ראייה אנושית גישה חישובית
תרגיל 3

מגישים: רז דרזי ורן שחם

11 ביוני 2017

חלק I
תיאורטי

נגדיר את הקונבולוציה של שתי מטריצות X,Y כך:

$$Z\left[m,n\right]=\sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty}X\left[i,j\right]Y\left[m-i,n-j\right]$$

ונסמן:

$$Z=X*Y$$

שאלה 1

נראה שקונבולוציה היא אסוציאטיבית. יהיו X,Y,Z מטריצות. אזי:

$$\begin{aligned}
\left(\left(X*Y\right)*Z\right)\left[l,k\right] &= \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}\left(X*Y\right)\left[m,n\right]Z\left[l-m,k-n\right]\\
&= \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}\left(\sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty}X\left[i,j\right]Y\left[m-i,n-j\right]\right)Z\left[l-m,k-n\right]\\
&= \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}\sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty}\left(X\left[i,j\right]Y\left[m-i,n-j\right]Z\left[l-m,k-n\right]\right)\\
&= \sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty}\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}X\left[i,j\right]Y\left[m-i,n-j\right]Z\left[l-m,k-n\right]\\
&= \sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty}X\left[i,j\right]\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}Y\left[m-i,n-j\right]Z\left[l-m,k-n\right]\\
&= \sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty}X\left[i,j\right]\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}Y\left[m,n\right]Z\left[\left(l-i\right)-m,\left(k-j\right)-n\right]\\
&= \sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty}X\left[i,j\right]\left(Y*Z\right)\left[l-i,k-j\right]=\left(X*\left(Y*Z\right)\right)\left[l,k\right]
\end{aligned}$$

לכל l,k, ולכן שוויון מתקיים:

$$(X*Y)*Z = X*(Y*Z)$$

שאלה 2

יהיו a,b סקלרים ו־X,Y,Z מטריצות. אזי:

$$\begin{aligned}
(X*(aY+bZ))[m,n] &= \sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty} X[i,j](aY+bZ)[m-i,n-j] \\
&= \sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty} X[i,j]\,aY[m-i,n-j] + X[i,j]\,bZ[m-i,n-j] \\
&= \sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty} (X[i,j]\,aY[m-i,n-j]) + \sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty} (X[i,j]\,bZ[m-i,n-j]) \\
&= a\sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty} (X[i,j]\,Y[m-i,n-j]) + b\sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty} (X[i,j]\,Z[m-i,n-j]) \\
&= aX*Y[m,n] + bX*Z[m,n]
\end{aligned}$$

לכל m,n, לכן מתקיים $X*(aY+bZ) = aX*Y + bX*Z$.

שאלה 3

עבור K_x, K_y נגדיר את הלפלסיאן של I באופן הבא:

$$\nabla^2 I = (I*K_x)*K_x + (I*K_y)*K_y$$

נניח ש־ $K_y = \frac{1}{2}\begin{pmatrix}-1\\1\end{pmatrix}, K_x = \frac{1}{2}\begin{pmatrix}1 & -1\end{pmatrix}$ ונגדיר $K_l = \frac{1}{4}\begin{pmatrix} & 1 & \\ 1 & -4 & 1 \\ & 1 & \end{pmatrix}$. מתקיים:

$$\begin{aligned}
\nabla^2 I &\overset{1}{=} (I*K_x)*K_x + (I*K_y)*K_y \overset{2}{=} I*(K_x*K_x) + I*(K_y*K_y) \\
&\overset{3}{=} I*(K_x*K_x + K_y*K_y) \overset{4}{=} I*\left(\left(\frac{1}{2}\begin{pmatrix}1 & -1\end{pmatrix}\right)*\left(\frac{1}{2}\begin{pmatrix}1 & -1\end{pmatrix}\right) + \left(\frac{1}{2}\begin{pmatrix}-1\\1\end{pmatrix}\right)*\left(\frac{1}{2}\begin{pmatrix}-1\\1\end{pmatrix}\right)\right) \\
&\overset{5}{=} I*\left(\frac{1}{4}\begin{pmatrix}1 & -2 & 1\end{pmatrix} + \frac{1}{4}\begin{pmatrix}1\\-2\\1\end{pmatrix}\right) \overset{6}{=} I*\frac{1}{4}\begin{pmatrix} & 1 & \\ 1 & -4 & 1 \\ & 1 & \end{pmatrix} \overset{7}{=} I*K_l
\end{aligned}$$

כאשר:

1. מהגדרת הלפלסיאן
2. שאלה 1 (אסוציאטיביות הקונבולציה)
3. שאלה 2

4. הגדרת K_x, K_y
5. הגדרת הקונבולוציה. למשל, $\left(K_x' * K_x'\right)[0,0]$ כאשר $K_x' = \begin{pmatrix} 1 & -1 \end{pmatrix}$ היא:

$$\begin{aligned}\left(\begin{pmatrix} 1 & -1 \end{pmatrix} * \begin{pmatrix} 1 & -1 \end{pmatrix}\right)[0,0] &= \sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty} \begin{pmatrix} 1 & -1 \end{pmatrix}[i,j] \begin{pmatrix} 1 & -1 \end{pmatrix}[0-i,0-j] \\ &= \sum_{j=-\infty}^{\infty} \begin{pmatrix} 1 & -1 \end{pmatrix}[0,j] \begin{pmatrix} 1 & -1 \end{pmatrix}[0,-j] \\ &= \sum_{j=-1}^{1} \begin{pmatrix} 1 & -1 \end{pmatrix}[0,j] \begin{pmatrix} 1 & -1 \end{pmatrix}[0,-j] \\ &= 0 \cdot (-1) + 1 \cdot 1 + (-1) \cdot 0 = 1\end{aligned}$$

ובאופן דומה $\left(K_x' * K_x'\right)[0,1] = -2$ ו־$\left(K_x' * K_x'\right)[0,2] = 1$ ולכן $K_x * K_x = \frac{1}{4}\begin{pmatrix} 1 & -2 & 1 \end{pmatrix}$.

6. הגדרת חיבור מטריצות (בהתאם להנחות בתרגיל שחלק שאינו מוגדר במטריצה הוא 0 ושהאיבר האמצעי במטריצה הוא האינדקס ה־0 בכל מימד)
7. הגדרת K_l

שאלה 4

נניח שקיים K_α כך ש־$K_l * K_\alpha = \overline{\delta}$ כאשר:

$$\overline{\delta}[i,j] = \begin{cases} 1 & i,j = (0,0) \\ 0 & \text{otherwise} \end{cases}$$

נשים לב:

$$\left(\nabla^2 I\right) * K_\alpha = (I * K_l) * K_\alpha = I * (K_l * K_\alpha) = I * \overline{\delta}$$

מהגדרת הקונבולוציה, לכל m,n:

$$\left(I * \overline{\delta}\right)[m,n] = \sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty} I[i,j]\,\overline{\delta}[m-i,n-j] = I[m,n]$$

כאשר המעבר האחרון נובע מהגדרת $\overline{\delta}$:

$$\overline{\delta}[m-i,n-j] = 1 \iff m-i = n-j = 0 \iff i = m, j = n$$

ולכן:

$$\left(\nabla^2 I\right) * K_\alpha = I$$

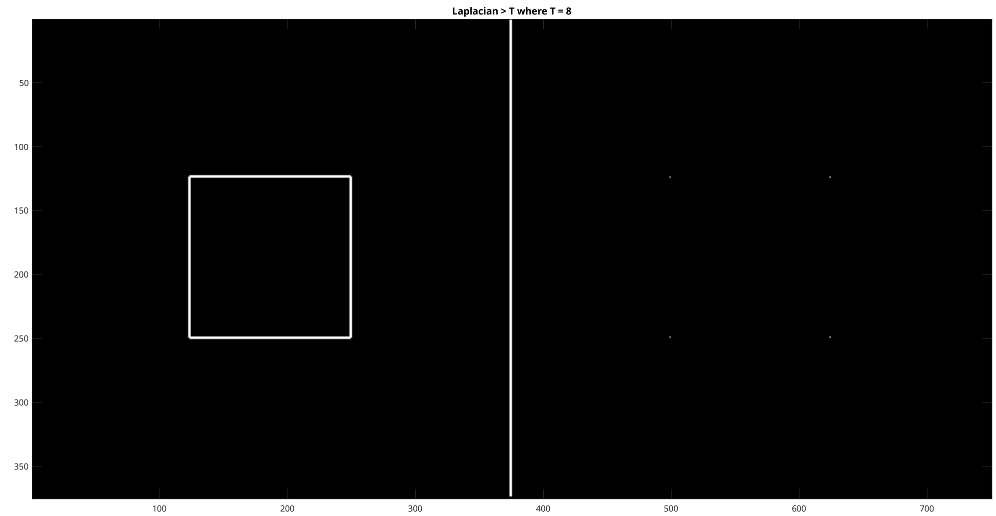
כנדרש.

