

עיבוד תמונה

תוכן עניינים

3	שיעור ראשון
3	כיצד ניצג תמונה
3	Intensity Transformation
6	היסטוגרמָה – Histogram
8	קונוטיזציה – Quantization
10	קונבולוציה וקורולציה – Convolution & Correlation
10	תכונות הקונבולוציה
11	הבדל בין קונבולוציה לקורולציה
11	Padding – Output size of convolution
13	שימושים
14	זמן ריצה
14	הגדלה (פילטר נתן להפרדה)
14	על מנת לדעת אם פילטר K נתן להפרדה נסתכל על פירוק ה-SVD שלו, כלומר שקיים ערך יחיד ($-singular$) בודד s שאינו אפס. במידה וכן $s \cdot u \cdot s^{-1} \cdot v^T$ הם הפילטרים.
15	זיהוי קווי מתאר – Edge Detection
15	הקדמה
16	המקרה הכללי
17	Guassian filter
17	Laplacian Of Guassian
19	Sobel's Algorithms
19	Canny's Algorithm
20	Hough Transform
23	מעגלים
25	מודל המצלמה
27	עדשות
33	מציאת הפרמטרים המתאימים לטרנספורמציה
36	התאמות של פיצ'רים
36	פיצ'רים טובים
37	Harris Corner Detector
40	האלגוריתם
40	תכונות של אלגוריתם Harris Detector
41	אלגוריתם Scale Invariant
42	Key Point Localization
43	התאמת של פיצ'רים
44	SIFT – Scale Invariant Feature Transform

44	Orientation Assignment
45	Key-Points Descriptor
45	התאמת של פיצ'רים
47	יתרונות וחסרונות של RANSAC
48	Object Recognition
49	Optical Flow
49	הגדלה (זרימה אופטית).
49	זרימה אופטית היא הורימה הנראית של אובייקטים או משטחים.
50	נושאים חשובים שלא דיברתי עליהם
60	Bibliography

שיעור ראשון

בתמונה דיגיטלית ישנו רכיבים שתפקידם לקלוט את קרני האור, הרכיבים האלה נקראים סנסורים והם בעצם מערכ דו-מימדי כאשר כל איבר במערך מכיל קולטן שבודק כמה פוטונים מתקבלים בה בטוחה זמן מסוים. כאשר אנו מצלמים תמונה אנו פותחים את המחשום וכל אחד מהסנסורים שומר את מספר הפוטונים שהתקבלו אצלו. ככל שייהו לנו יותר סנסורים התמונה תתפרק לייחדות יותר קטנות וכן יהיה לנו יותר סנסורים ליחידת שטח (או נפח) וכך נוכל ליצג את התמונה בצורה יותר טובה. אומנם חשוב לציין שיש לנו הגבלה של זיכרון וכן יש off-trade-off בין האיכות לזמן.

כיצד נציג תמונה

נוהג ליצג תמונה באמצעות מטריצה דו מימנית שנסמנה ב- I מלשון *Image*, כאשר $I_{x,y}$ יהיה הערך המופיע בשורה x תחת עמודה y . כל איבר (y, x) נקרא פיקסל. לעומת זאת נניח שכל סנסור במערך הסנסורים ימופה לפיקסל.

ישנו מספר פרמטרים נוספים המשפיעים מהותית על אופי התמונה:

1. כמה פיקסלים יש בתמונה – הרזולוציה של התמונה.
2. לכל פיקסל יש טווח של ערכים אותו הוא קיבל, כלל שהטווח זה יהיה יותר גדול ככל ליצג הרבה יותר גוונים של הצבע אותו אנו מודדים. טווח ערכים זה נקרא קוונט של תמונה, כאשר אנו מחליטים מהו יהיה אנו מבצעים קוונטיציה. בפועל אנו קובעים את היחידה היכי קטנה שנוכל ליצג, כלומר הקוונט הוא מספר הפיקסלים + הטווח של כל פיקסל.

בדרכ כל אנו נציג תמונה צבעונית באמצעות שלושה מערכ דו-מימדיים שככל אחד מייצג צבע יסודי כלשהו, לדוגמה: אדום, ירוק וכחול. היות וכל צבע אחר יכול להיות מייצג ע"י שילוב שלהם, נוכל ליצג את כל הצבעים.

Intensity Transformation

זהו סוג של טרנספורמציות בווא המיפוי נעשה באמצעות קלט יחיד לפולט יחיד. אנחנו נדבר על מיפוי של עצומות של דרגות של אפור. נראה מספר שימושים לפונקציות כנ"ל:

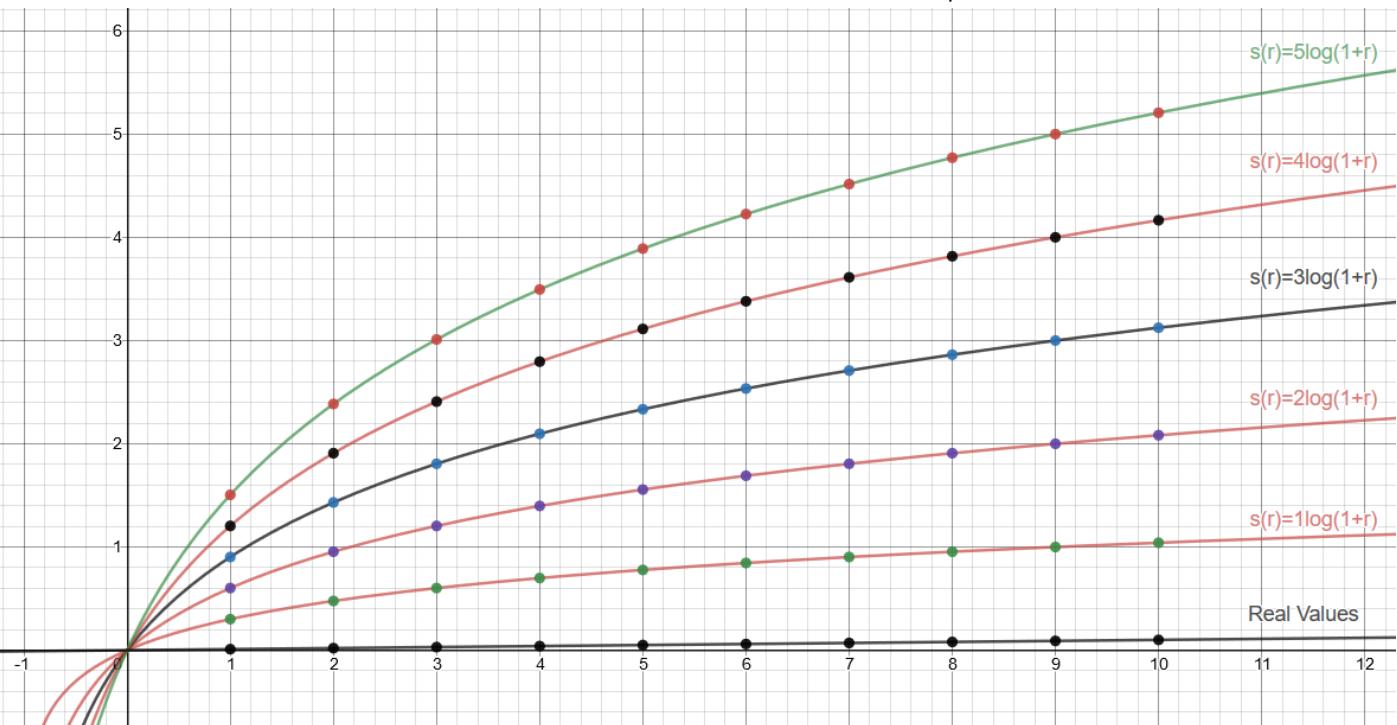
$$1. \text{ היפוך: } r = 1 - s(r)$$



$$2. \text{ טרנספורמציות לוגריתמיות: } r = c \cdot \log(1 + r)$$

המטרה של טרנספורמציה מהסוג הנ"ל היא למפות תמונות בהם הטווח קטן, לטווח של ערכים שמתפרש באופן "יותר אחד" אך בהתאם למה שהוא המקורי, לומר אם היונו נתונים אותו ערך

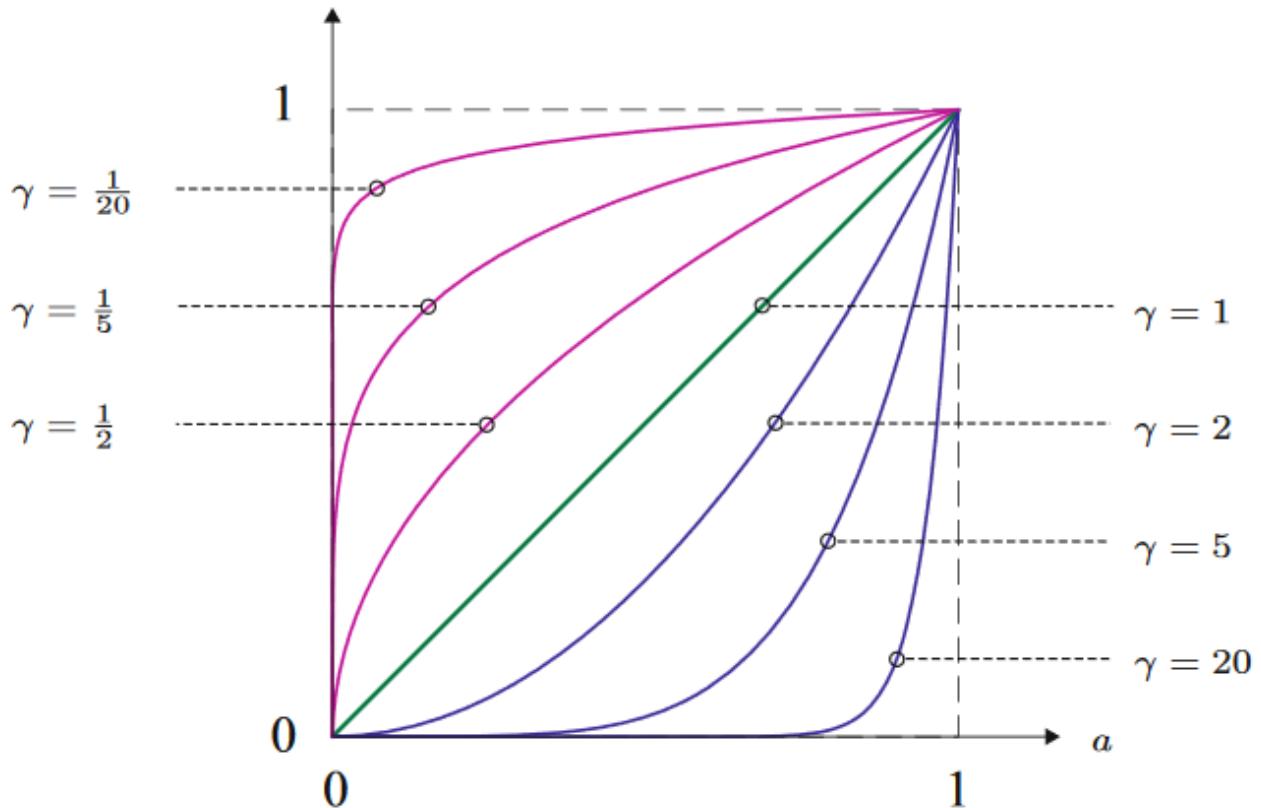
לכלם, זה היה הורס את התמונה. לדוגמה אם הערכים שאנו חnage מקבלים הם $\{1, 0.9, \dots, 0.2, 0.1\}$ נפעיל את הטרנספורמציה הבאה נקבל:



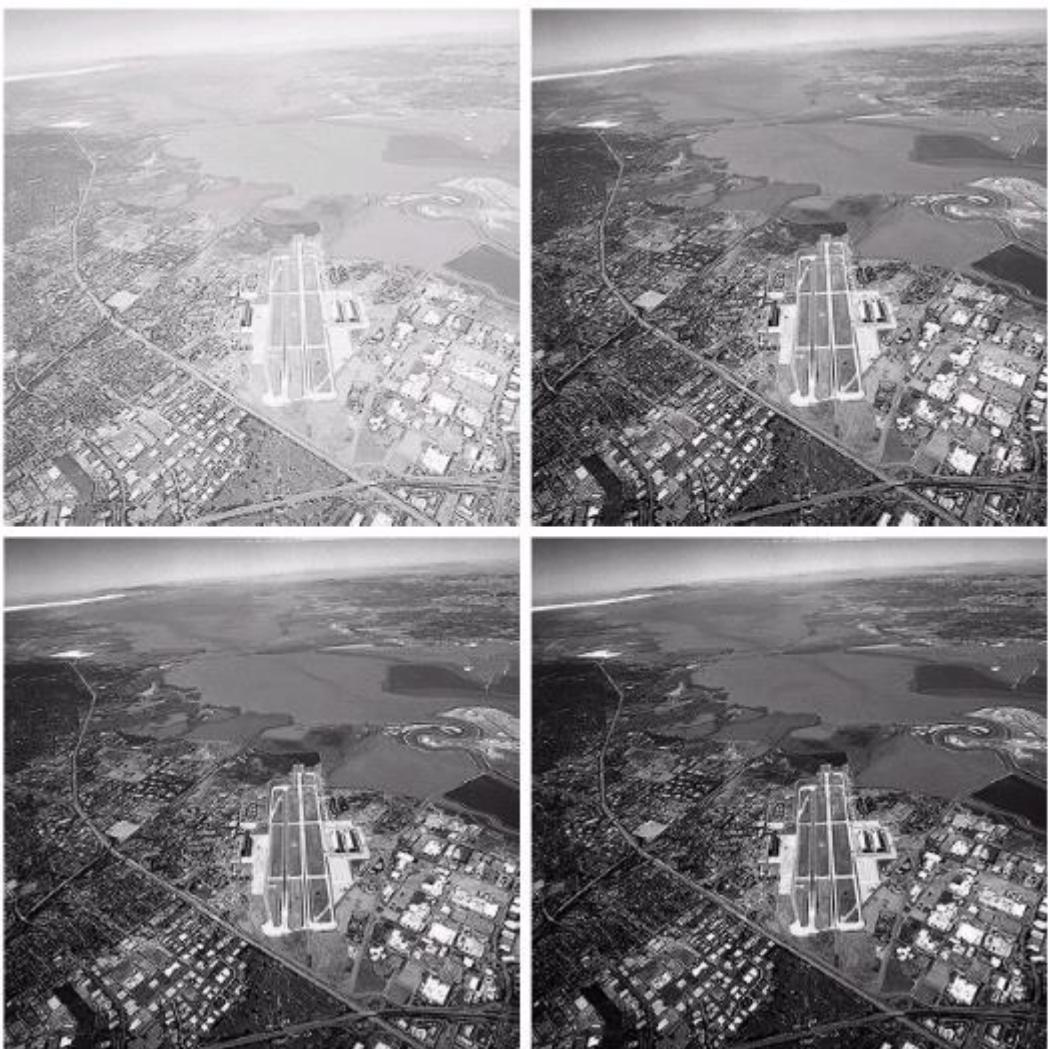
ואכן ניתן לראות פיזור יחסית יותר טוב.

