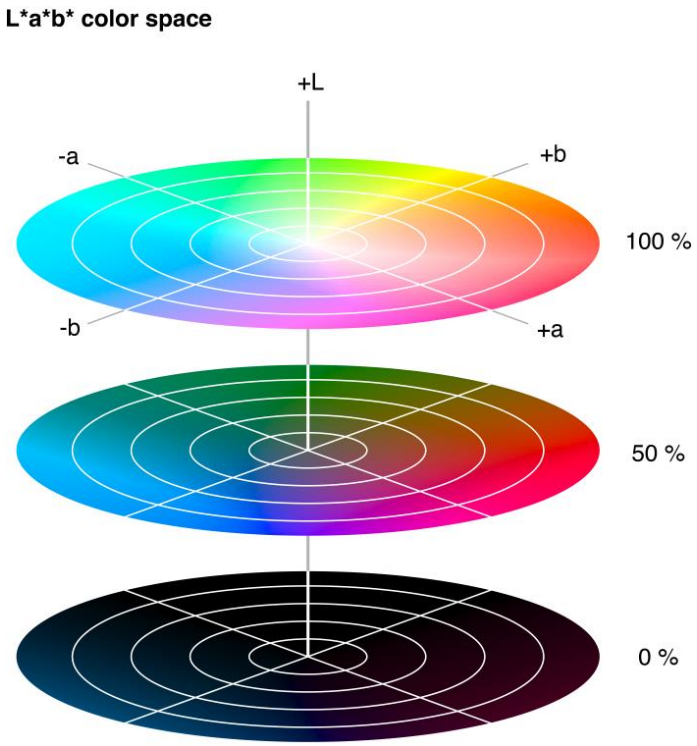
1. מרחב הצבעים ***L*a*b*** הוא מרחב תלת מימדי המכסה את כל טווח תפיסת הצבעים האנושית. L* מתאר את הבהירות ואילו a* ו-b* מתארים את ארבעת הצבעים הייחודיים של הראיה האנושית : אדום, ירוק, כחול וצהוב.

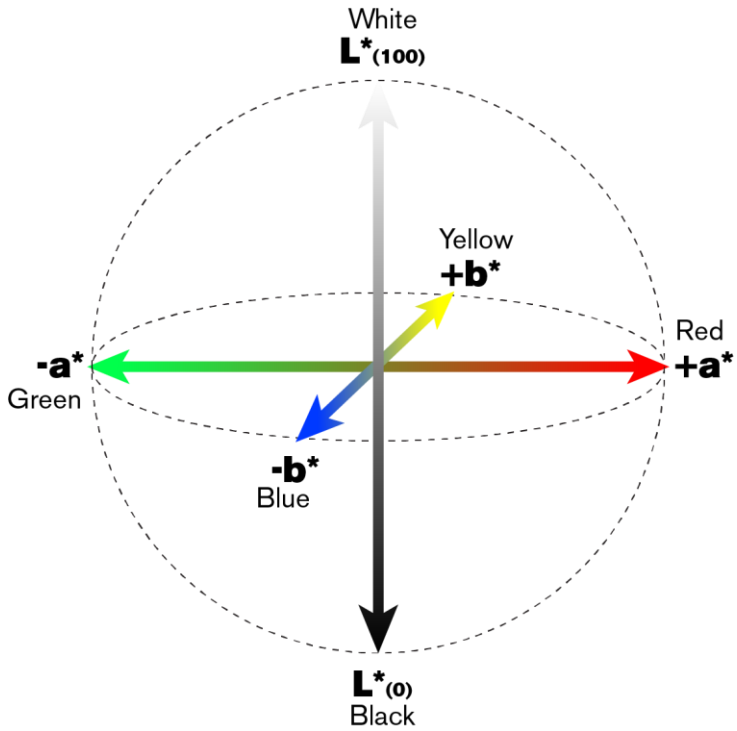
מרחב זה מבוסס על הניגודיות בין זוגות הצבעים אדום וירוק לעומת צהוב וכחול. ציר a* יחסי לצבעים ירוק – אדום, כאשר ירוק מיוצג על ידי ערכים שלילים יותר ואילו אדום בערכים חיובים. ציר b* יחסי לצבעים כחול – צהוב כאשר ערכים שלילים יותר מייצגים את הצבע הכחול ואילו ערכים חיוביים מייצגים את הצבע הצהוב. ערך הבהירות L* מוגדר להיות על ציר שנע בין 0 ל – 100 כאשר 0 הוא שחור ו-100 הוא לבן.