חלק II

תכנותי

שאלה 4

הריבוע השמאלי נראה בהיר יותר. לאחר חיפוש אחר ערכי T אפשריים, מצאנו כי $T = 8$ מקיים את התכונה הבאה: עבור החצי השמאלי של התמונה ישנם $> 1{,}000$ פיקסלים שערכם בערך מוחלט גדול מ־T; עבור החצי הימני ישנם 4 פיקסלים כאלה. באיור המצורף אפשר



איור 1: שאלה 4 ־ הלפלסיאן בריבוע השמאלי גדול מ־T ובריבוע הימני קטן מ־T (למעט ב־4 פינות הריבוע)

לראות תכונה זאת.

שאלה 5

כמובן שניתן להבחין בצלב הבהיר על אלכסוני התמונה (פינות הריבועים). גם כאן, לאחר חיפוש ממצה על ערכי T אפשריים התקבל ש־$T = 5$ מקיים את התכונה: ישנם פיקסלים גדולים מ־T על אלכסון התמונה (52 כאלה) וכן אין פיקסלים גדולים מ־T מחוץ לאלכסון. כמו בשאלה הקודמת, האיור המצורף מתאר את הממצא.

שאלה 6

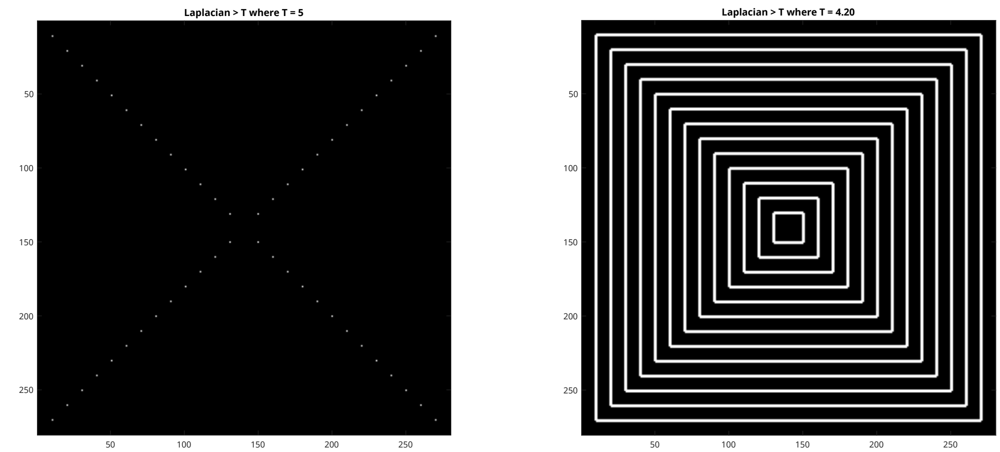
לאחר חיפוש ממצה על ערכי T אפשריים, לא מצאנו ערך שיכול להסביר את התופעה ־ כלומר לא קיים ערך T עבורו הלפלסיאן בחצי הטבעת העליון גדול מ־T ובחצי התחתון קטן מ־T. נסיק שלא קיים T כזה ־ כי כמות הפיקסלים בשני חציי הטבעת יורדת באופן הדרגתי ל־0 עבור ערכי T גדלים, ובקצב דומה בשניהם (ראו איור).

שאלה 7

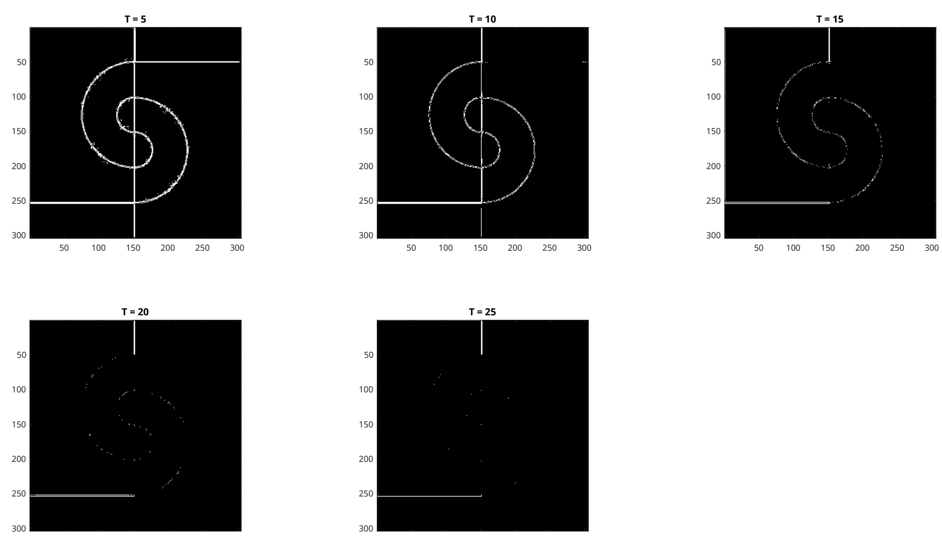
כפי שראינו בשאלות 4־6, חלק מהאשליות ניתן להסביר באמצעות ה־Laplacian וחלק לא. בפרט, ראינו שאת האשליות שהופיעו בשאלות $4, 5$ ניתן להסביר באופן זה וזו שב־6 לא.

היפותזה אלטרנטיבית היא שהצבע הנתפס הוא פונקצייה כלשהי של מאפיינים בסיסיים בתמונה ושל ידע מוקדם (prior) לגבי התמונה (או לגבי תמונות טבעיות, באופן כללי). כלומר, ישנה רגולריות בתמונות אותה אנחנו לומדים ומנסים להכליל לתמונות חדשות. דוגמה ל־prior כזה הוא שלכל חומר יש גוון אחיד. כלל זה יסביר את האפקט ב־kofka-ring באופן הבא: אנחנו תופסים את החצי התחתון של הטבעת כעשוי מאותו "חומר" של הרקע השמאלי, ואת החצי העליון כמאותו חומר של הרקע הימני.[1] תחת ההנחה שאובייקט מאותו חומר הוא בעל גוון אחיד, נתפוס את שני חציי הטבעת כבעלי צבע שונה (בגלל ההבדל הגדול בצבעים בין הרקע הימני לשמאלי). מאפייני התמונה במקרה הזה יתפסו תפקיד משני בהבנת הצבע של הטבעת. במילים אחרות, ידע סמנטי ("גבוה") משפיע על תהליכי תפיסה "נמוכים".

[1] גם כאן נכנס prior לפיו אובייקטים הם בעלי צורה "חלקה" יחסית, כלומר הם לא נחתכים ומשתנים במהירות גדולה מדי.



איור 2: שאלה 5 ־ משמאל: הלפלסיאן בפינות הריבוע (על האלכסון) גדול מ־T ובשאר התמונה קטן מ־T; מימין: תמונה דומה עבור ערך סף $T-\varepsilon$ ־ כלומר מראה שהסף T שמצאנו הדוק



איור 3: שאלה 6 ־ ערכי הלפלסיאן הגדולים מ־T עבור ערכי T שונים. אפשר לראות שאין ערך בו החלק העליון הוא בעל יותר פיקסלים הגדולים מ־T גדול מבחלק התחתון.