3. ישנה הכללה של הטרנספורמציה הקודמת באופן הבא:

$$b = a^\gamma$$



על דוגמה אמיתית:

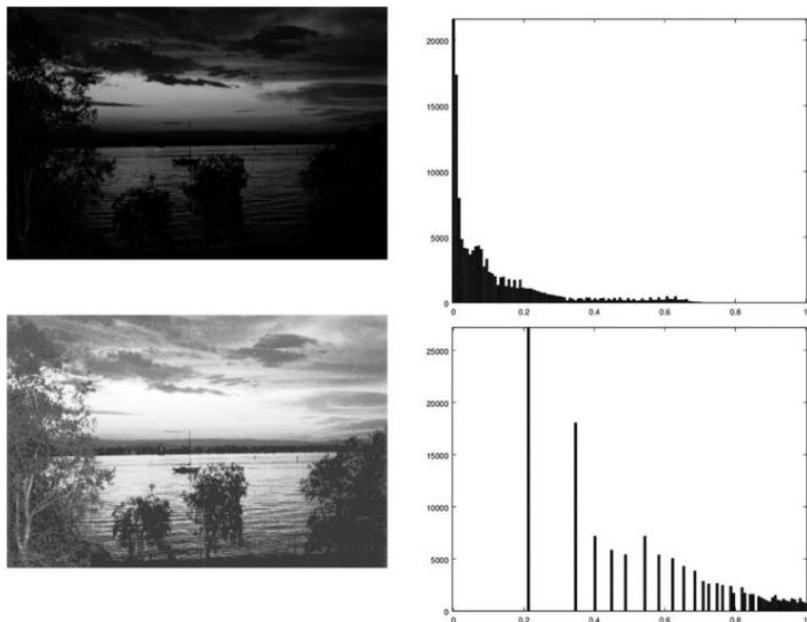


רואים כי התמונה הראשונה הכל בערך באותו גוון של אפור, כלומר השני בטוווי אפור אינו ממשמשותי אך קיים, אנחנו נרצה לתקן את זה, כלומר לפזר אותם על טווח יותר גדול, נפעיל טרנספורמציה גמא עבור $\gamma = 3,4,5$ משמאל לימין ומלמטה למעלה.

ה unin האנושית מבינה דגש על שינוי צבעים בהירים מאשר קהים, כלומר אנו נבחן יותר בשינויים בטוווי בין 0 ל-127 מאשר בין 128 ל-255. לכן אנחנו נרצה להגדיל את הטווח עבור ערכים בהירים ולהקטין את הטווח לערכים קהים שכן בכל מקרה אנחנו לא נשים לב אליהם.

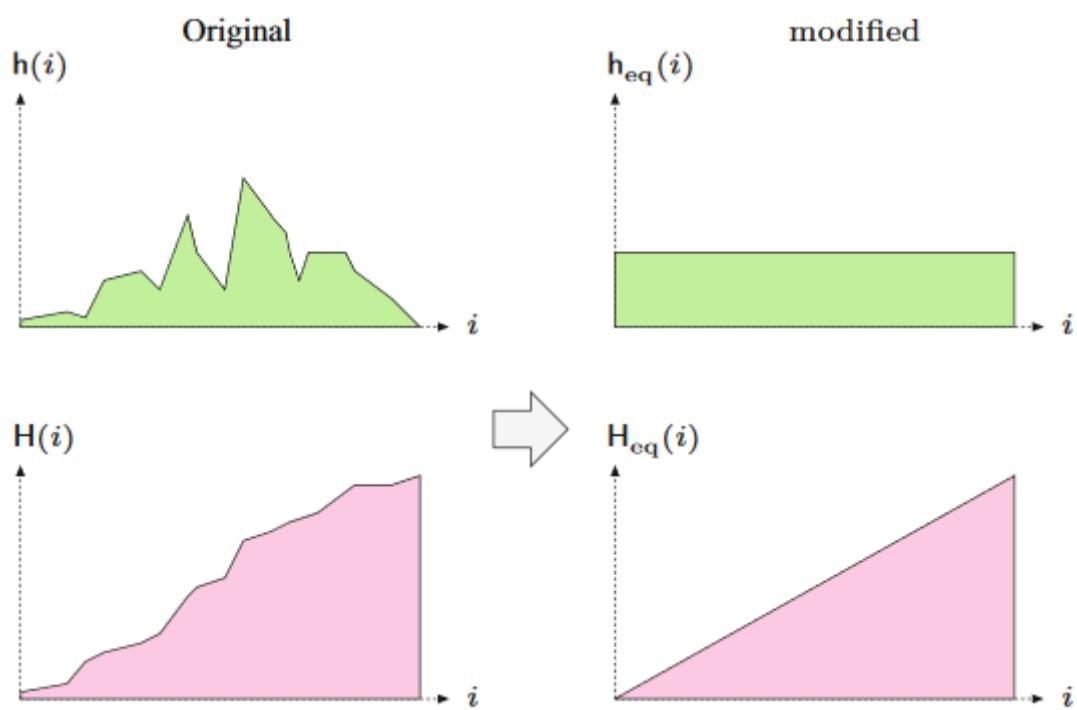
Histogram

ההיסטוגרמה של תמונה היא פונקציה בדידה שנוצרת מספרת כמה פיקסלים קיימים בגוון אפור מסוים. כאשר הפונקציה מנורמלת כך שהסכום של הערכים שלה שווה ל-1, ניתן להתייחס אליה כפונקציית צפיפות שmbטאת מה שכיחות של גוון אפור מסוים בתמונה. מנוקודת מבט זאת הערך של פיקסל בתמונה הינו משתנה מקרי שמקבל ערכים בהתאם לניסוי רנדומי כלשהו.



באופן אינטואיטיבי אנו נרצה שההיסטוגרמה תהיה אחידה¹, כלומר שמספר הפיקסלים יהיה שווה (כל הניתן) בין כל גווני האפור של התמונה. אולם לעיתים התמונה מתפזרת על טווח מצומצם בעיקר של גווני אפור, בשל חסיפה לא מספקת של התמונה. אנו נוכל להעшир את התמונה ע"י תהליך שנקרא שוויזון היסטוגרמות שפורס את ההיסטוגרמה בצורה שווה ככל הניתן בין גוונים האפור. כדי להמchio אינטואיציה זאת נסתכל על התמונה משמאל המתארת שקעה של השימוש.

באופן כללי הסכמה של שוויזון היסטוגרמות מתוארת באIOR הבא:



¹ הקורא השן יכול לשים לב שאכן האנלוגיה והאינטואיציה מגבילה להסתברות אחידה (במובן הרצוי).

באופן כללי נשים לב שהתוכן של המידע עובר באמצעות היחסים בין גווני האפור של התמונה ולא הערך האבסולוטי של הגוון אפור. דהיינו לגווני אפור אין משמעות מבחינה אבסולוטית אלא מבחינה יחסית בלבד. לאבחן זאת שני מסקנות:

1. ניתן לבצע שינוי של ההיסטוגרמה כך שהיא תתרטט בצורה אחת יותר באמצעות טרנספורמציה $[1 - L] \rightarrow [0, L]$: שתלויה רק בערכי עצמת הגוון האפור r .
2. הטרנספורמציה T חייבת להיות פונקציה מונוטונית במובן הבא:

$$r_2 \geq r_1 \implies T(r_2) \geq T(r_1)$$
3. שוב, להיות ומספר הפיקסלים שקטנים מערך אינטנסיטי r נשאר אותו דבר לגבי ה-CDF של שני הערכים $s = T(r)$ ושל r , שווים. דהיינו

$$F_{old}(r) = F_{new}(s) \iff \int_{g_{min}}^r f_{old}(x) dx = \int_0^s f_{new}(x) dx$$

היות ונחנו מתחשקים עם ערכים בדידים, נסמן ב- g_{min} את העוצמה המינימלית בתמונה המקורית ונקבל:

$$F_{old}(r) = F_{new}(s) \iff \sum_{x=g_{min}}^r f_{old}(x) = \sum_{x=0}^s f_{new}(x)$$

היות ונחנו שאנו רוצים את התפלגות אחידה נובע כי $1/G$ כאשר G הוא מספר הדרגות עצמה (אפור) שיש בתמונה, בדר"כ $256 = G$. לפיקר נקבל:

$$\sum_{x=g_{min}}^r f_{old}(x) = \sum_{x=0}^s \frac{1}{G} = (s+1)/G \iff s = \left\lceil G \cdot \sum_{x=g_{min}}^r f_{old}(x) - 1 \right\rceil$$

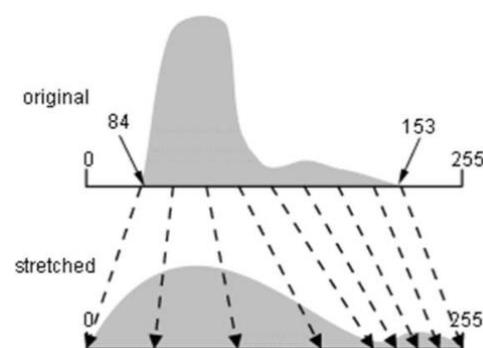
היות ונחנו מתחשקים עם ערכים שלמים נבצע מספר בדיקות על מנת להשתכנע שאכן הערך העליון הגווני. אם $r = g_{min}$ אז נקבל $1 - f_{old}(g_{min}) = G \cdot f_{old}(g_{min}) < s$ ועבור $1 < s < G - 1$ – וכן ניקח ערך עליון על מנת לקבל 0, באופן דומה אם $r = g_{max}$ אם $G - 1 < s < 1$ – וכן ניקח ערך עליון על מנת לקבל 0, באופן דומה אם $r = g_{max}$ אם $0 < s < 1$. כנדרש.

4. דרך נוספת לראות את זה היא כיוון שאנו יודעים שה-CDF של התפלגות אחידה בקטע $[a, b]$ היא $\frac{x-a}{b-a}$. לפיקר $F_{new}(x) = \frac{x}{G-1}$, ולכן $F_{old}(r) = \frac{s}{G-1} = \text{כלומר } (s+1)/G$.
5. היתרון של שוויון ההיסטוגמות על גבי מתיחה של ההיסטוגרמה, לדוגמה ע"י:

$$T(r) = \left\lceil \frac{r - g_{min}}{g_{max} - g_{min}} \cdot G + \frac{1}{2} \right\rceil$$

היא שוויון ההיסטוגמות נתון תוצאה בא הסבירות לכל גוון היא שווה ככל הנition, לעומת זאת במתיחה של ההיסטוגרמה מספר גוונים האפור נשאר זהה. ניתן לראות זאת בדוגמה הבאה: אם ישנים שני ערכים בלבד ניתן אך המתיחה תיקח את הערך הגבוה לבן והנמוך לשחור, לעומת זאת השווון יקח את הערך הנמוך ל-127 ואילו את הגבוה לשחור. מתיחה היא תמיד העתקה חח"ע ועל.

הערה: הערך פלוס חצי אומר לחת את הערך השלם הקרוב ביותר.