הצירים a* ו – b* אינם מוגבלים והם יכולים בקלות לחרוג מ – ± 150 כדי לכסות את טווח האדם, אך נהוג להגביל את הצירים לטווח של $[-128\,,127]$, כאשר הערכים מחושבים ביחס לצבע הלבן.

מרחב זה מייצג את כל מרחב הצבעים האנושי וזה היתרון הבולט שלו. רכיב ה-L* תואם כמעט לחלוטין לתפיסת הבהירות האנושית. אין אחידות גמורה בצירים של מרחב צבעים זה אך הוא יעיל בזיהוי שינויים קטנים בצבע.

נציג תמונה הממחישה איך מרחב צבעים זה נראה:

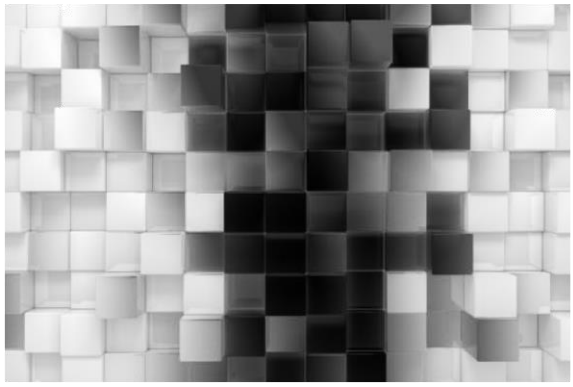




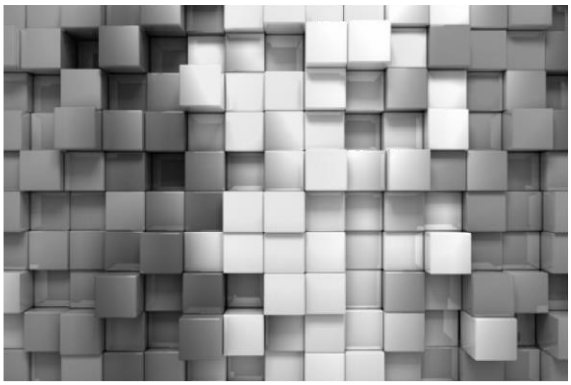
2. המרנו את התמונה המנורמלת שלנו למרחב L*a*b על ידי הפונקציה של מטלב *'rgb2lab()'*.

כעת נציג כל ערוץ בנפרד:

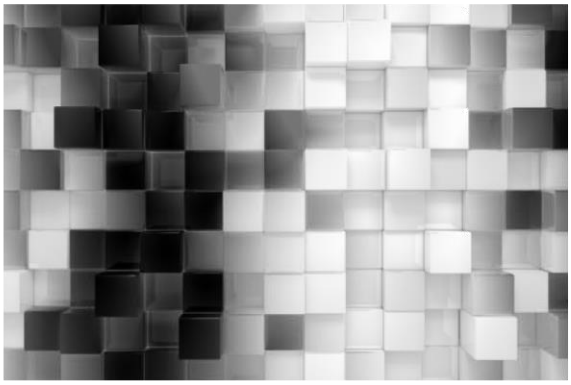
L*a*b* color space - Channel a



L*a*b* color space - Channel L



L*a*b* color space - Channel b



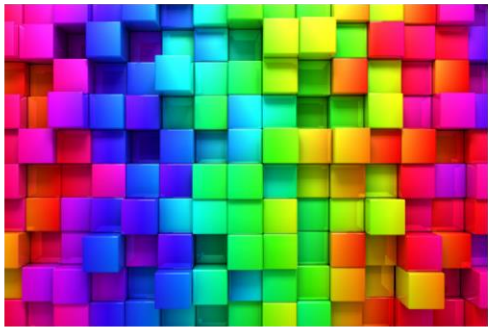
3. נחזור על סעיף 6 משאלה 1:

נפעיל מניפולציות על תמונת ה – L*a*b בכל ערוץ בנפרד באופן הבא:

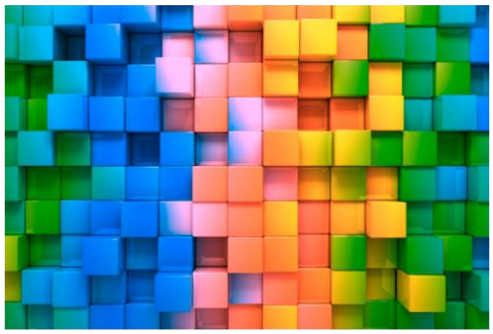
- על הערוץ השני (a) נפעיל פונקציה לינארית: $F(a) = -a + 1$.
- נחליף בין הערוץ השני (a) לבין הערוץ השלישי (b). כעת, הערוץ השלישי יהיה בעל תכונות הצבעים של ערוץ a, והערוץ השני יהיה בעל תכונות של הערוץ b .
- על הערוץ הראשון (L) נפעיל פילטר גאוסיאן ($filter = 5 \times 5$ ו - $\sigma = 1$.) כתוצאה מכך נסיר את כל התדרים הגבוהים של הערוץ, כלומר נגרום לשפות של הערוץ להיות פחות בולטות וכך ניצור מריחה או טשטוש של השפות בתמונה.

נציג את הערוצים לפני ואחרי הפעלת המניפולציות בצבעים:

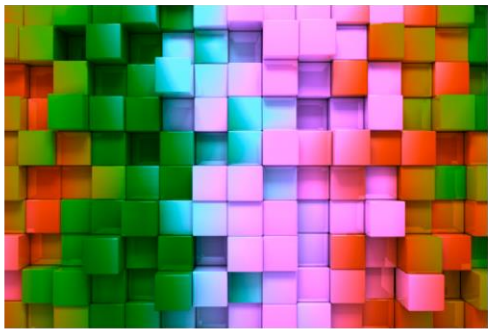
Original image



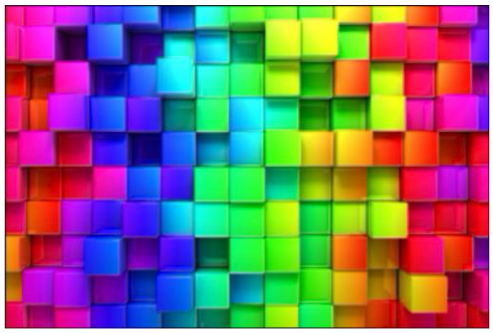
Linear manipulated a Channel



Manipulated swap Channels a and b



Gaussian manipulated L Channel



ניתן הסבר לכל מניפולציה:

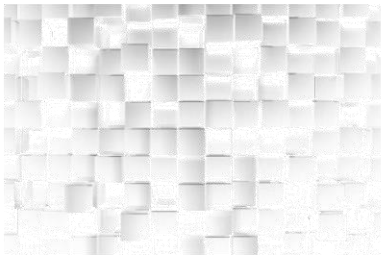
- נשים לב כי כאשר הפעלנו מניפולציה של פונקציה לינארית על ערוץ (a), בעצם ביצענו היפוך של הציר כך שכעת הערכים החיוביים של (a) מבטאים את הצבע הירוק ואילו ערכים שליליים מבטאים את הצבע האדום. לכן, קיבלנו כי באזורים בהם היה הצבע האדום, כעת יהיה הצבע הירוק ולהפך. בנוסף נשים לב כי הגוונים של הצבעים השתנו וזאת מאחר ששילובי הצבעים השתנו, לדוגמה באזור הצהוב, לפני המניפולציה באזור זה הערכים של ערוץ (a) היו שליליים ולאחר המניפולציה הם יהיו חיוביים ולכן הגוון של הצבע הצהוב ישתנה לצבע יותר כתמתם מאחר שכעת שילוב הצבעים הוא בין צהוב לאדום.
- במניפולציה הבאה החלפנו בין הערוצים (a) ו- (b). לכן נקבל שבאזור שהיה בצבע ירוק לפני המניפולציה, כעת יהיה בצע ורוד וזאת מאחר שאזור זה מקבל כעת ערכים שליליים בערוץ (b) וערכים חיוביים בערוץ (a), לכן שילוב הצבעים החדשים של אדום וכחול מייצר את הצבע הסגול. דוגמה נוספת היא באזור שלפני המניפולציה היה בצבע כחול וכעת הפך לצבע ירוק ככה, וזאת מאחר שכעת באזור זה יש ערכים חיוביים בציר (b) וערכים שליליים בציר (a) ולכן נקבל את שילוב הצבעים של ירוק עם צהוב, ובגלל הבהירות הנמוכה באזור זה שנובעת מערכים נמוכים בציר (L) נקבל ירוק כהה. לעומת זאת, הצבע