שאלה 9

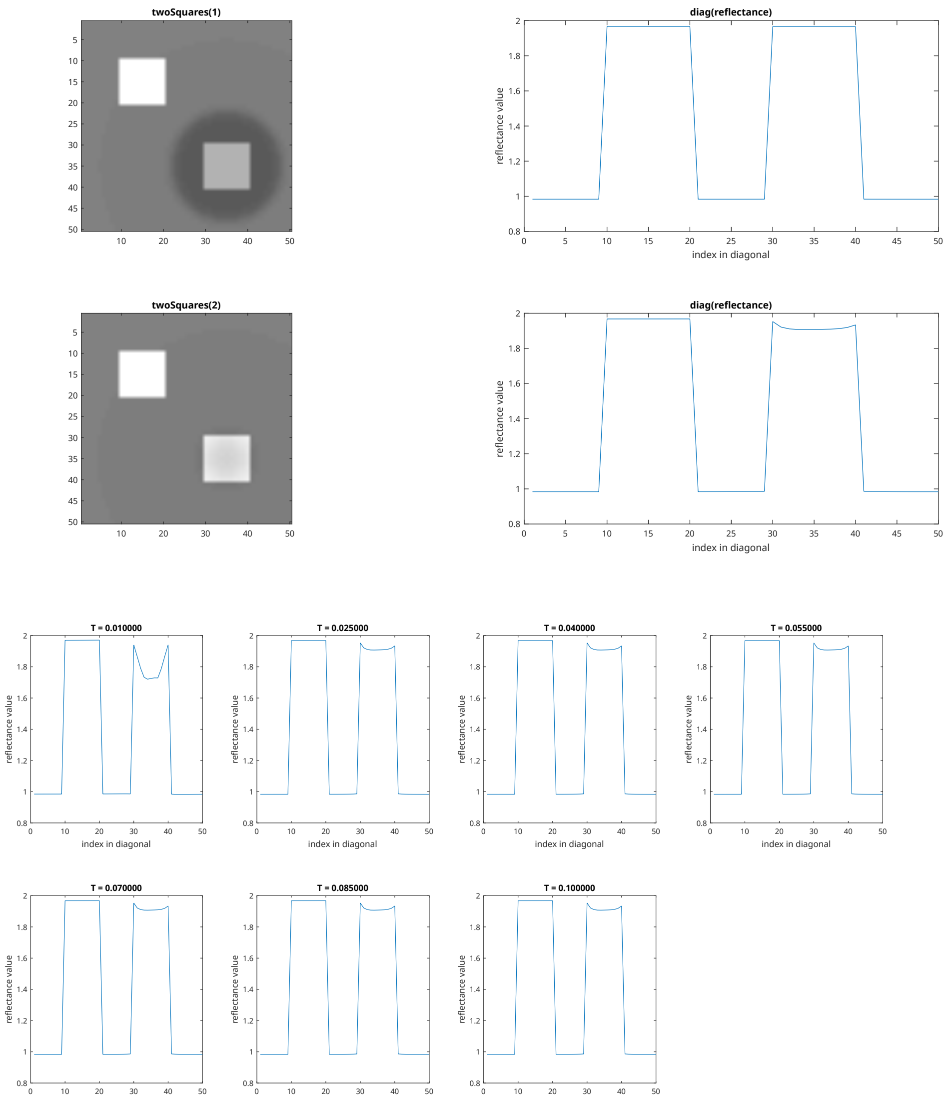
עבור הגירוי הנוצר עם `twoSquares(1)` וערך סף $T = 0.07$ האלגוריתם מחזיר בדיוק את אותו ערך reflectance. עבור הגירוי השני האלגוריתם מחזיר ערך החזרה משתנה בתוך הריבוע המוצל - כלומר מחזיר שהריבוע המוצל כהה יותר במרכזו ובהיר יותר בפריפריה - בהתאמה לצל העמום המוטל עליו.

נזכור כי ההנחות עליהן האלג' מתבסס הן:

- שינוי בסיגנל הוא שינוי במקדם ההחזרה או בתאורה, אך לא בשניהם
- שינוי חד בתמונה הוא שינוי במקדם ההחזרה ואילו שינוי הדרגתי הוא שינוי בתאורה

ההנחה המופרת בגירוי השני היא ההנחה לפיה השינוי בסיגנל הוא רק באחד מהגורמים, ולא בשניהם. במקרה זה, הריבוע התחתון מורכב מסיגנל משתנה הן בגלל הצל המוטל עליו והן בגלל ההפרדה שלו מהרקע. שני הגורמים משתנים במקביל ואילו האלגוריתם שמתבסס על הנחה (1) משווה את ה(שינוי ב)סיגנל לערך הסף כדי להפריד כל שינוי לאחד משני הגורמים האפשריים. כאשר שני הגורמים לא ברי הפרדה, השוואה כזו לסף נאיבי אינה יעילה ובטח שלא יכולה לחלץ את מידת השינוי בשני הגורמים ביחד.

שינוי ערכי T לא עוזר, כפי שאפשר לראות באיור ובהתאם לנדון למעלה.



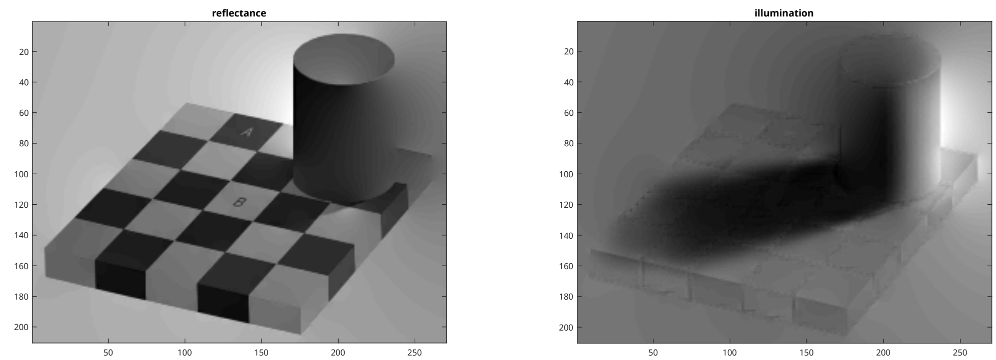
איור 4: שאלה 9 ־ על גירוי המקיים את ההנחות האלגוריתם מבצע בצורה מושלמת (עם ערך הסף הנבחר). עבור הגירוי השני, אף ערך סף לא נותן תוצאה מושלמת (כפי שאפשר לראות בגרפים התחתונים)

שאלה 10

הנקודות אכן בגוון זהה: $A=B=0.4196$. אחרי הרצת האלגוריתם, בתמונת מקדמי ההחזרה המתקבלת מתקיים ש־$R\left(A\right)=0.566<$ $0.876=R\left(B\right)$, בהתאמה לחוויית האשליה. הנקודה A נתפסת כאפורה כהה יחסית, בעוד ש־B נתפסת כלבנה.

על לוח השחמט האלגוריתם מחזיר תוצאות לא רעות, אם כי לא מושלמות. ניתן לראות שהריבועים השחורים בצל כהים יותר מאלה מחוץ לצל ובאופן דומה עבור ריבועים בהירים, אע"פ שהאנחנו תופסים אותם כבעלי צבע (מקדם החזרה) זהה. גם הגליל מקבל ערכי מקדם החזרה שונים בחלקיו השונים בשונה מהתפיסה.

גם כאן מופרת הנחת השינוי הבודד ־ הצל מוטל על אזורים בהם ישנו גם שינוי בצבע המשבצת, ולכן שינוי ערך הסף לא יפתור את טעות האלגוריתם בחלק זה. בנוסף, מופרת גם ההנחה השנייה לפיה שינוי חד בסיגנל הוא כתוצאה משינוי במקדם ההחזרה ־ למשל בחלקו העליון של הגליל קיים שינוי חד שנובע משינוי בתאורה ולא ב־ρ. באופן דומה גם בחלקו הימני־תחתון של הגליל יש שינוי קטן בסיגנל שנובע משינוי ב־ρ ולא משינוי בתאורה. גם כאן, שינוי ערך הסף לא צפוי לשפר את התוצאה מסיבות דומות לאלו שהוצגו בשאלה הקודמת.