קווינטיציה - Quantization

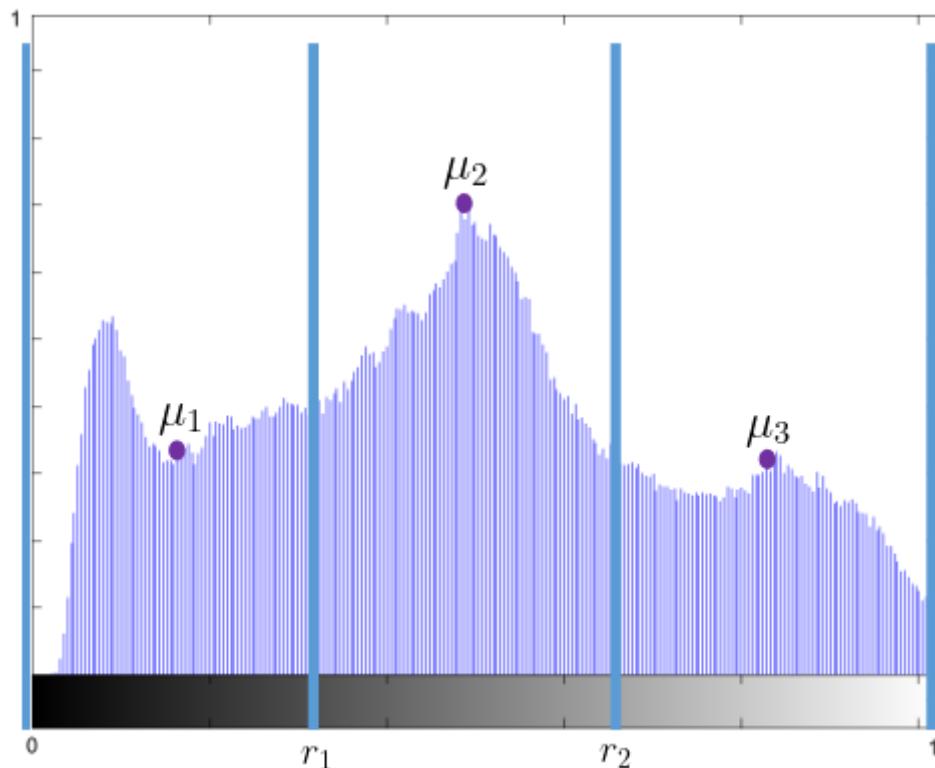
בהתבונת מטריצה שעריכה מעל הממשיים באינטרוול $[0,1]$, נרצה להשתמש ב- N צבעים בשביל לקרב את התמונה טוב ככל הניתן, כלומר נרצה למצוא נקודות r_0, r_1, \dots, r_N , כך שבכל קטע $I_i = [r_i, r_{i+1}]$ הטעות

$$\mathcal{E}_{I_i} := \mathbb{V}(r) = \mathbb{E}[(r - \mu_i)^2] = \int_{r_i}^{r_{i+1}} (r - \mu_i)^2 \cdot P(r) dr$$

כאשר μ_i יהיה הערך שמחלייר את כל הצבעים בקטע I_i ו- $P(r)$ תהיה ההסתברות לקבל את העוצמה r .
בנוסף נגדיר את הטעות הכלולת של התמונה אותה נרצה להקטין ככל הניתן להיות

$$\mathcal{E} := \sum_{i=0}^{N-1} \mathcal{E}_{I_i} = \sum_{i=0}^{N-1} \int_{r_i}^{r_{i+1}} (r - \mu_i)^2 \cdot P(r) dr$$

האיור הבא ממחיש את התפקידים של המשתנים והתהיליך אותו אנחנו מנסים לפרמל:



כעת נגזר ונקבל:

$$\frac{\partial}{\partial \mu_k} \sum_{k=0}^{N-1} \int_{r_k}^{r_{k+1}} (r - \mu_k)^2 \cdot P(r) dr = 2 \int_{r_k}^{r_{k+1}} (\mu_k - r) \cdot P(r) dr = 0 \quad (1)$$

$$\begin{aligned} \frac{\partial}{\partial r_k} \sum_{k=0}^{N-1} \int_{r_k}^{r_{k+1}} (r - \mu_k)^2 \cdot P(r) dr &= \frac{\partial}{\partial r_k} \left(\int_{r_{k-1}}^{r_k} (r - \mu_k)^2 \cdot P(r) dr + \int_{r_k}^{r_{k+1}} (r - \mu_k)^2 \cdot P(r) dr \right) \\ &= (r_k - \mu_{k-1})^2 \cdot P(r_k) - (r_k - \mu_k)^2 P(r_k) \\ &= P(r_k) \cdot [(r_k - \mu_{k-1})^2 - (r_k - \mu_k)^2] \\ &= P(r_k) [(r_k - \mu_{k-1} - (r_k - \mu_k))((r_k - \mu_{k-1} + r_k - \mu_k))] \\ &= P(r_k) [(2r_k - \mu_k - \mu_{k-1})] = 0 \end{aligned} \quad (2)$$

כאשר בשני המשוואות השווין הראשון נובע היות וכל שאר האלמנטים בסכום קבועים שכן אינם תלויים במשתנה הגזירה, וכך שגוררים אותם הם מתאפסים. בנוסף, היות והסכום סופי מותר לשנות את סדר הסכימה הגזירה והאנטגרציה. בפתרון (2) השווין השני נובע משני אבחנות פשוטות:

$$\frac{\partial}{\partial y} \int_{-\infty}^y f(x) dx = f(y), \quad \frac{\partial}{\partial y} \int_y^{\infty} f(x) dx = -f(y)$$

שאר השוויונים מתקיימים ע"י פישוטים אלגבריים פשוטים. ממשואה (1) מתקבלת אם ורק אם

$$\cdot \int_{r_k}^{r_{k+1}} (\mu_k - r) P(r) dr = 0 \iff \mu_k \int_{r_k}^{r_{k+1}} P(r) dr - \int_{r_k}^{r_{k+1}} r \cdot P(r) dr = 0 \iff \boxed{\mu_k = \frac{\int_{r_k}^{r_{k+1}} r \cdot P(r) dr}{\int_{r_k}^{r_{k+1}} P(r) dr}}$$

הקורא השנון יבחן כי $\mu_k = \mathbb{E}(r|I_k \in I_k)$.

במשואה (2) נשים לב כי לא ניתן כי $\mu_{k-1} = \mu_k$ כי אז יכולנו לאחד את I_{k-1} ו- I_k לקטע אחד ולקבל אותו פיתרון, דבר שלא מתיישב. לפיכך נקבל כי

$$2r_k - (\mu_{k-1} + \mu_k) = 0 \iff \boxed{r_k = \frac{\mu_{k-1} + \mu_k}{2}}$$

כנדרש.

משוואות אלו הינן משוואות סתומות ולכן אנו נצורך לחלק ביטויים סגורים באופן איטרטיבי, בו מתחילה חלקה אחת של העוצמות אפור וمعدכנים איטרטיבית לפי הנוסחאות. כזכור, כל התנאים לעיל הם הכרחיים אך לא מספיקים אפילו למינימוםлокלי.

ענין חשוב שההעמלנו ממנו הוא מקור האקרואיות של פונקציית ההסתברות, היות והפונקציה לא חלהה בדרך"כ, וכך נרצה לבצע החלקה של פונקציית ההסתברות ע"י קונבולוציה עם הגausיאן לדוגמה באופן הבא:

$$P_G(r) = \frac{1}{|\Omega|} \cdot \sum_{(i,j) \in \Omega} G(r - I[i, j], \sigma)$$

כאשר I היא מטריצת התמונה.

קונבולוציה וקורסילציה – Convolution & Correlation

קונבולוציה היא אופרטור מתמטי הפועל על שני פונקציות. פעולה הנקראת kongvolutzia את דרגת ה-overlap (חפיפה) של שני הפונקציות לכל ערך של $t \in (-\infty, +\infty)$, כאשר הקונובלוציה גבוהה יותר כאשר הפונקציות חופפות/מתלכדות/דומות יותר. כתה נפרמל זאת: בהינתן שני פונקציות f, g , נגידו

$$f * g(t) = \int_{-\infty}^{+\infty} f(\tau)g(t - \tau) d\tau$$

ניתן לחשב על $(t - \tau)$ כהיפוך של (τ) שביב ציר ה- τ ואז מזוזים את הפונקציה לאורך ציר ה- x ובודקים עבור כל חזזה של t ייחודות ימינה כמה הפונקציות מתלכדות. להמחשה מומלץ לראות את הסרטון (1).

דרך נוספת לקבל אינטואיציה לפועלת הקונובלוציה היא להסתכל על דוגמה הסתברותית, נניח שאנו מטילים שני קוביות ויהיו X, Y המשתנים המקרים שמתארים את התוצאות של הטילות קוביות. אם נרצה לדעת מה ההסתברות שסכום הקוביות הוא t אז נוכל להסתכל על תוצאה הקוביה הראשונה נניח שיצא τ , אז בקוביה השנייה נרצה שיצא $t - \tau$, כלומר ביחסם משלימים $t - \tau$. היה ועשינו את זה לערך t ספציפי ו- t יכול לקבל ערכים בין 1 ל-6 נקבל:

$$f_{X+Y}(t) = \sum_{\tau=1}^6 f_X(\tau) \cdot f_Y(t - \tau)$$

כאשר כאן f_X, f_Y הם פונקציות הצפיפות של המשתנים X, Y , בהתאם.

תכונות הקונובלוציה

להלן תכונות בהם השתמש במהלך הקורס:

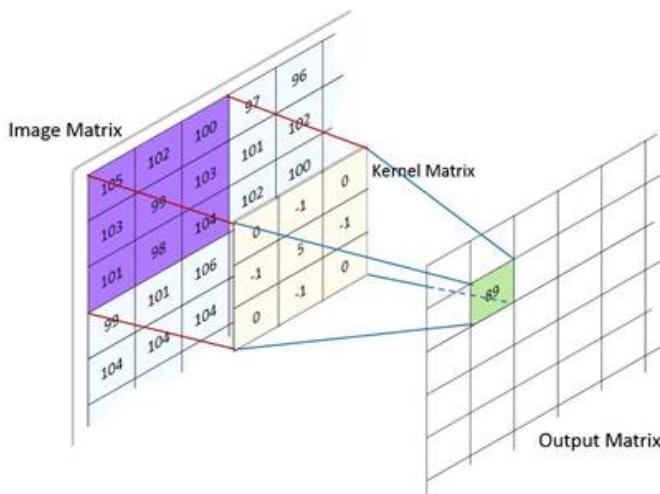
1. קומוטטיביות: $f * g = g * f$.
 2. אסוציאטיביות: $(f * g) * h = f * (g * h)$.
 3. חוג הפילוג (דיסטריבוטיביות): $f * (g + h) = f * g + f * h$.
 4. אסוציאטיביות בכפל בסקל: $g * (\alpha f) = (\alpha f) * g = \alpha(f * g)$.
 5. מ-4,3 ניתן להסיק שקונובלוציה היא טרנספורמציה ליניארית.
 6. שינוי סדר סכימה: $\frac{d}{dx}(f * g) = \frac{df}{dx} * g = f * \frac{dg}{dx}$, היות וקונובלוציה ונגזרת פועלות ליניארית.
 7. באופן דומה: $\frac{\partial}{\partial x_i}(f * g) = \frac{\partial f}{\partial x_i} * g = f * \frac{\partial g}{\partial x_i}$.
 8. אופרטור ההפרש $D_{f*g} \equiv D_f * g \equiv f * D_g$ מקיים $D_f(n) = f(n+1) - f(n)$
- הערה. קונובלוציה מגדרה מרכיב מכפלה מעלה מרחב ליניארי של הפונקציות האינטגרביליות.

הגדרה (קורסילציה): בהינתן שתי פונקציות רציפות f, g , נגידו את פעולה הקורלציה באופן הבא:

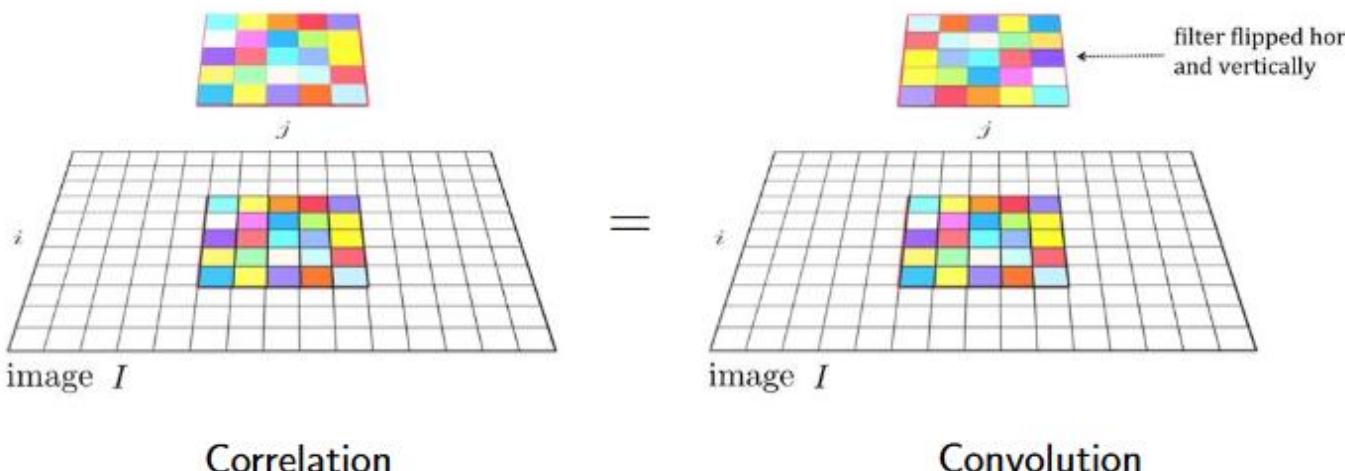
$$f \star g := \int_{-\infty}^{+\infty} f(t) \cdot g(t + \tau) d\tau$$

אזהרה: קומוטטיביות לא בהכרח מתקימת בקורסילציה.

אנחנו עושים בתמונות ולכן אנחנו נתעסק בסכום ולא באינטגרציה, בנוסף באופן החישוב הסכמתי של פעולה הקורלציה מתואר באIOR הבא והчисלוב המתאר את תוצאה הקונובלוציה לצד:



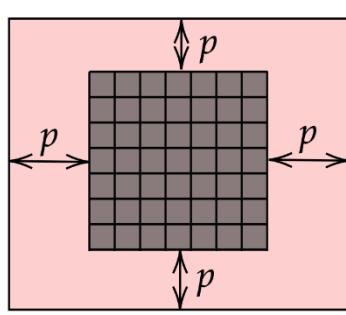
הבדל בין קונבולוציה לקורולציה



Padding – Output size of convolution

משפט (גודל תוצאה הקונבולוציה). בהינתן שני מטריצות ריבועיות בגודל $n \times n$ ו- $k \times k$ המתארים את התמונה והפילטר, תוצאה הקונבולוציה כאשר תבוצע חפיפה מלאה בין שני התמונות היא $(n+k-1) \times (n+k-1)$.