האדום נשאר אותו הדבר וזאת מאחר שאזורים אלה מקבלים פחות או יותר את אותם הערכים בערוץ (a) ו- (b).

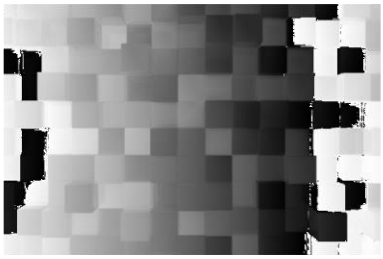
- במניפולציה האחרונה ביצענו פילטר גאוסיאן על ערוץ (L) וקיבלנו תמונה מטושטשת יותר שבה השפות של התמונה פחות חדות כי הוצאנו את התדרים הגבוהים של התמונה. דבר זה התאפשר כי בתמונה כולה הערכים של ערוץ (L) יחסית גבוהים.

4. נשווה לערוצים של מרחב הצבעים HSV בשאלה 3.6:

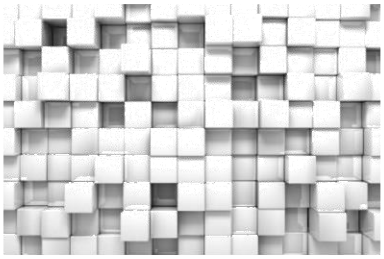
S channel



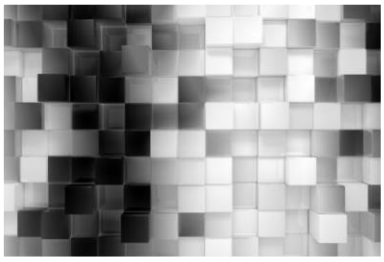
H channel



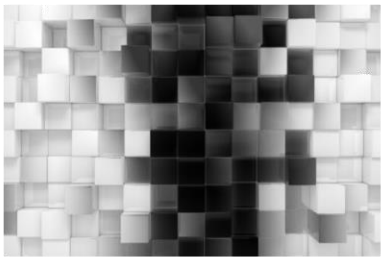
V channel



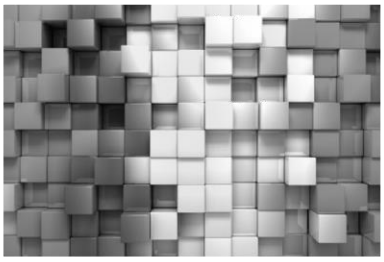
b channel



a channel



L channel



ראשית, נבחין כי ערוץ (L) במרחב הצבעים L*a*b דומה בתפקידו ובגווניו לערוץ V מאחר ששני ערוצים אלה אחראיים על בהירות התמונה. בנוסף, ניתן לתאר את גלגל הצבעים של מרחב HSV על ידי מערכת הצירים של מרחב הצבעים L*a*b, כלומר בעוד שבמערכת HSV הצבעים מתוארים על ידי זווית בערוץ H, הצבעים במרחב L*a*b מתוארים על ידי שני צירים מאונכים זה לזה שיוצרים את כל קומבינציות הצבעים.