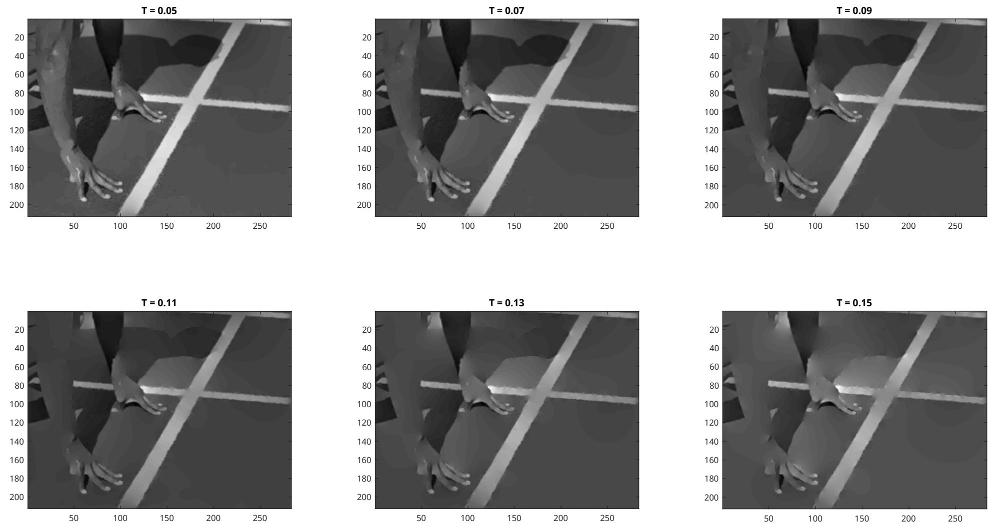


איור 5: שאלה 10 ־ על לוח השחמט הנחות האלגוריתם מתקיימות בצורה לא רעה ולכן גם ביצועי האלגוריתם בהתאם. על הגליל התוצאות בינוניות בגלל הפרת ההנחות.

שאלה 11

האלגוריתם נכשל בהסרת הצל. כפי שנדון בסעיפים הקודמים, ההנחה לפי שינוי חזק בסיגנל הוא תוצאה של שינוי מקדם ההחזרה אינה מתקיימת. לפיכך, הגדלת T תוביל לסיווג של שינויים חדים יותר כנובעים משינויי תאורה "תעלים" את הצל. עם זאת, מכיוון שהשינוי בסיגנל באזור קצוות הצל דומה לשינוי בקצוות היד, ולכן גם ההבדל בין היד למשטח (הרקורטן?) תסווג כשינוי בתאורה, וגם זו תוצאה שגוייה.

רמזים שאפשר להוסיף לאלגוריתם הם "מסכה" במימדים זהים לתמונה המקורית, עם ערכים $\{1,\ldots,k\}$ עבור קבוע $k \in \mathbb{N}$. תמונה זו תציין לאלגוריתם שבמקומות בהם הערך במסכה זהה, החומר בתמונה המקורית זהה. כך, כאשר נבדק תנאי ה־threshold בקורדינטה כלשהי, נפסול סיווג שינוי בסיגנל כשינוי בתאורה אם הערכים במסכה בקורדינטה זו זהים. באופן דומה, אפשר להזין מסכה דומה המתארת אזורים עם תאורה דומה ושינוי אנלוגי לזה שהוצג קודם בתנאי הסף.



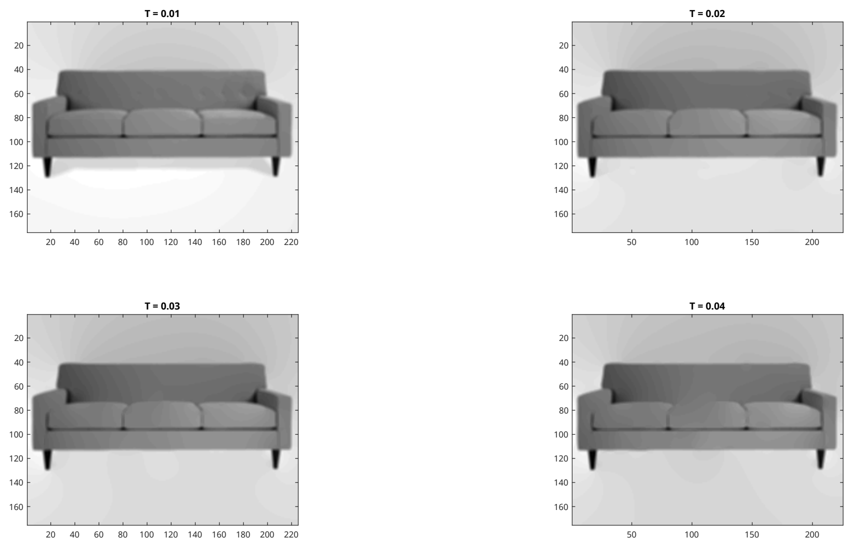
איור 6: שאלה 11 ־ כישלון בהסרת הצל ־ כאשר מגדילים את ערך הסף הצל נבלע במשטח, אך גם יד האצן.

שאלה 12

קל לראות שהאלגוריתם מסיר את הצל בהצלחה, עבור ערכי T לא קטנים מדי. עבור ערכי הסף הגדולים יותר, גם השינויים בתאורה בגב הספה (איפה שהכפתורים) נעלמים, ואנחנו מקבלים תמונה כמעט מושלמת של מקדמי ההחזרה כפי שהם נתפסים.

בתמונה זו ההנחות של האלגוריתם מתקיימות בצורה טובה: שינויים חדים בסיגנל נובעים משינוי בחומר ושינויים חלשים נובעים משינוי בתאורה.

ההבדל האיכותי בין תמונה זו לבין התמונות הקודמות, לדעתנו, הוא הפשטות של התמונה ־ הגיאומטריה של האובייקט המוצג היא פשוטה (משטחים ארוכים וישרים, אין רקע, אין עצמים עגולים המחזירים אור בצורה משתנה לאורך פני שטחם וכו') וכן התאורה היא אחידה ונראית כמוטלת ממקור יחיד מגבוה. לעומתה, בתמונה הקודמת יש הרבה פרטים ושינויים חזקים בסיגנל שנובעים מתאורה חזקה ומשטחים מורכבים (הידיים) המחזירים את האור בצורה שונה באופן דרמטי בחלקיהם השונים.



איור 7: שאלה 12 - אין הרבה הבדל בין ערכי T השונים (למעט אולי הקטן ביותר בו האלגוריתם לא ממש עובד) - כולם מצליחים במידה סבירה.

חלק III

קוד

```
function [Ix, Iy] = ImageDerivatives(I)
%IMAGEDERIVATIVES Returns the image x,y derivatives

    % convolution parameter
    MODE = 'full';
    % preallocate empty matrices
    Ix = zeros(size(I));
    Iy = zeros(size(I));
    % compute the derivatives
    Dx = conv2(I, Kx, MODE);
    Dy = conv2(I, Ky, MODE);
    % discard last column/row and zero out borders
    Ix(2:(end-1), 2:(end-1)) = Dx(2:(end-1), 2:(end-2));
    Iy(2:(end-1), 2:(end-1)) = Dy(2:(end-2), 2:(end-1));

end
```

Published with MATLAB® R2015a

```
function L = Deriv2Laplace(Ix, Iy)
%DERIV2LAPLACE Given the derivatives, computes the laplacian

    MODE = 'same';
    L = conv2(Ix, Kx, MODE) + conv2(Iy, Ky, MODE);

end
```

```matlab
function [R, L] = do_retinex(I, T)
%DO_RETINEX run the retinex algorithm with a given image I and threshold T

    % take the log of I and compute its derivative
    logI = log(I);
    [Ix, Iy] = ImageDerivatives(logI);
    % zero out values where norm < T
    normI = (Ix.^2 + Iy.^2).^0.5;
    smallerThanT = normI < T;
    Ix(smallerThanT) = 0;
    Iy(smallerThanT) = 0;
    % compute the laplacian and convolve with the inverse laplacian kernel
    L = Deriv2Laplace(Ix, Iy);
    Ka = invDel2(size(I));
    logR = conv2(L, Ka, 'same');
    % get the `real' reflectance and illumination
    R = exp(logR);
    L = I ./ R;

end
```