הוכחה. היוות והמטריצה ריבועים מספיק לשקל רק כמה מקומות בשורה אפשר למקם (בה"כ) את הקצה השמאלי העליון של הפילטר. התשובה לכך היא דיאגרמה, נסמן ב- i את האינדקס הימני ביותר שמקיים זאת ונשים לב שמתיקים $a \leq (1-k+1) + i \leq n$. האוי-שוויון הנ"ל מתקיים מכיוון שהאינדקס i כולל את העמודה הראשונה של הפילטר וכן שאר הפילטר נפרש על עוד $1-k$ עמודות ימניות, היית והאינדקס של העמודה הימנית ביותר הוא a האינדקס של העמודה הימנית ביותר שנשקלה ע"י הפילטר חייב להיות קטן מ- n , הטענה נובעת.



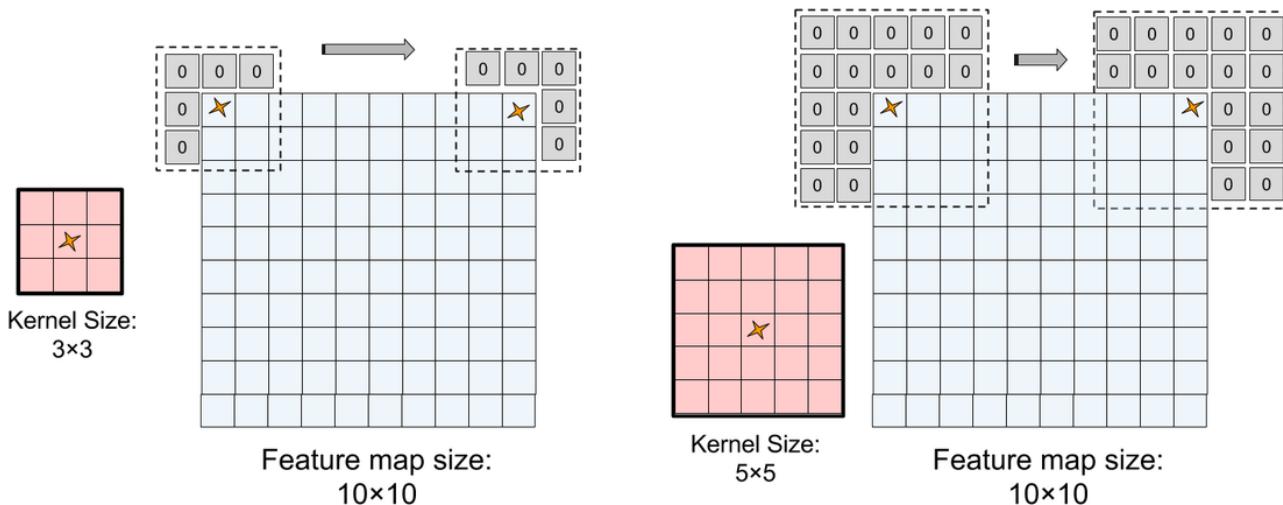
איור 1. תמונה מרופדת באמצעות פרמטר p .

היות ואנו עושים קונבולוציה עם תמונות אנחנו רוצים שהתמונה הנוצרת תהיה באותו גודל, לשם כך אנחנו מرفדים את התמונה באמצעות פרמטר p כמתואר משמאל.

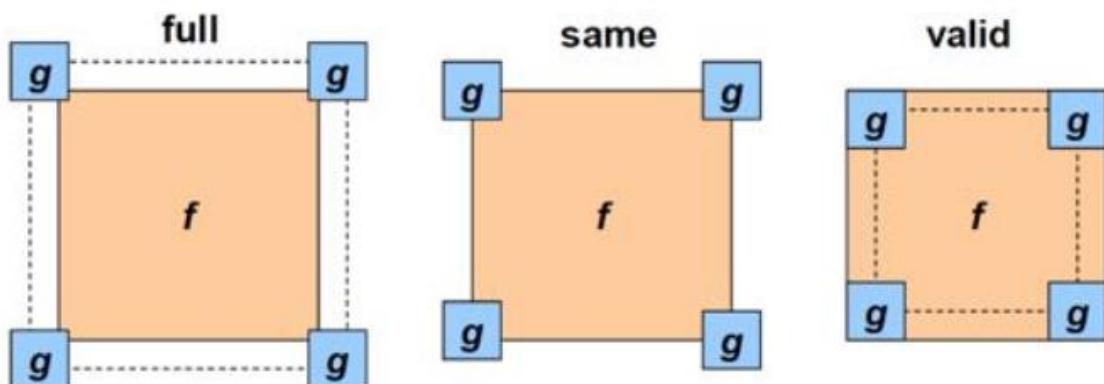
אם נרצה שהתמונה נשמר על גודלה נצטרך לפתור את המשוואה הבאה:

$$(n+2p)-k+1 = n \iff 2p = k-1 \iff p = \frac{k-1}{2}$$

היות פילטר בגודל זוגי הוא דבר לא קובנציונלי שכן הוא לא נותן משמעות לערך האמצעי (כי אין אמצע) אז תוצאת הריפוד תמיד תהיה מספר שלם.

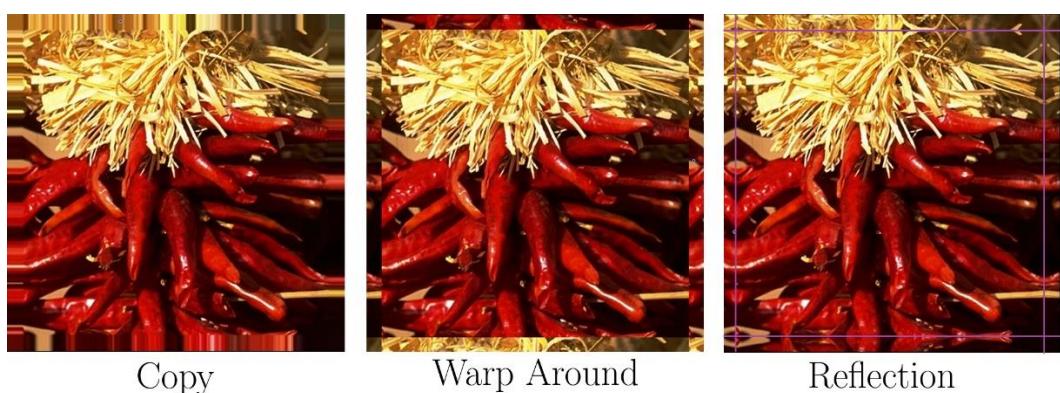


נסכם את הדרכים השונות לבצע קובנולוציה:



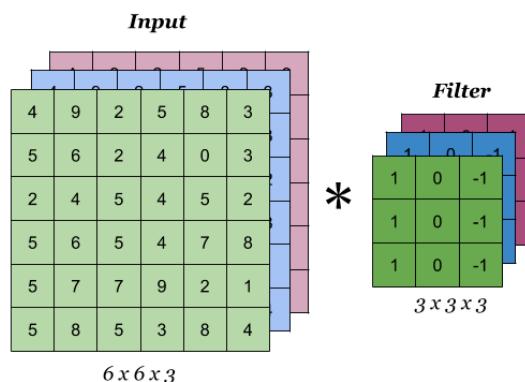
- Full – נרצה את תוצאת הקובנולוציה באופן מלא, כלומר בשניה שני פיקסלים נחפפים כבר תהיה תוצאה, זה שקול לעשות padding עם $1 = p = k - 1$ ונקבל תוצאה של מטריצה ריבועית בגודל $(n + 2p) - k + 1 = n + 2k - 2 - k + 1 = [n + k - 1]$.
- Same – המטרה שגודלו הפליט יהיה כמו גודל התמונה המקורית لكن נרף עם $2/(k-1)$.
- Valid – רצים חפיפה מלאה בין שני הפונקציות, ולכן נקבל ע"פ המשפט התוצאה תהיה $n - k + 1$.

בנוסף יש מספר דרכים לرفד את המטריצה:



- Zero Padding – לרפד באמצעות אפסים.
- Reflection – לשקוף את התמונה, לדוגמה אם הייתה שורה cba אז נוסיף אחר cba .
- Warp around – מחזרי, כלומר חוזרים מלמטה למעלה וככה עוטפים את התמונה.
- Copy – מעתקים את הקצוות.

שני יתרון ל-Reflection: הוא לא ערך קבוע, ובنוסף הסטטיסטיקה של הצבע והעוצמה היא ביחס לסביבה, שהיא מועלה במקורה של הגאומיאן פילטר. לסיום נזכיר שאנו מRADIM תמונה צבעונית אז יש שלושה עורצים וכאן נפעיל את הפילטר על כל ערוץ בנפרד.



שימושים

בלורינג (Blurring) מתייחס לתשטווש של התמונה. לדוגמה באמצעות מיצוע של הסביבה, אחד היתרון של פילטר זה שנקרא גם box-filter הוא שהוא מעלים רעש בצורה טובה.



★ =

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9



דרך נוספת לבצע תשטווש היא באמצעות הגאומיאן:



★ =

0	0	0	5	0	0	0
0	5	18	32	18	5	0
0	18	64	100	64	18	0
5	32	100	100	100	32	5
0	18	64	100	64	18	0
0	5	18	32	18	5	0
0	0	0	5	0	0	0



יתרון של הגאומיאן על המיצוע הוא שהוא שוראים את edges הרבה יותר טוב. זהירות גם ה-box filter וגם הגאומיאן הם לא edge-preserving filters. בסופו יש קרנלים שמאפשרים לזהות קווים, לדוגמה:



★

-1	-1	-1
2	2	2
-1	-1	-1

Horizontal lines

=



דוגמה לקרנל שמאפשר לזהות קווי מתאר שנלמד בהמשך הוא הלפלסיאן:



★

0	-1	0
-1	4	-1
0	-1	0

The laplacian operator

=



כזכור שהקרナル יכול להיות רחוב יותר ואז נקבל את התוצאה הבאה:



★

0	0	-1	0	0
0	-1	-2	-1	0
-1	-2	16	-2	-1
0	-1	-2	-1	0
0	0	-1	0	0

=



שאלה בסיסית שעולה היא מה המשמעות של "תדיות" בתמונה? בהקשר של עיבוד תמונה תדיות מתייחסת לקצב השינוי של ערכי פיקסלים. לפיכך, נוכל לומר שקצבות חדים יהיו בעלי תדיות תוכן גבואה בಗל שערך הפיקסלים משתנים בצורה חדה באותו אזור.

זמן ריצה

נניח שהתמונה בגודל $a \times a$ והקרナル בגודל $k \times k$ אז סיבוכיות זמן הריצה היא $O(n^2k^2)$, אומנם כאשר פילטר נתון להפרדה, כלומר ניתן להציג אותו כתרור $f * g$ כאשר f הוא וקטור عمودה בגודל $1 \times k$ ו- g הוא וקטור שורה $k \times 1$ אזי בוצות תכונת האסוציאטיביות נוכל לחשב $f * g = (f * g) * I = I * (f * g) = K = I * K$. כלומר נפעיל פעמיים קורלציה חד מימדית ונקבל $C = a^2 k^2$ זמן הריצה הינו $O(n^2k^2)$, שזה הרבה יותר טוב.

הגדירה (פילטר נתון להפרדה)

על מנת לדעת אם פילטר K נתון להפרדה נסתכל על פירוק ה-SVD שלו, כלומר $U\Sigma V^T = K$ ונovedא שקיים ערך יחיד (singular-value) בוודא σ_1 שאינו אפס. במידה וכן $u \cdot \sigma_1^{-1} v^T$ הם הפילטרים.

דיהוי קווי מתחאר – Edge Detection

lezioyi kooi matzar shimoshim ravisim v meganim. bagadol behinten kooi matzar anchno b'dar'c nocol lehasik at rob hamidu hadrash leno mahatmona v nebin mah anchno roaim, shkan cel almenet batmona chosom u'i kooi matzar shelou lfi hogdara. aflikzah tebuit makr hia dchishe amatzut kooi matzar. nocol lechshob ul edge (koo matzar) baofen haba: am nazir at batmona b'thalat mid caser hagoba/heuomk b'hatams leutzmat goon haefor az kooi matzar ioafinu amatzut gabbot v bekuvot chdim cpi shmatzar b'ayor haba:

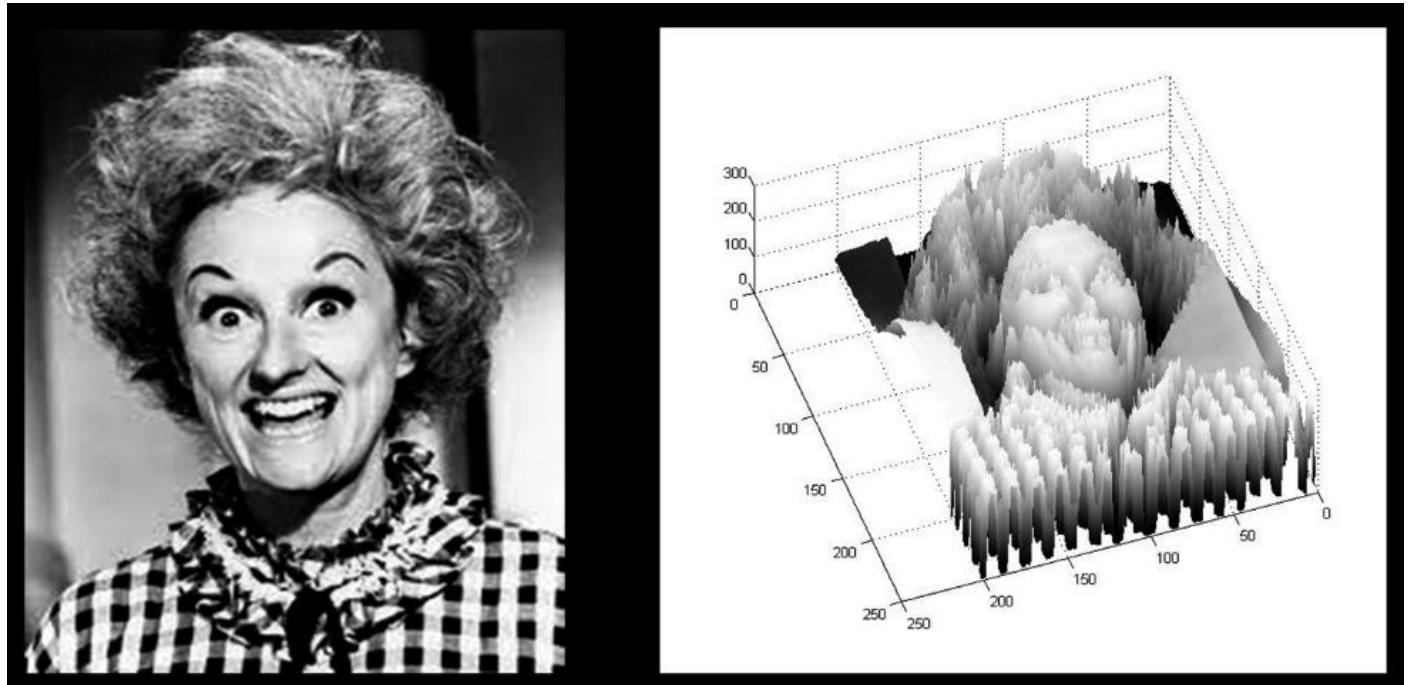
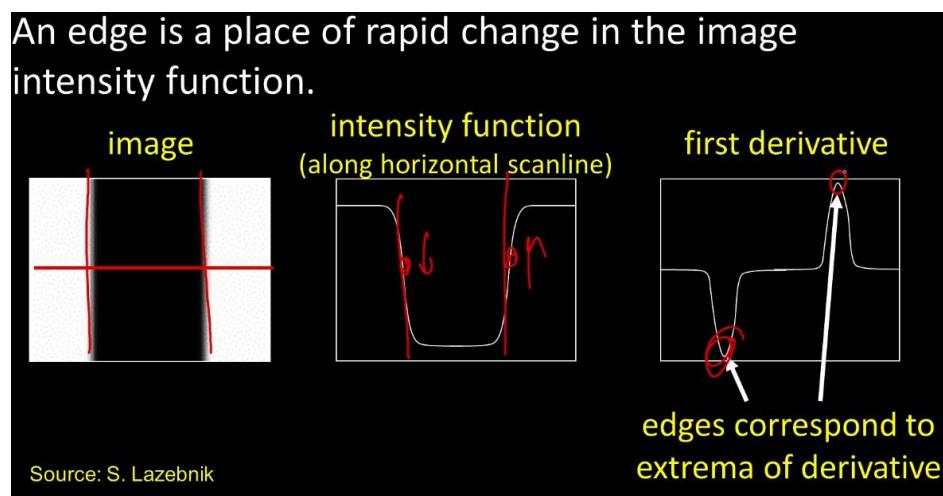
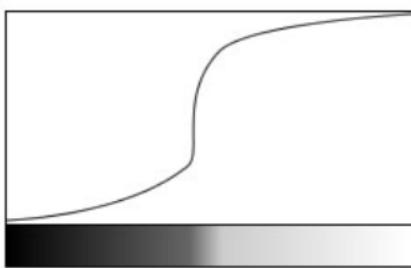


Figure 1: Edges are steep cliffs/slopes.

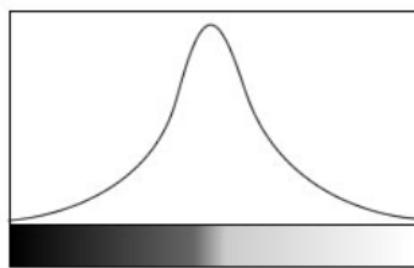
baofen cmotni nocol loemer shkooi matzar mofuiim bgabilot shbin azorim b'ali'i zbeuim shonim, uzemot shonot ao merkamim shonim. hiot anchno m'dbarim ul shinuiim hci gadolim tbeu li'skola at ponkzit hngzrt shkn hia ponkzit shel shinuiim. nstchl ul hadogma haba:



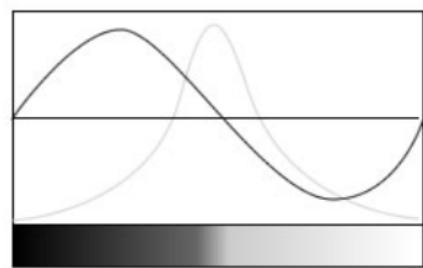
nitan leravot shnitun laafin at kooi matzar amatzut nukodot hakizun hngzrt, l'shem cr nreza l'matzoa nukodot b'hem hngzrt ha'suniya matfasht. o'mannim zeho makraha shbo shinui b'zir haofek b'lbvd. dogma nosftet:



The curve representing intensity



The first derivative of the curve at the left



The second order derivative

המקרה הכללי

נחזיר תחילת על ההגדרה של קו מתאר:

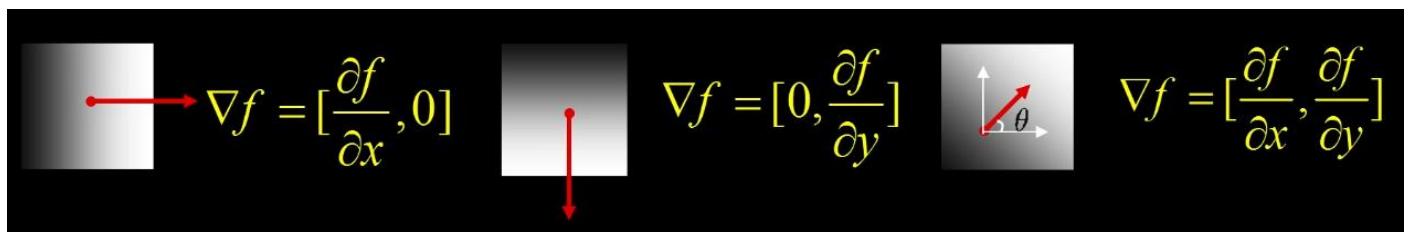
הגדרה (קו מתאר). קו מתאר הוא מיקום של שינוי חד בעוצמת הגוונים.

בקורס חישוב אינפיניטסימלי הגדרנו מושג בשם הגרדיינט שבא לdefs בדיק את מושג זה, ולכן השימוש ב-
cut עולה באופן טבעי. נזכיר כי הגרדיינט מוגדר באופן הבא:

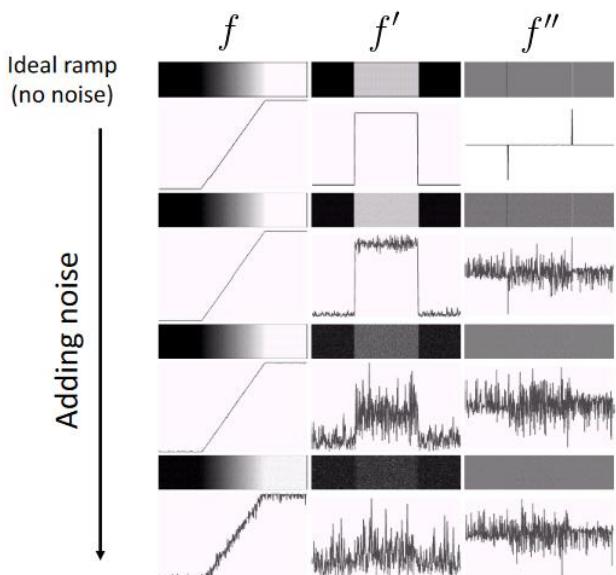
$$\nabla f := \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right)$$

הגרדיינט מצבע על השינוי הקיצוני ביותר. גודלו את גודל השינוי ביחס לצעד מיוחד בכוון הגרדיינט, וכיוונו מאונך לקווי המתאר, כפי שמתואר באיר 2, נתיחס לאיזור הבא משמאלי לימין:

1. השינוי מתבצע רק בכוון האופקי ואין שינוי בכוון האנכי ולכן הרכיב האנכי של הגרדיינט הינו אפס. כלומר כיוון הגרדיינט הוא אופקי.
2. באופן דומה השינוי מתבצע בכוון האנכי ולכן הרכיב האופקי הוא אפס, מכאן כיוון הגרדיינט הוא אנכי.
3. פה השינוי הגדל ביותר אלכסוני, ולכן יש שני רכיבים לגרדיינט.



אייר 2. אילומטרציה של הגרדיינט במקרים שונים.



אכן, ההופעה של נגזרות בעיה זאת אינה מפתיעה כלל שכן נגזרות מחפשות הפרשיים וקו מתאר הוא סוג מיוחד של הפרש בעוצמות של הפונקציה.

אנחנו יודעים שבעוד התמונה המקורית אינה מושפעת באופן משמעותי מרעש הנגזרות רגשות לרעש באופן קיצוני.

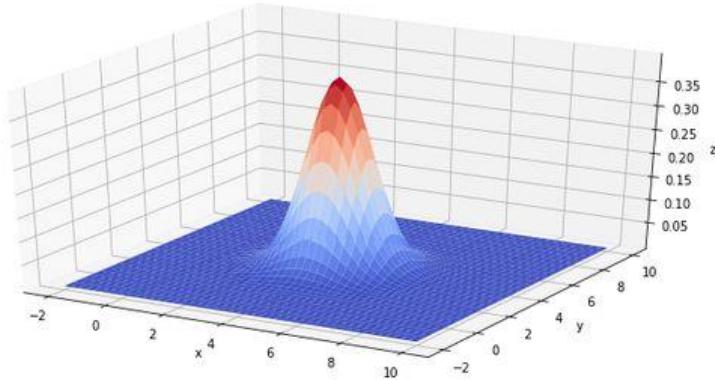
מайдך, אנו יודעים שטשטוש של התמונה מקטין את הרעש ומחליק את העקומה. لكن פתרון מתבקש הוא להחליק ולגזר.

Gaussian filter

זיכרון שהגאוסיאן מחליק את התמונה ושמור על ה-Edges באופן יחסית טוב.

הגאוסיאן מוגדר באופן הבא:

$$G_\sigma(x, y) := \frac{1}{2\pi\sigma^2} \cdot e^{-\frac{x^2+y^2}{2\sigma^2}}$$



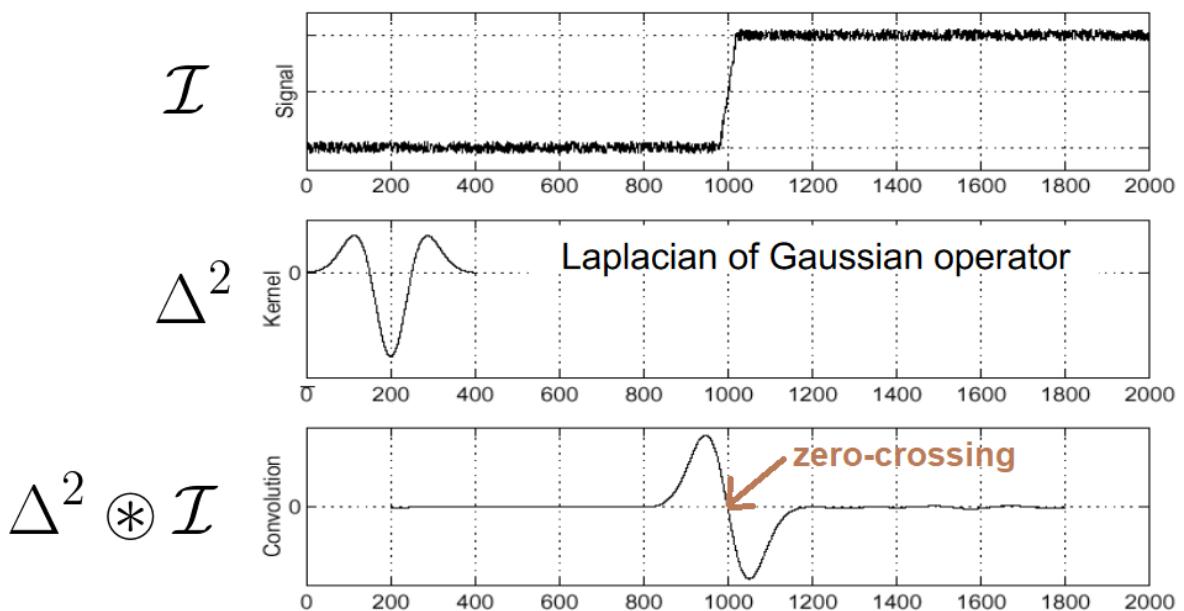
כל ש-σ גדולה יותר כהה התמונה תימרחב יותר.

Laplacian Of Gaussian

על מנת לזהות את ה-Edges הći חזקים אנחנו נגזר עוד פעמיים ונחפש את הנקודות שבהם הנגזרת השנייה היא אפס (zero-crossing) מעבר הנגזרת השנייה חיובית לשילילת ולהפך. לכן נגדיר את הפילטר הבא:

$$LoG := \nabla^2 f := \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} = \begin{pmatrix} 0 & 0 & 0 \\ 1 & -2 & 1 \\ 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 1 & 0 \\ 0 & -2 & 0 \\ 0 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

נחשב את הדוגמה ב- \mathbb{R}^2 :



Where is the edge?

Zero-crossings of bottom graph !

\mathbb{R}^3 הלפלסיאן נראה באופן הבא ואף לעיתים מכונה אופרטור הקובי מקסיקני (mexican hat operator).