5. Color Segmentation

המטרה שלנו הייתה למצוא את הפקק הכחול של בקבוק הסודה בכל אחת מהתמונות הנתונות.

נפרט על האלגוריתם שבנינו ונציג את התמונות:

- ראשית קראנו כל תמונה בנפרד והעברנו אותה ל – *Type double* ולאחר מכן ביצענו נרמול.
- לאחר מכן יצרנו את ה – Binary mask של כל תמונה בנפרד על ידי פונקציה שבנינו בשם: *'Binary_mask'*. פונקציה זו מבצעת ראשית המרה ממרחב הצבעים RGB למרחב הצבעים HSV על ידי שימוש בפונקציה המובנית של MATLAB, *'rgb2hsv()'*.
  פונקציה זאת מקבלת את התמונה המנורמלת, ערך מינימלי וערך מקסימלי עבור הערוץ Hue, שיגדירו תחום המייצג את הצבע הרצוי (צבע כחול של הפקק) אותו נרצה למצוא בתמונה. לאחר מכן הפונקציה הופכת כל ערך של פיקסל שלא נמצא בטווח הערכים שנקבע ל – 0. לאחר מכן נשתמש בפונקציה שבנינו בעבודה קודמת שמבצעת פילטר חציון, וזאת על מנת שנוכל להימנע משפות ורעשים אשר בהמשך יאפשר לנו למצוא את מיקום הפקק בקלות יותר. לאחר המעבר בפילטר נבחר כל ערך בתמונה החדשה שערכו אינו 0, כלומר הוא נמצא בטווח הערכים של הערוץ כפי שהגדרנו, נהפוך אותו להיות ערך של 1, כך שבסוף הפונקציה מחזירה Binary mask.
- לאחר שנקבל תמונה בינארית, נמצא את מיקום הפקק על ידי פונקציה שבנינו בשם: *'find_loc'*. פונקציה זו מקבל את התמונה הבינארית וערך k אשר יגדיר גודל של פילטר אשר כולו מורכב מאחדות. לאחר מכן נעביר את הפילטר על גבי התמונה הבינארית, ונסכום את האיברים של התמונה (הכפלה של הפילטר בחלק של התמונה בגודל של הפילטר, וסכימה של התמונה שהתקבלה מההכפלה). נמצא את הסכום הגדול ביותר בכל איטרציה ונחזיר את מיקום הפילטר על גבי התמונה (נמצא את המרכז), כך שלאחר כל האיטרציות הסכום הגדול ביותר יהיה לאחר המעבר של הפילטר על הפקק ועדכון מיקום הפילטר על גבי התמונה יהווה את מיקום הפקק בתמונה כולה. בסיום נקבל את מיקום הפקק על גבי התמונה.
- נשרטט עיגול כל גבי התמונה המקורית במיקום שמצאנו ברדיוס אותו נבחר, על ידי שימוש בפונקציה המובנית של MATLAB : *'insertShape()'*.

האלגוריתם שבנינו לא מושלם, ולכן הוא יכול להיכשל במקרה שבו ניקח פילטר בגודל שלא מתחלק בגודל התמונה או כאשר נבחר k קטן מידי כל שהפילטר יהיה מאוד גדול ויהיה קשה לאלגוריתם לזהות את הפקק על ידי הסכימה לאחר מעבר הפילטר. בחרנו את הפרמטרים כך ש k יהיה מספיק גדול כדי שיצליח לזהות את הפקק עצמו ובנוסף בחרנו את טווח הערכים להיות $\left[0.61, \frac{2}{3}\right]$, מכיוון שזהו הטווח המנורמל עבורו הערוץ Hue הוא בצבע כחול.

נציג את התמונות שהמקוריות, את ה – Binary mask ואת התמונה אשר הפקק מוקף בעיגול אדום עבור כל אחד מהמקרים:

<u>עבור Cap 1:</u>

Original cap 1

Binary mask cap 1

circle the cap 1 of the soda

עבור Cap 2:

Original cap 2

Binary mask cap 2

circle the cap 2 of the soda

עבור Cap 3:

Original cap 3

Binary mask cap 3

circle the cap 3 of the soda