Contents

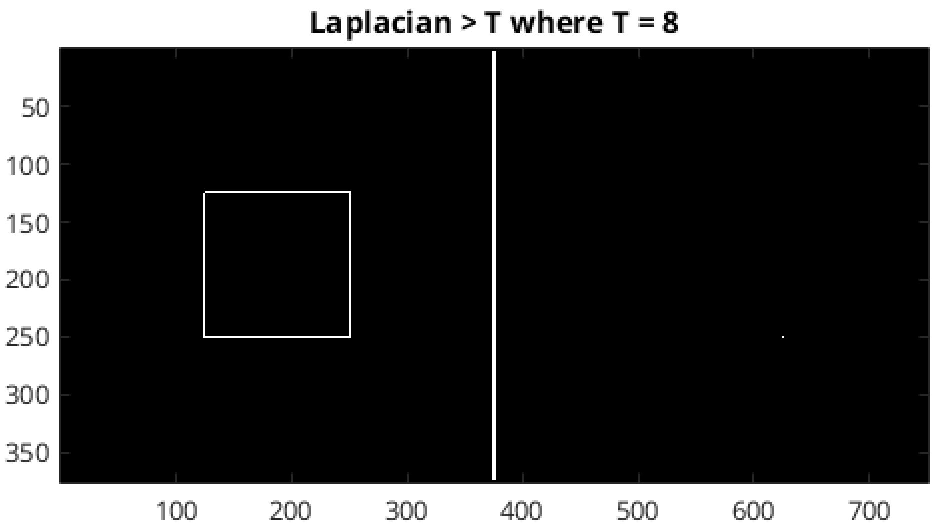
constants

```
Q4_IMAGE = 'simul_cont_squares.tif';
Q5_IMAGE = 'cross.tif';
Q6_IMAGE = 'kofka_ring.tif';

Q10_DATA = 'checkerShadow';
Q11_DATA = 'runner';
Q12_DATA = 'couch';
```

question 4

```
figure();
% read the image & compute laplacian (absolute value)
I = double(imread(Q4_IMAGE));
[m, n] = size(I);
[Ix, Iy] = ImageDerivatives(I);
L = abs(Deriv2Laplace(Ix, Iy));

% define the left/right parts of the image (exclude border ~ 10 pixels
%                                           in the middle)
leftL = L(1:end, 1:(n/2-5));
rightL = L(1:end, (n/2+5):end);

epsilon = 10;
T_space = 1:10;

for T = T_space
    if (length(find(leftL > T)) > epsilon) && ...
       (length(find(rightL > T)) <= epsilon)
        fprintf('Found T = %u\n', T);
        break;  % the result is T = 8
    end
end
fprintf('Number of pixels > T in the left image = %u\n', ...
        length(find(leftL > T)));  % == 1004
fprintf('Number of pixels > T in the right image = %u\n', ...
        length(find(rightL > T))); % == 4
show(L > T);
title(['Laplacian > T where T = ', num2str(T)]);
```

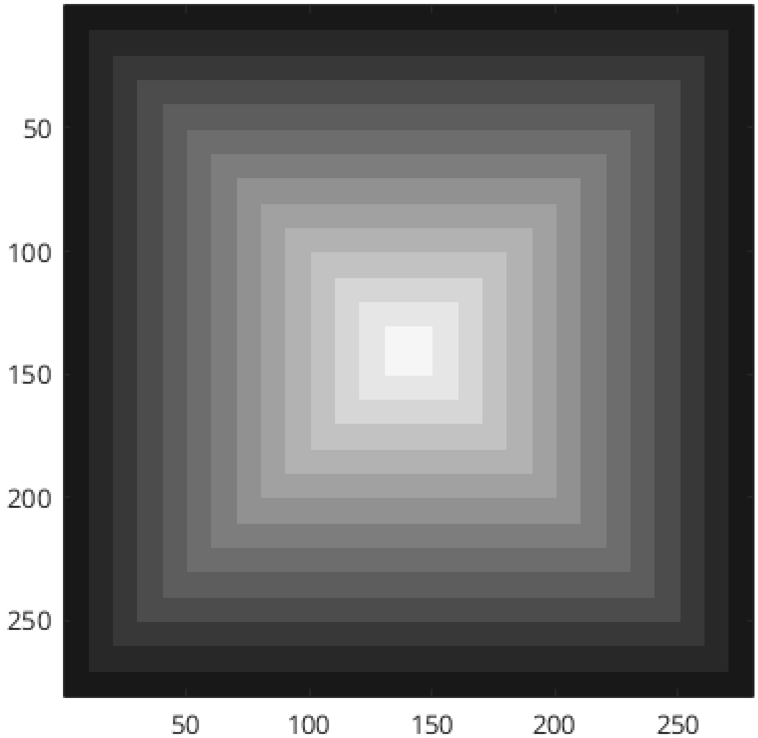
```
Found T = 8
Number of pixels > T in the left image = 1004
Number of pixels > T in the right image = 4
```

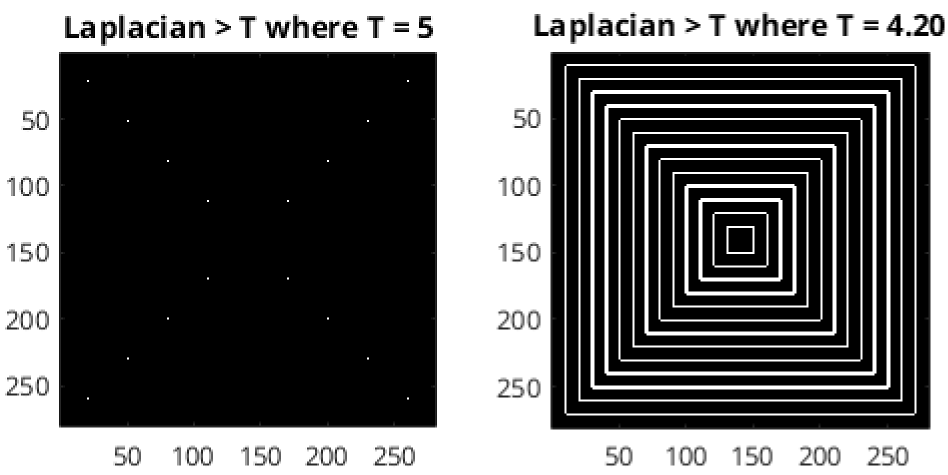


question 5

```
figure();
I = double(imread(Q5_IMAGE));
show(I, [0 255]);
[Ix, Iy] = ImageDerivatives(I);
L = abs(Deriv2Laplace(Ix, Iy));
Diag = logical(eye(length(L)));
Diag = Diag | rot90(Diag);
noDiag = not(Diag);
T_space = 1:10;
epsilon = 0.8;

figure();
for T = T_space
    if (~isempty(find(L(Diag) > T, 1))) && ...
         isempty(find(L(noDiag) > T, 1))
        fprintf('Found T = %u\n', T);
        break;
    end
end
% T = 0.017;

% show that this threshold is tight:
subplot(1, 2, 1);
show(L > T);
title(sprintf('Laplacian > T where T = %u', T));
subplot(1, 2, 2);
show(L > (T-epsilon));
title(sprintf('Laplacian > T where T = %.2f', T-epsilon));
% number of pixels > T on every part of the image
fprintf('Number of pixels > T on diagonal = %u\n', ...
    length(find(L(Diag) > T)));  % == 52
fprintf('Number of pixels > T NOT on diagonal = %u\n', ...
    length(find(L(noDiag) > T)));  % == 0
```

```
Found T = 5
Number of pixels > T on diagonal = 52
Number of pixels > T NOT on diagonal = 0
```