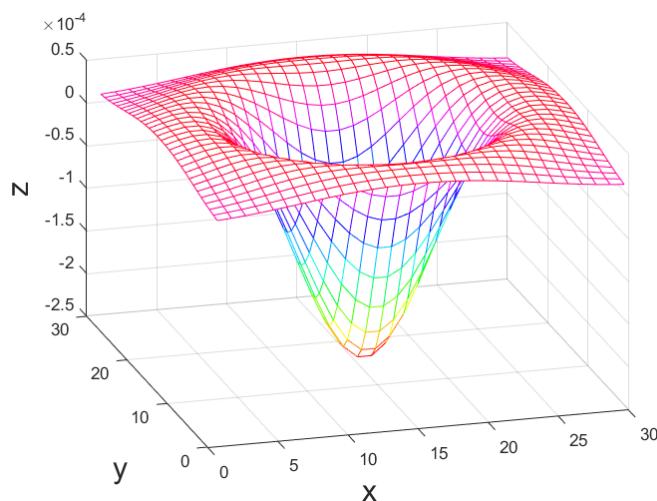
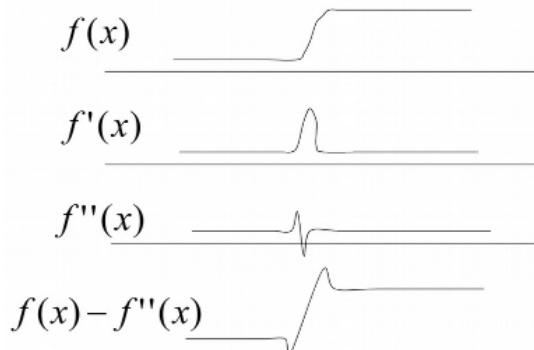


Image sharpening

המטרה: להציג פרטים בתמונה, להגדיל את הקצוות.

השיטה: החסרת הלפלסיאן מהתמונה.



ניתן לראות שלאחר החסרת הלפלסיאן השינויים גדולים יותר ולכן ניתן לבדוק בהם יותר = חדות.

נלקח את הערך של כל פיקסל נחסיר ממנו את הלפלסיאן ונקבל פילטר חידוד:

Subtracting
from the image:

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$



תמונה מקורית

תמונה חדה (אחרי החסרת לפלסיאן)

Sobel's Algorithms

סובל אמר אני אחיליק בציר x ואז אגזר בכוון המאונך כלומר לאורך ציר y ואני אקבל תוצאה טוביה יותר כיון שבכוון שמעניין אותו לא פגעתי בטיבה של התמונה (כי לא טשטשתי בכוון של הגדרה) ויחד עם זאת כן התחמודדתי עם הרעש כי החלטתי בכוון שלא מעוניין אותו.

לדוגמא אם נרצה לקבל את קווי המתאר בציר x ע"י ה-kernel של סובל מסדר 3×3 נחליט בציר y ע"י הגאוסיאן ונגזר בציר x ונקבל:

$$S_x^{(3)} := \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} * [-1 \ 0 \ 1] = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

$$S_y^{(3)} := [1 \ 2 \ 1] * \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Canny's Algorithm

בקצרה Canny תכנן אלגוריתם העונה על התנאים הבאים:

1. **זהוי מדויק.** כל השפות האמיתיות ורק הם צריכות להימצא.
2. **Localization (מיוקומית).** השפות המזוהות ע"י האלגוריתם מאוד קרובות לשפות האמיתיות.
3. **תגובהות ייחודית.** לכל שפה אמיתית יש לבדוק שפה אחת שמזוהה ע"י האלגוריתם (כלומר המקיים הлокאלי עברו כל שפה אמור להיות ייחוד).
4. **שימוש בфиילטרים לינאריים בלבד.**

הוא הבין שהוא צריך להניח מספר הנחות ע"י שיוכל לענות על המטרות שלו:

1. הרעים הם משתנים מקרים בלתי-תלויים וכולם נדגו מאותה התפלגות גאוסיאנית (i.i.d. Gaussian R.V.).
2. השפות הם רציפות וממושכות.

אומנם האלגוריתם משארים מקום ל-Trade-Off שכן ככל שנחליק יותר את התמונה נוכל לזהות את השפות בצורה יותר ברורה וודאית, אומנם זה יפגע בлокאליות של השפה.

- 1. Smooth the input image with a Gaussian filter.**
- 2. Compute the gradient magnitude and angle images.**
- 3. Apply nonmaxima suppression to the gradient magnitude image.**
- 4. Use double thresholding and connectivity analysis to detect and link edges.**

השלב הראשון והשני סטנדרטיים ואמורים להיות ברורים בשלב זה. בשלב שלישי שנקרא Edge Localization ניקח את המקדים הлокאלי של הגרדיינט בסביבה של ה-Edge; נסתכל על היזוית של הגרדיינט ונבדוק האם הערך הנוכחי גדול מהערך לידיו בכיוון הגרדיינט במידה ולא הוא אינו מקדים לוקאלי ולכן נשים אותו, כלומר נעדכן את ערכו לאפס.

השלב האחרון אומר את הדבר הבא: תקבע סף שהחל ממנו אתה בטוח שאתה Edge, ותקבע סף שמתוחת ממנו אתה בטוח שאתה לא Edge, ובמוצע אם אתה שכן של Edge תקבע שאתה Edge, אחרת אתה לא Edge. היות-I-Canny הנית שהצלעות הם ארוכים הוא ימצא את כל הצלעות.

פורמלית נקבל את הפסאדו-קוד הבא:

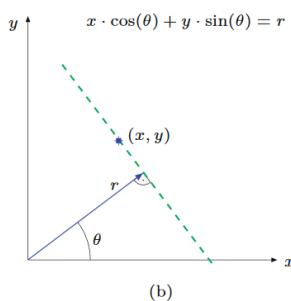
Algorithm Canny Edge Detector

```

1: procedure CANNY EDGE DETECTOR( $\mathcal{I}, \sigma, T_1, T_2$ )
2:    $\mathcal{I} \leftarrow \text{Gaussian}_{\sigma}(\mathcal{I})$ .            $\triangleright$  Smooth using Gaussian kernel with parameter  $\sigma$ 
3:   Compute  $\nabla \leftarrow (\mathcal{I}_x, \mathcal{I}_y)$ .
4:   Preform NON-MAXIMUM SUPPRESSION (Lines 5-7):
5:   if A pixel is not the maximum along the orientation of its gradient then
6:      $\mathcal{I}(p) \leftarrow 0$ .
7:   end if
8:   HYSTERESIS( $\mathcal{I}, T_1, T_2$ )
9: end procedure
10:
11:
12: procedure HYSTERESIS( $\mathcal{I}, T_1, T_2$ )
13:   for  $p \in \mathcal{I}$  do
14:     if  $\|\nabla(p)\| \geq T_1$  then
15:        $p$  presumed to be an edge pixel.
16:     end if
17:     if  $\|\nabla(p)\| \geq T_2$  and  $p$  is connected to an edge pixel then
18:        $p$  considered as an edge.
19:     end if
20:   end for
21: end procedure
```

Hough Transform

עד עתה זיהינו רק קווים מתאר וציינו שבהם חזק מאוד, כתעת ראה שימושים לקווים התמונה; בפרט ראה כי דרך זיהות כל צורה שנייתנת ליצוג באמצעות פרמטר, כלומר ישרים, מעגלים, אליפסות וכו'ב.



אנחנו נתחיל מזיהוי קוויים, לשם כך נזכיר שנייתן ליצג קו ישר באמצעות המשוואה $b + ax = y$, כאשר a מייצג את שיפוע הישר ו- b מייצג את נקודת החיתוך עם ציר ה- y . הבעה בהציג את היעד היא שלא ניתן ליצג בה קוויים אנכיים כדוגמת $x = 0$. לשם כך אנחנו נעבור להציג פולרית של משוואת הקו הישר, כלומר נציג את משוואת הקו הישר באמצעות הזווית θ , והරחק r מרأسית הצירים. לשם כך נשים לב שמתקיים:

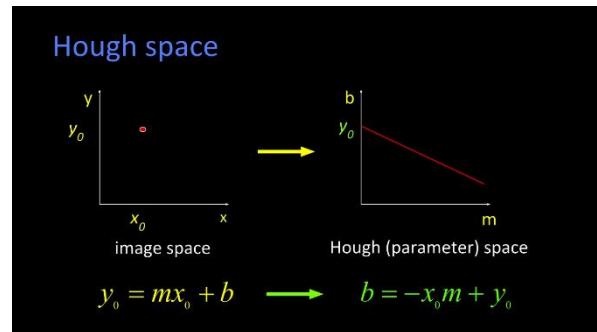
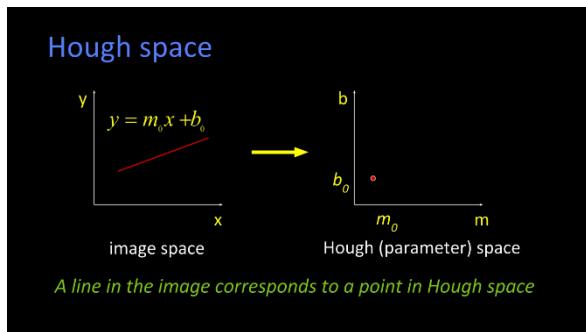
$$x = r \cdot \cos \theta \quad y = r \cdot \sin \theta \quad (\star)$$

אנו יודעים שנייתן להציג את משוואת הישר בתרור $c = n(a, b) = ax + by$ כאשר c הוא וקטור הנורמל. אצלנו הנורמל הוא $(\cos \theta, \sin \theta)$ ולכן נקבל את הביטוי $c = r(\cos \theta, \sin \theta)$, על מנת למצאו את c נציב את $x \cos \theta + y \sin \theta = r$ ונקבל $r = r(\cos^2 \theta + \sin^2 \theta) = r \cdot 1 = r$. \star

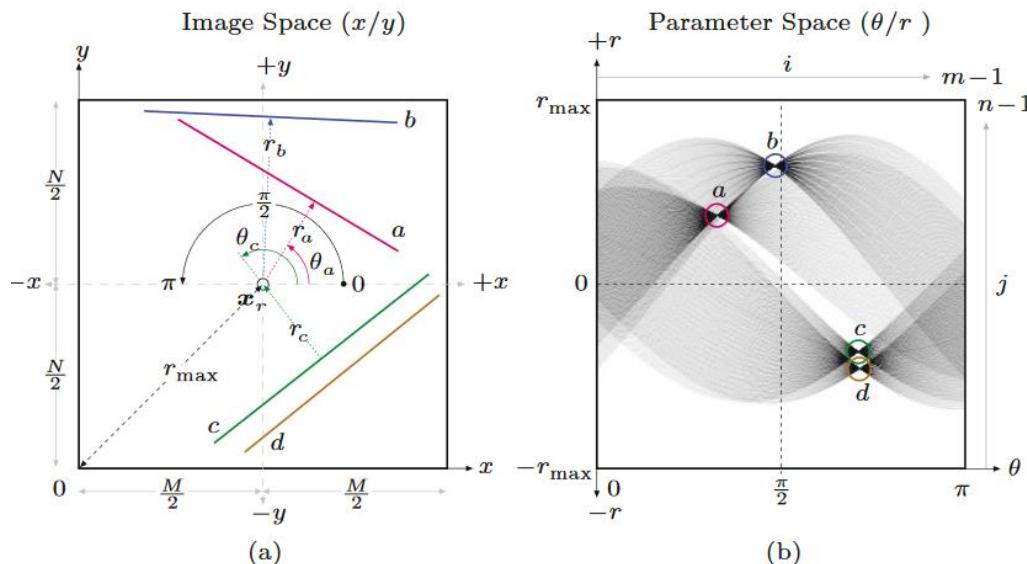
נסמן ב- H_ℓ את הטרנספורמציה למרחב Hough של ישרים ונשים לב שמתקיים:

- כל ישר במרחב התמונה מתאים לנקודה במרחב H_ℓ .
- כל נקודה במרחב התמונה מתאים לקו במרחב H_ℓ .

לשם פשטות נתחיל מציגה הקרטזית של משוואת הישר ולאחר מכן נעבור לפולרית.



בhzga פולרית זה נראה כך:



כעת נציג את הקוד:

1: **HoughTransformLines**(I, m, n, a_{\min})

Input: I , a binary image of size $M \times N$; m , angular accumulator steps; n , radial accumulator steps; a_{\min} , minimum accumulator count per line. Returns a sorted sequence $\mathcal{L} = (L_1, L_2, \dots)$ of the most dominant lines found.