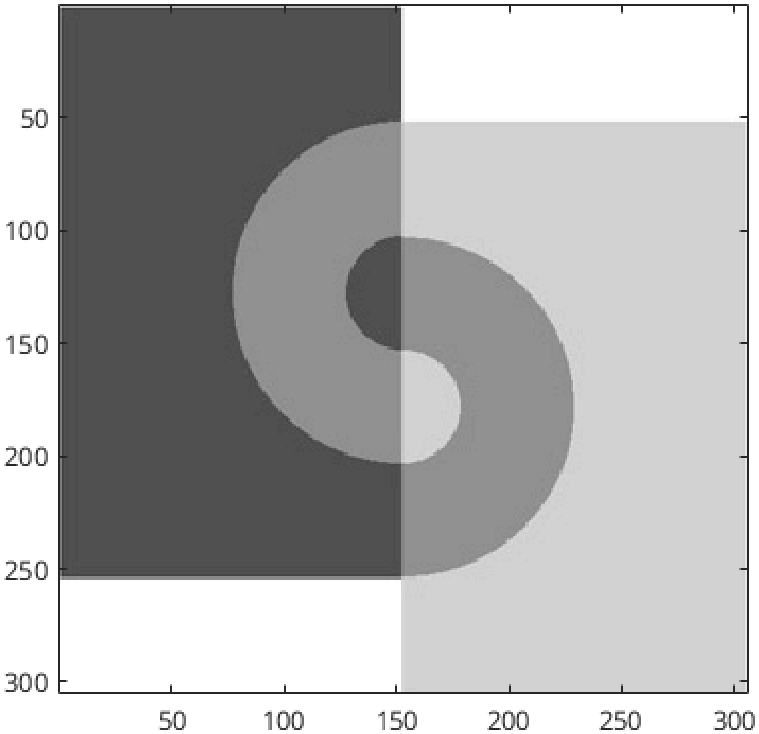


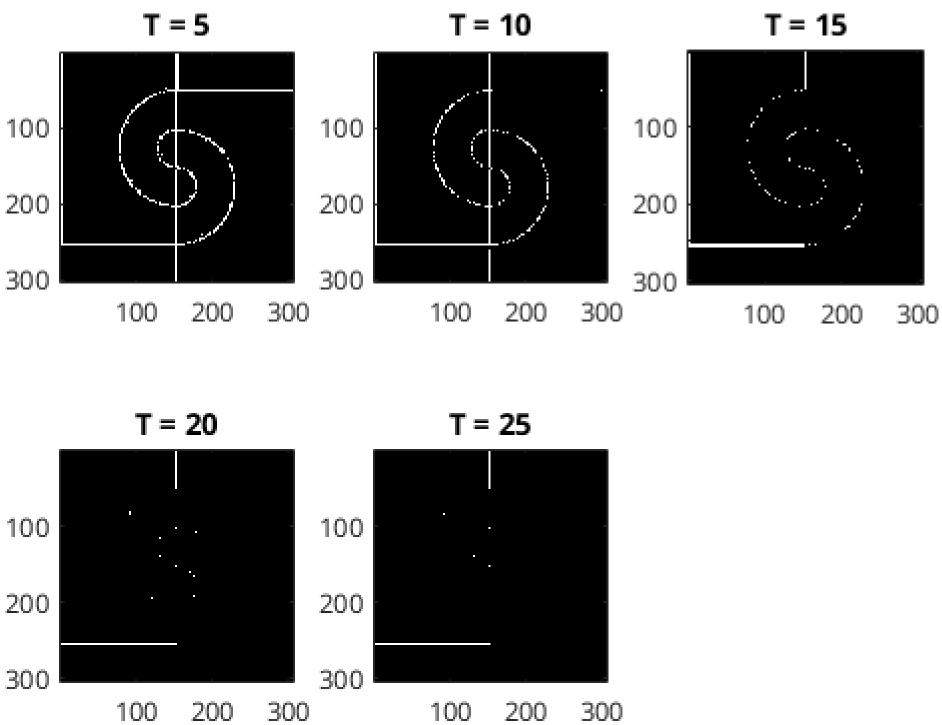


question 6

```
figure();
I = double(imread(Q6_IMAGE));
```

```
show(I, [0 255]);
[Ix, Iy] = ImageDerivatives(I);
L = abs(Deriv2Laplace(Ix, Iy));

figure();
T_space = 5:5:25;
k = numel(T_space);
rows = 2;
cols = ceil(k/2);

for i = 1:k
    subplot(rows, cols, i);
    T = T_space(i);
    show(L > T);
    title(sprintf('T = %u', T));
end
```



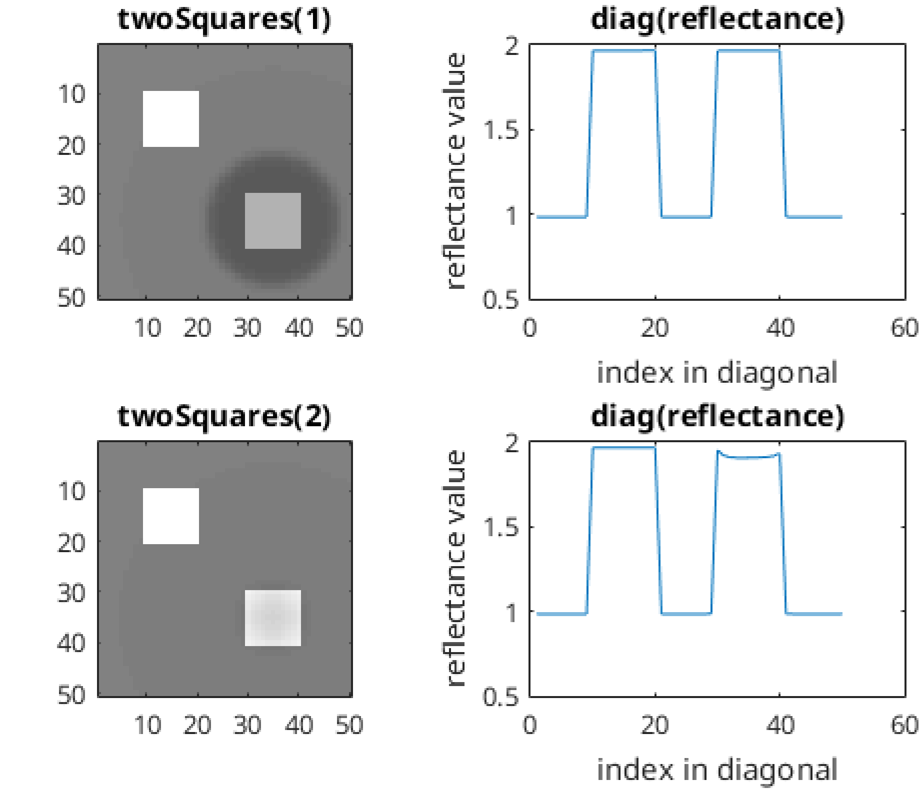


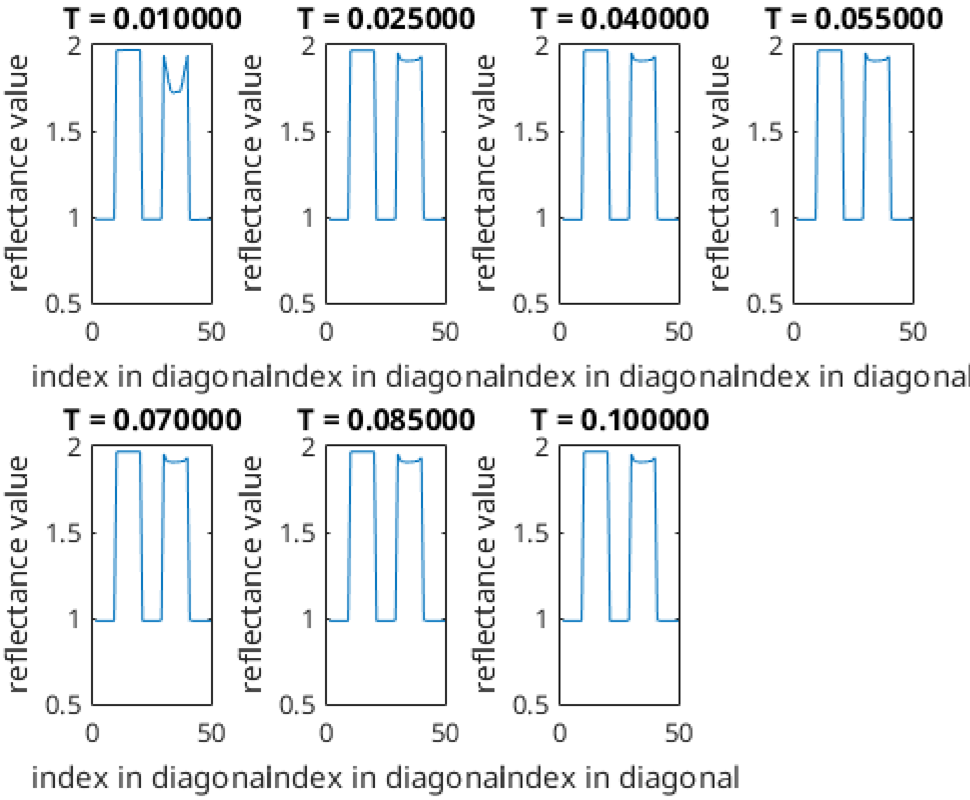
question 9

generate & show the stimuli

```
figure();
I1 = twoSquares(1);
I2 = twoSquares(2);

subplot(2, 2, 1);
show(I1, [0 2]);
```

```
title('twoSquares(1)');
subplot(2, 2, 3);
show(I2, [0 2]);
title('twoSquares(2)');

% run retinex
T = 0.07;
[R1, L1] = do_retinex(I1, T);
subplot(2, 2, 2);
plot(diag(R1));
title('diag(reflectance)');
xlabel('index in diagonal');
ylabel('reflectance value');

[R2, L2] = do_retinex(I2, T);
subplot(2, 2, 4);
plot(diag(R2));
title('diag(reflectance)');
xlabel('index in diagonal');
ylabel('reflectance value');

% check if changing T helps

figure();
T_space = 0.01:0.015:0.1;
k = numel(T_space);

for i = 1:k
    T = T_space(i);
    [R, ~] = do_retinex(I2, T);
    subplot(2, ceil(k/2), i);
    plot(diag(R));
    title(sprintf('T = %f', T));
    xlabel('index in diagonal');
    ylabel('reflectance value');
end

% it does not.
```



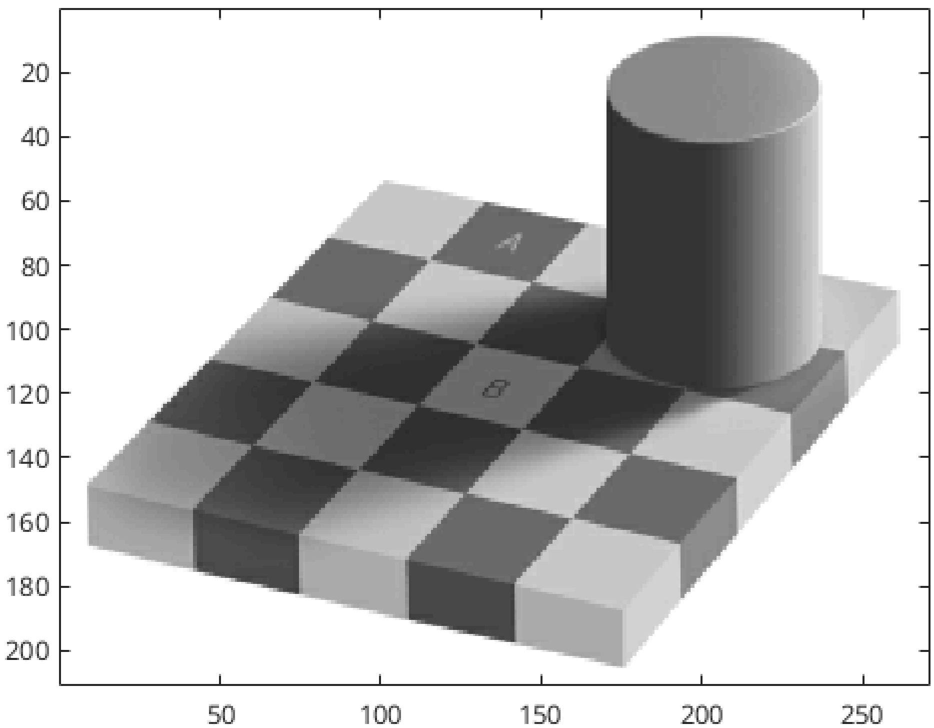


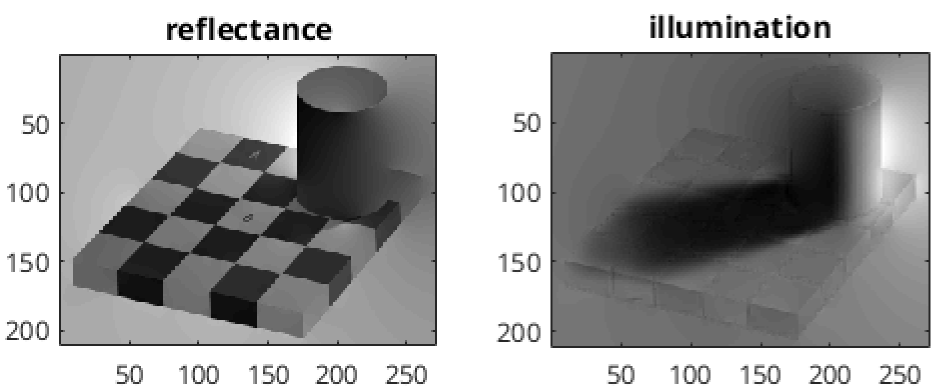
qeustion 10

```
figure();
checker = load(Q10_DATA);
show(checker.im1, [0 1]);
x1 = checker.x1; x2 = checker.x2;
y1 = checker.y1; y2 = checker.y2;
fprintf('A = %f \t B = %f\n', checker.im1(y1,x1), checker.im1(y2,x2));
% indeed, A = B = 0.419608
T = 0.07;
```

```
[R, L] = do_retinex(checker.im1, T);
fprintf('R(A) = %f \t R(B) = %f\n', R(y1,x1), R(y2,x2));
% as perceived: R(A) = 0.566 < 0.876 = R(B)
figure();
subplot(1, 2, 1);
show(R);
title('reflectance');
subplot(1, 2, 2);
show(L);
title('illumination');
```

```
A = 0.419608     B = 0.419608
R(A) = 0.566459          R(B) = 0.876770
```

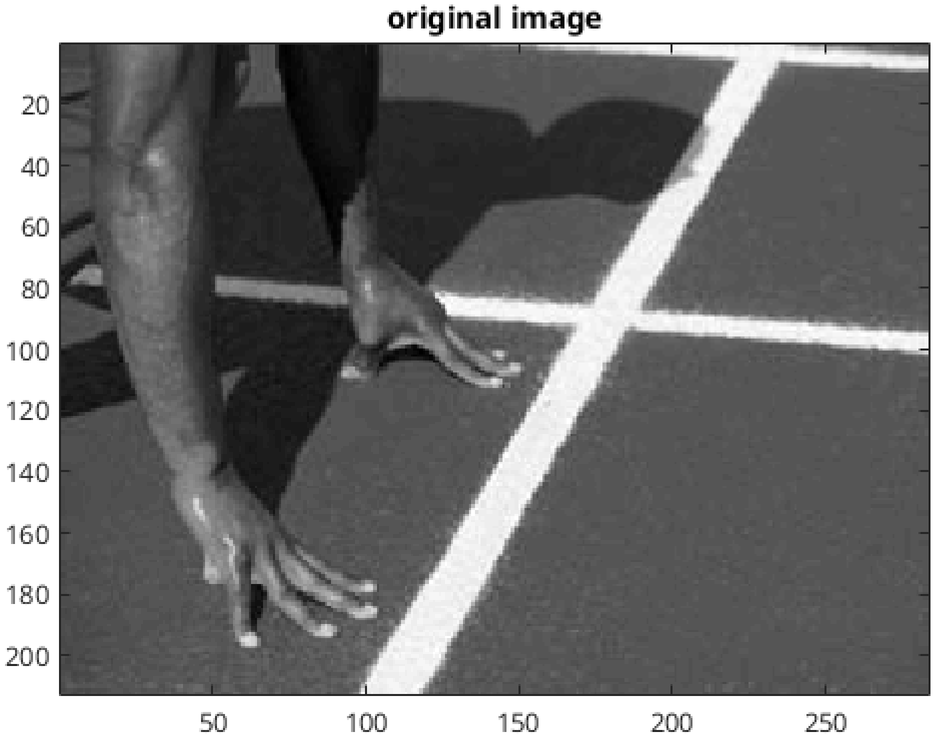


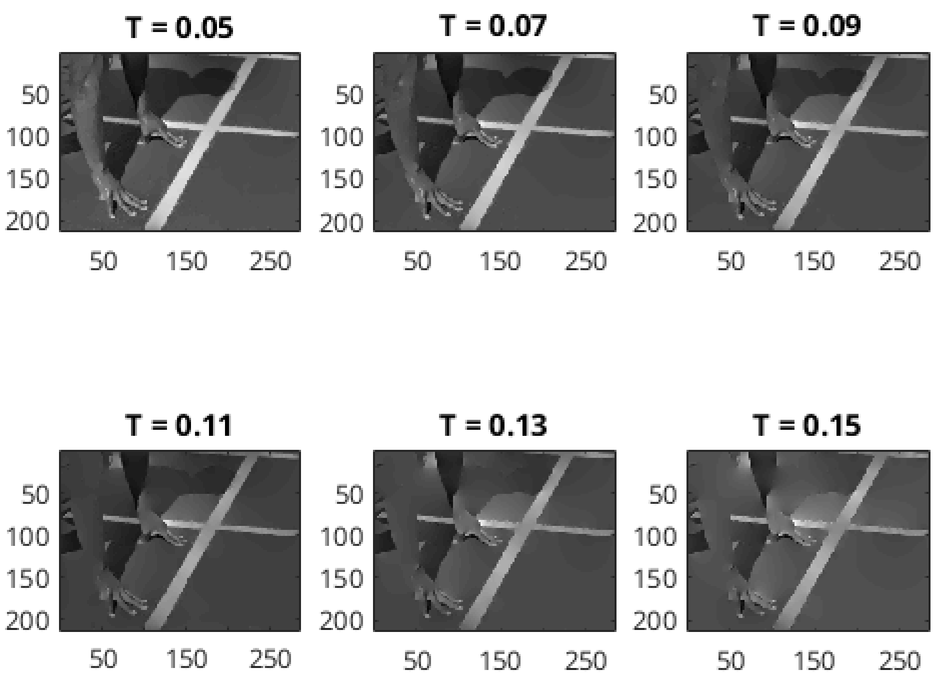


question 11

```
figure();
runner = load(Q11_DATA);
show(runner.im1);
title('original image');
T_space = 0.05:0.02:0.15;
k = numel(T_space);

figure();
```

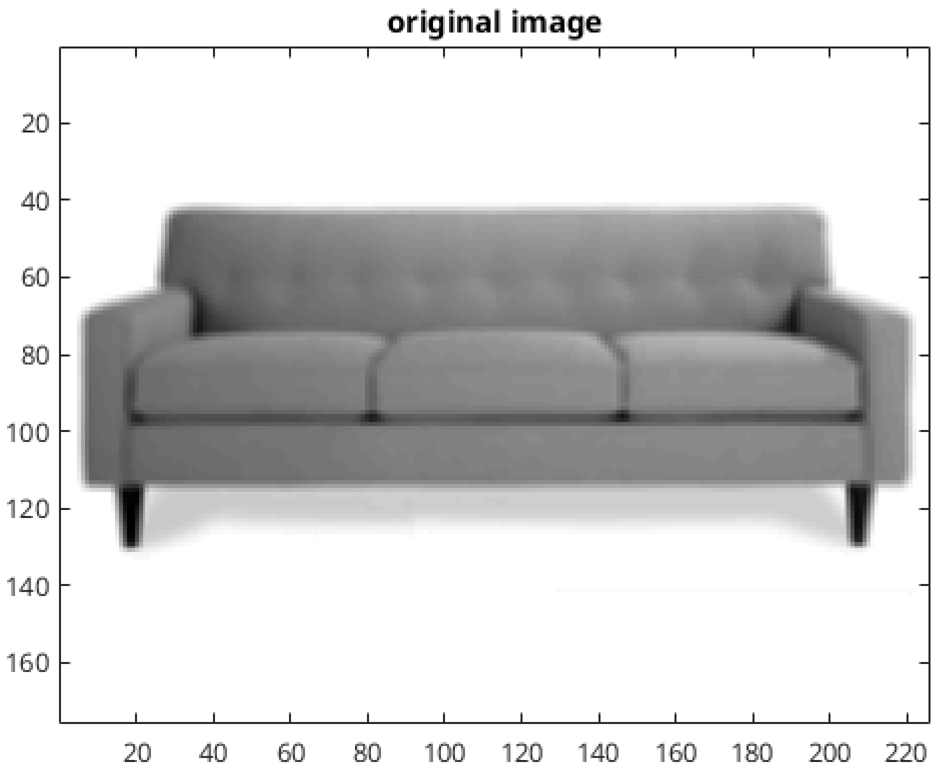
```
for i = 1:k
    subplot(2, ceil(k/2), i);
    T = T_space(i);
    [R, L] = do_retinex(runner.im1, T);
    show(R);
    title(['T = ' num2str(T)]);
end
```

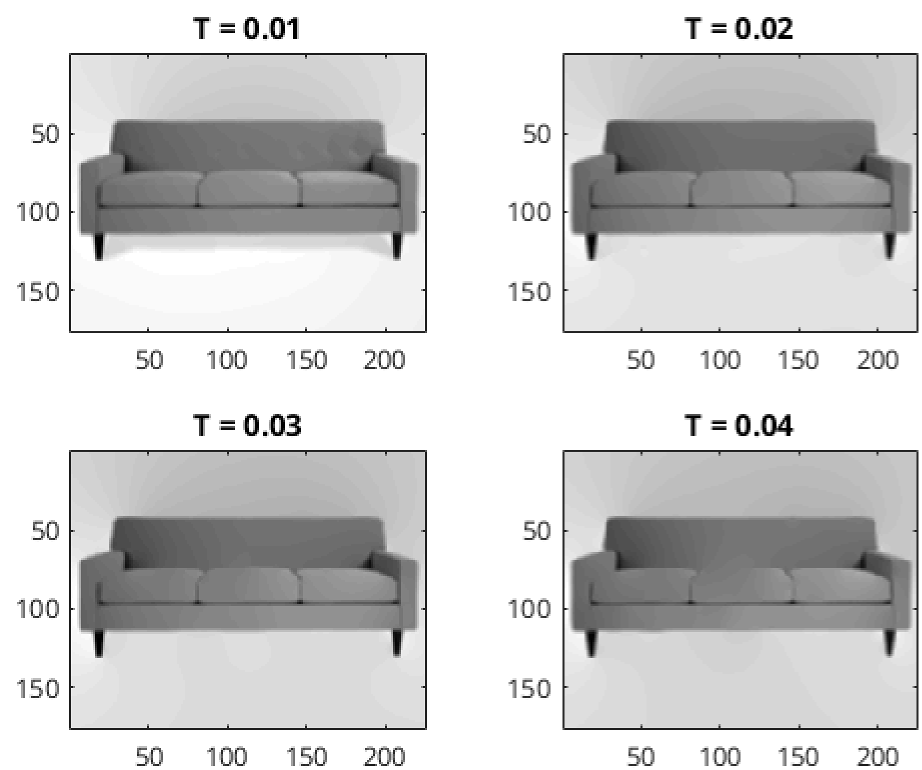




question 12

```
figure();
couch = load(Q12_DATA);
show(couch.im1);
title('original image');
T_space = 0.01:0.01:0.04;
k = numel(T_space);

figure();
for i = 1:k
    subplot(2, ceil(k/2), i);
    T = T_space(i);
    [R, L] = do_retinex(couch.im1, T);
    show(R);
    title(['T = ' num2str(T)]);
end
```





Published with MATLAB® R2015a