- 2: $(M, N) \leftarrow \text{Size}(I)$
- 3: $(x_r, y_r) \leftarrow \frac{1}{2} \cdot (M, N)$ \triangleright reference point x_r (image center)
- 4: $d_\theta \leftarrow \pi/m$ \triangleright angular step size
- 5: $d_r \leftarrow \sqrt{M^2 + N^2}/n$ \triangleright radial step size
- 6: $j_0 \leftarrow n \div 2$ \triangleright map index for $r = 0$

Step 1 – set up and fill the Hough accumulator:

- 7: Create map $A: [0, m-1] \times [0, n-1] \mapsto \mathbb{Z}$ \triangleright accumulator
- 8: **for all** accumulator cells (i, j) **do**
- 9: $A(i, j) \leftarrow 0$ \triangleright initialize accumulator
- 10: **for all** $(u, v) \in M \times N$ **do** \triangleright scan the image
- 11: **if** $I(u, v) > 0$ **then** $\triangleright I(u, v)$ is a foreground pixel
- 12: $(x, y) \leftarrow (u - x_r, v - y_r)$ \triangleright shift to reference
- 13: **for** $i \leftarrow 0, \dots, m-1$ **do** \triangleright angular coordinate i
- 14: $\theta \leftarrow d_\theta \cdot i$ \triangleright angle, $0 \leq \theta < \pi$
- 15: $r \leftarrow x \cdot \cos(\theta) + y \cdot \sin(\theta)$ \triangleright see Eqn. 8.7
- 16: $j \leftarrow j_0 + \text{round}(r/d_r)$ \triangleright radial coordinate j
- 17: $A(i, j) \leftarrow A(i, j) + 1$ \triangleright increment $A(i, j)$

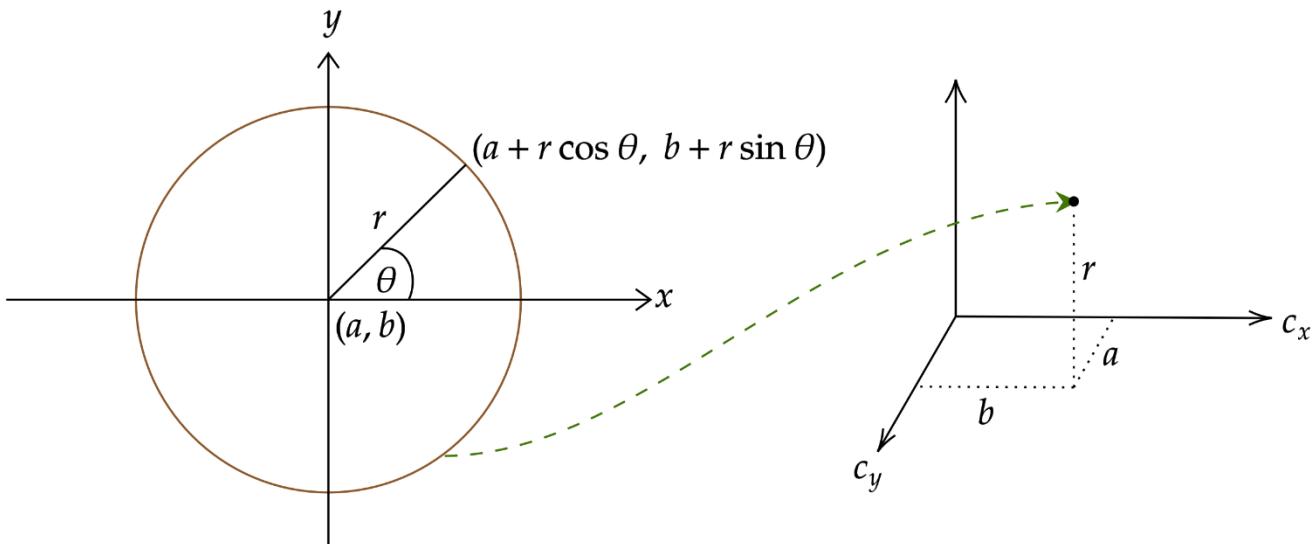
Step 2 – extract the most dominant lines:

- 18: $\mathcal{L} \leftarrow ()$ \triangleright start with empty sequence of lines
- 19: **for all** accumulator cells (i, j) **do** \triangleright collect local maxima
- 20: **if** $(A(i, j) \geq a_{\min}) \wedge \text{IsLocalMax}(A, i, j)$ **then**
- 21: $\theta \leftarrow i \cdot d_\theta$ \triangleright angle θ
- 22: $r \leftarrow (j - j_0) \cdot d_r$ \triangleright radius r
- 23: $a \leftarrow A(i, j)$ \triangleright accumulated value a
- 24: $L \leftarrow \langle \theta, r, a \rangle$ \triangleright create a new line L
- 25: $\mathcal{L} \leftarrow \mathcal{L} \cup (L)$ \triangleright add line L to sequence \mathcal{L}
- 26: Sort(\mathcal{L}) \triangleright sort \mathcal{L} by descending accumulator count a
- 27: **return** \mathcal{L}

מעגלים

תכונת המצב במעגלים היא דומה אם משתמש בצורה הקרטזית נוצר שולשה מימדים (r, a, b) כיוון שמשוואת המעגל הקרטזית נתונה ע"י $(y - c_y)^2 + (x - c_x)^2 = r^2$ היא משווה את המעגל שמרכזו (c_x, c_y) עם רדיוס r .

אם משתמש בצורה הפולרית נקבל θ נקבע $x = a + r \cos \theta, y = b + r \sin \theta$ כמפורט באור:



באופן דומה כל נקודה על המעגל מגדרה חרוט במישור Hough שכן בהינתן נקודה בתמונה אנחנו יכולים באמצעות הגרדיינט לחשב את המשיק לנגזרת ומכך נקבל שנקודות המרכז חייבות להיות על הישר שמאונון לגרדיינט. כלומר נקבל ערכי רדיוסים שונים וכל רדיוס r יגדיר את הנקודה $(x - r \cos \theta, y - r \sin \theta)$ בנקודת מרכז.

נambil את הפסאדו-קוד הבא:

1. $Edges \leftarrow Edge_Detection(Image)$.
2. *For* $(x, y) \in Edges$:
 - a. *Find the orientation* φ of $\nabla Image(x, y)$
 - b. *For* $radius r \in [r_{min}, r_{max}]$:
 - i. $a \leftarrow x - r \cos \varphi$
 - ii. $b \leftarrow y - r \sin \varphi$
 - iii. $Vote[a, b, r] \leftarrow Vote[a, b, r] + 1$

בעת מימוש האלגוריתם נשים לב שצריך ה- y גדול כלפי מטה ולכן נקבל את הקוד הבא:

```

def houghCircle(img:np.ndarray,min_radius:float,max_radius:float)->list:
    (n,m) = img.shape
    r_min = np.int(np.floor(min_radius))
    r_max = np.int(np.ceil(max_radius))
    votes_threshold = 7
    if (img.dtype == np.float):
        img *= 255
    Edges = cv2.Canny(img, 0.07*255,0.15*255)
    dx_kernel = np.array([[-1, 0, 1], [-2, 0, 2], [-1, 0, 1]])
    dy_kernel = np.array([[1, 2, 1], [0, 0, 0], [-1, -2, -1]])
    magnitude, orientation = GradientCV(img, dx_kernel,dy_kernel)
    H = defaultdict(int)
    for x,y in zip(*np.where(Edges == 255)):
        phi = orientation[x, y]
        for r in range(r_min, r_max + 1):
            a = x + int(r * np.sin(np.deg2rad(phi)))
            b = y - int(r * np.cos(np.deg2rad(phi)))
            H[(a,b,r)] += 1
    circles = []
    for a,b,r in {k: v for k, v in sorted(H.items(), key=lambda item: -item[1])}:
        if H[(a,b,r)] >= votes_threshold:
            # If all other centers lies outside the proposed circle with safe distance 1 pixels
            if all((x - a) ** 2 + (y - b) ** 2 > (rc+1) ** 2 for y,x,rc in circles) and \
                (a+r <= n and a-r >= 0 and b+r <= m and b-r >=0):
                circles.append((b,a,r))
        else:
            break
    return circles

```

הערה חשובה: הקERNEL של הנקודות משחק תפקיד מרכזי במיימוש של האלגוריתם זהה. אם נשתמש בנקודות הרגילה הרעש יהיה חזק מדי בדרך"כ ולכן נשתמש בנקודות ע"פ Sobel. בנוסף לא נחלק ב-8.

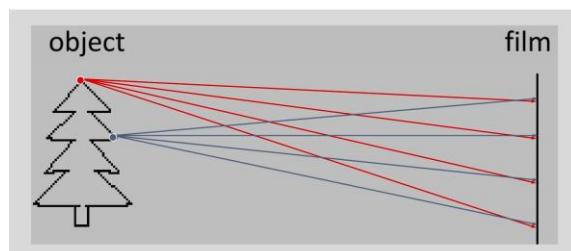
מודל המצלמה

תמונה היא בעצם הטלה של העולם התלת מימדי לעולם הדו מימדי. כתוצאה לכך יכולים להיווצר סיטואציות מעניינות, לדוגמה:

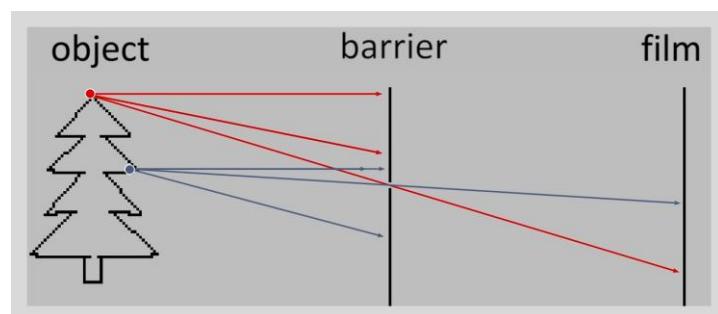


באופן אינטואיטיבי במעבר מתלת-מימד לדו-מימד אובד מידע.

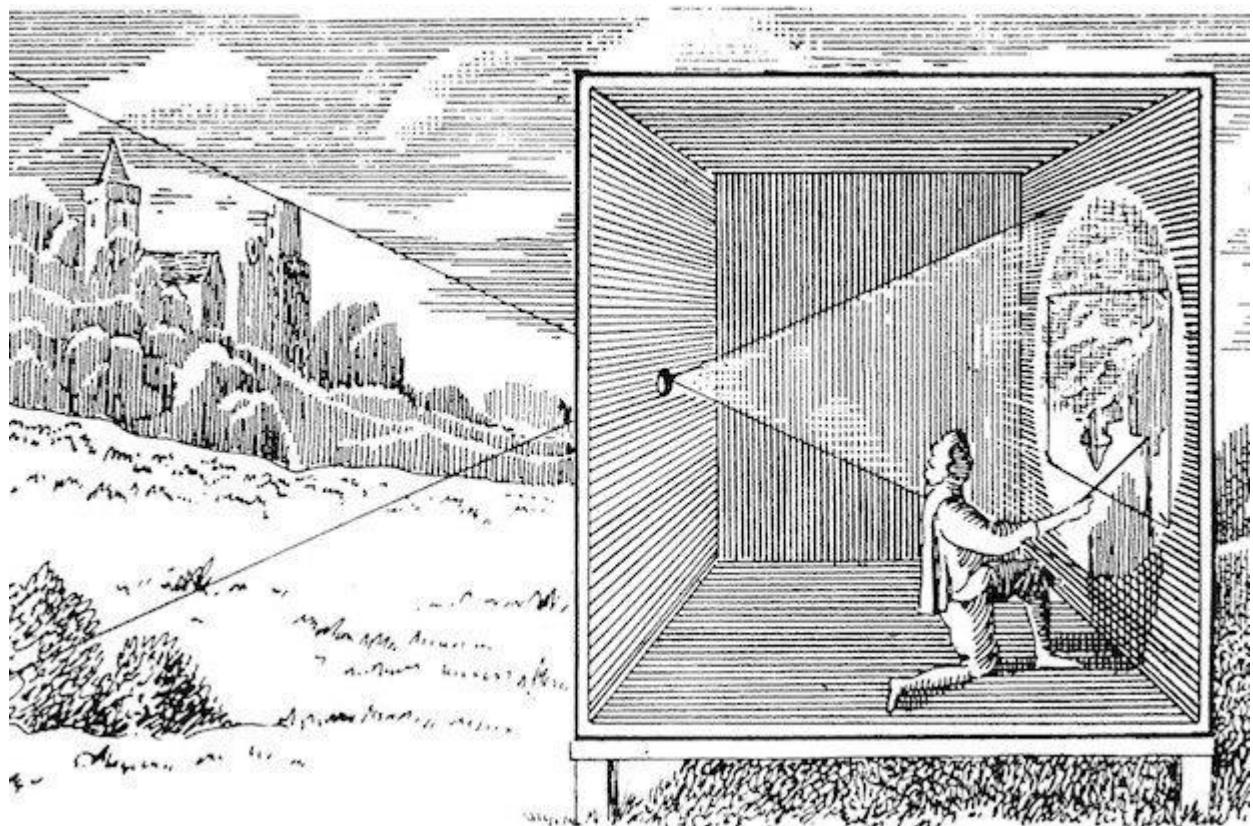
ההבנה הראשונה היא שמלכ נקודה במרחב נשלחות קרניים:



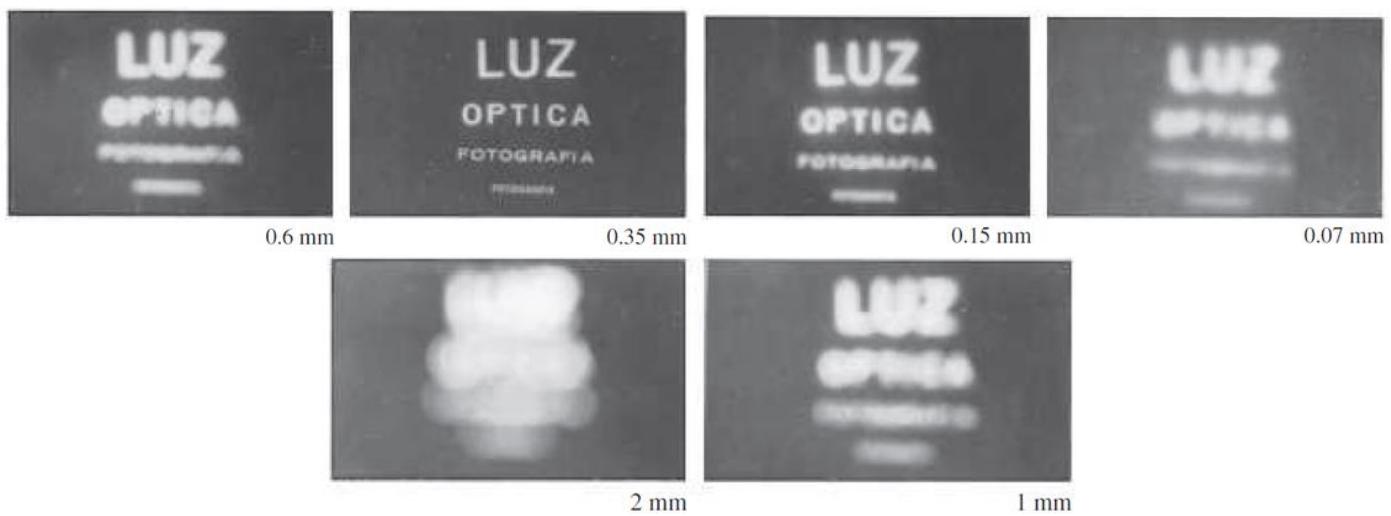
הבעיה היא שככל נקודה בפילם קיבלה את האור מכמה נקודות שונות. פתרון אפשרי הוא לשימוש מחסום (Barrier) ולאפשר חרץ (Aperture) שדרכו עברו קרני האור:



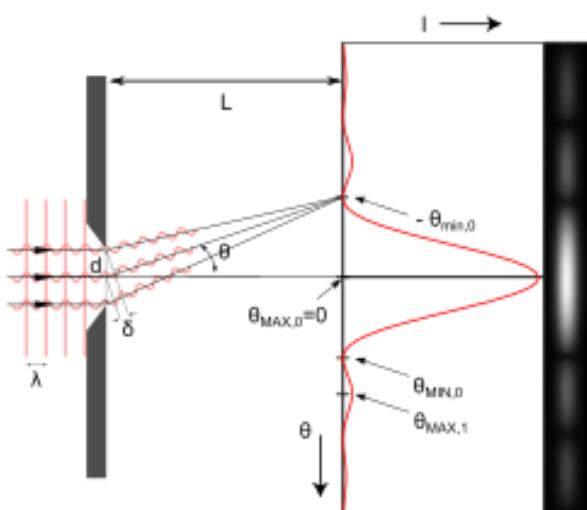
ואז מקבל את הדמות על ה-film, אומנם הדמות תהיה הפוכה. תופעה זאת הייתה ידועה מזמן וודה ויצ' ניסח זאת כך: "כאשר אור של דמיות חודר דרך חור קטן בחדר חשוך במיוחד רואים דמיות אלו בצורותם וצבעם המקוריים מוקטנים בגודלם והפוכים בכיוונם בעקבות חיתוכם של הקרניים".



באIOR הבא ניתן לראות את השפעת גודל הצלם על איכות התמונה:



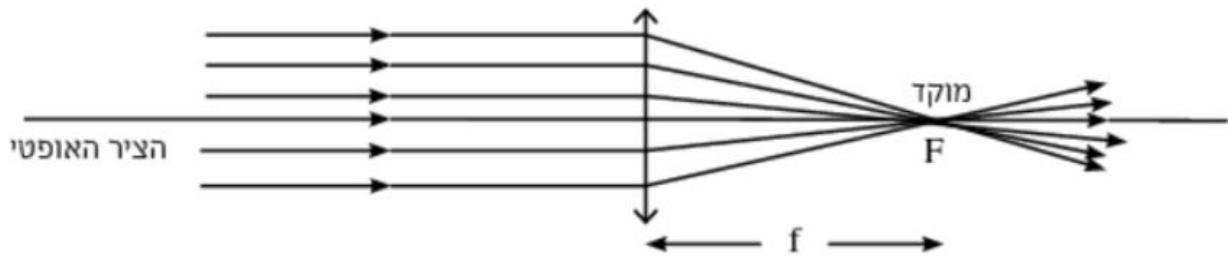
ניתן לראות שיש פה Trade-off אם הצלם קטן מדי אז התמונה יהיה מטושטש מהתאבכות (Diffraction) של הגלים (כמתואר באIOR משמאל). אם הצלם גדול מדי אין שום מיקוד לתמונה והכל יהיה מטושטש.



עדשות

התכונה האופטית של עדשה מרכזת

כל קרני האור המגיעה במקביל לציר האופטי ופוגעת בעדשה, ולאחר מכן נשברות בעדשה הן נפגשות בנקודה אחת הנקראת מוקד. ראו בתחרשים שלפניכם:



מוקד העדשה, F: הנקודה על הציר האופטי של העדשה, שבה נפגשות כל הקרןים המגיעות במקביל לציר האופטי לאחר שבירתו בעדשה.

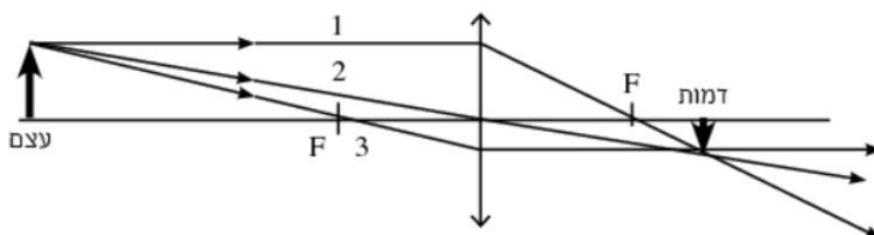
מרחק המוקד של העדשה, f: המרחק מןוקד המוקד אל מרכזו העדשה. לעדשה שני מוקדים שונים צדיה, כאשר הקרןים מגיעות משמאלו, הן נפגשות בנקודה הימני וההפך.

גם כאשר העדשה אינה סימטרית, שני המוקדים נמצאים במרחקים שווים משני צדי העדשה.

נסמן:

- n – מרחק העצם מהעדשה.
- v – מרחק הדמota מהעדשה.

נעיר שמספר קרני האור היוצאות מעצם הוא אינסופי. אנו יודעים כי כל קרן היוצאת מראש העצם וועברת דרך העדשה מגיעה אל ראש הדמota.



עצם ודמותו בעדשה – שימוש בנוסחאות

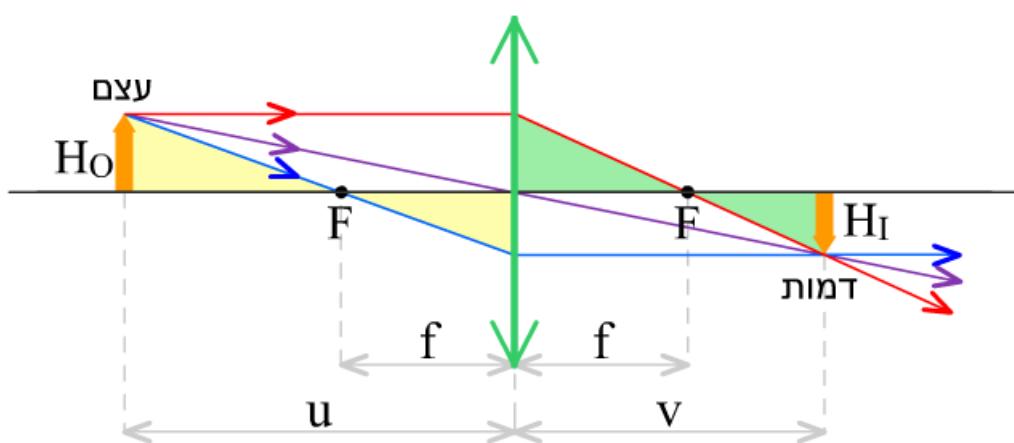
הנוסחאות:

סימונים

- u – מרחק העצם מהעדשה.
- v – מרחק הדמota מהעדשה.
- H_o – גובה העצם.
- H_i – גובה הדמota.
- m – ההגדלה הקויה.

כללי הסימנים:

גודל ממשי נחسب חיובי, גודל מודומה נחسب שלילי, כלומר אם הדמota ממשית אז $m > 0$ ושלילי. לעדשה מרכזת f חיובי, ולעדשה מפזרת f שלילי (המוקד מודומה).



$$\frac{H_I}{H_O} = \frac{f}{u - f}$$

$$\frac{H_I}{H_O} = \frac{v - f}{f}$$

$$\frac{f}{u - f} = \frac{v - f}{f}$$

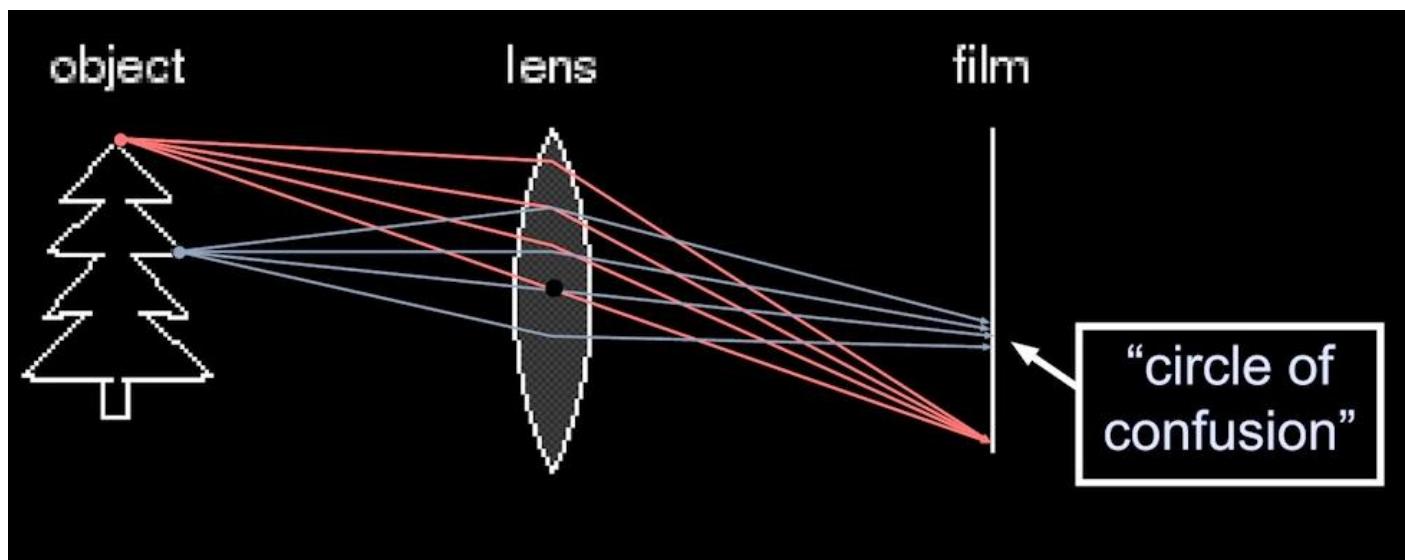
$$f^2 = (u - f)(v - f)$$

$$f^2 = uv - uf - vf + f^2$$

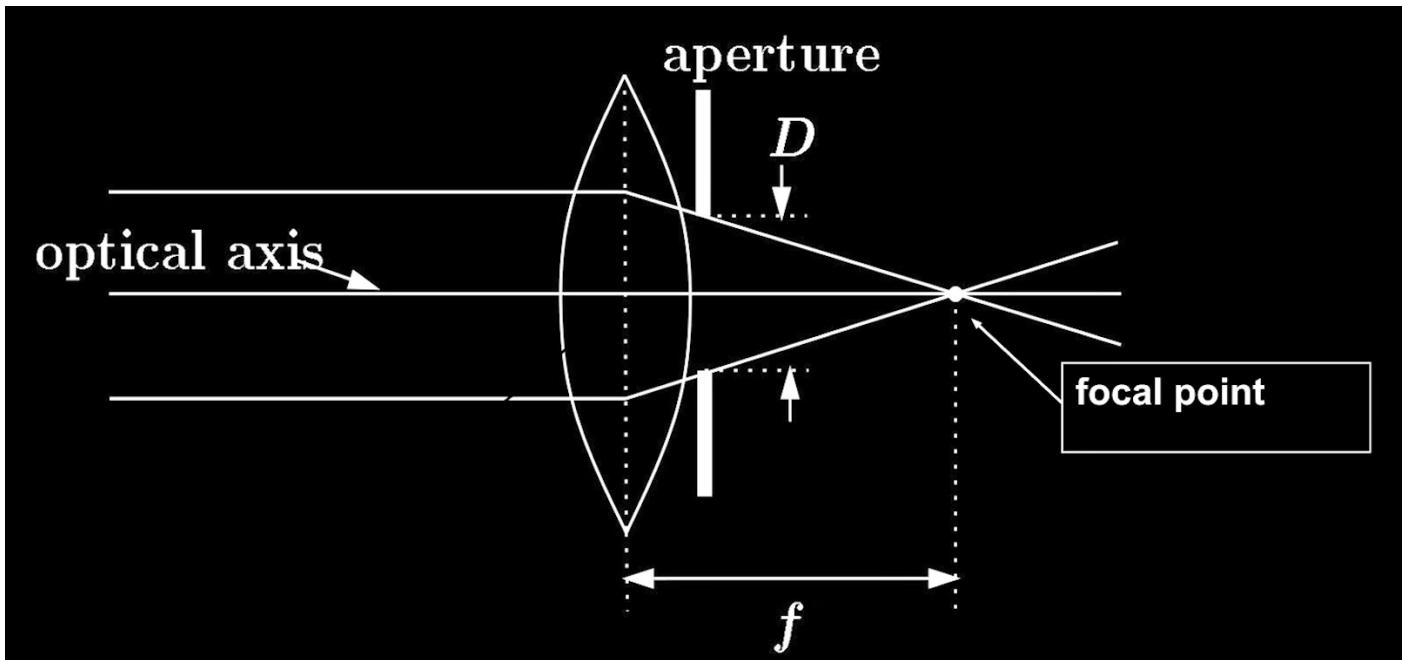
$$vf + uf = uv \quad / :uvf$$

$$\frac{1}{u} + \frac{1}{v} = \frac{1}{f}$$

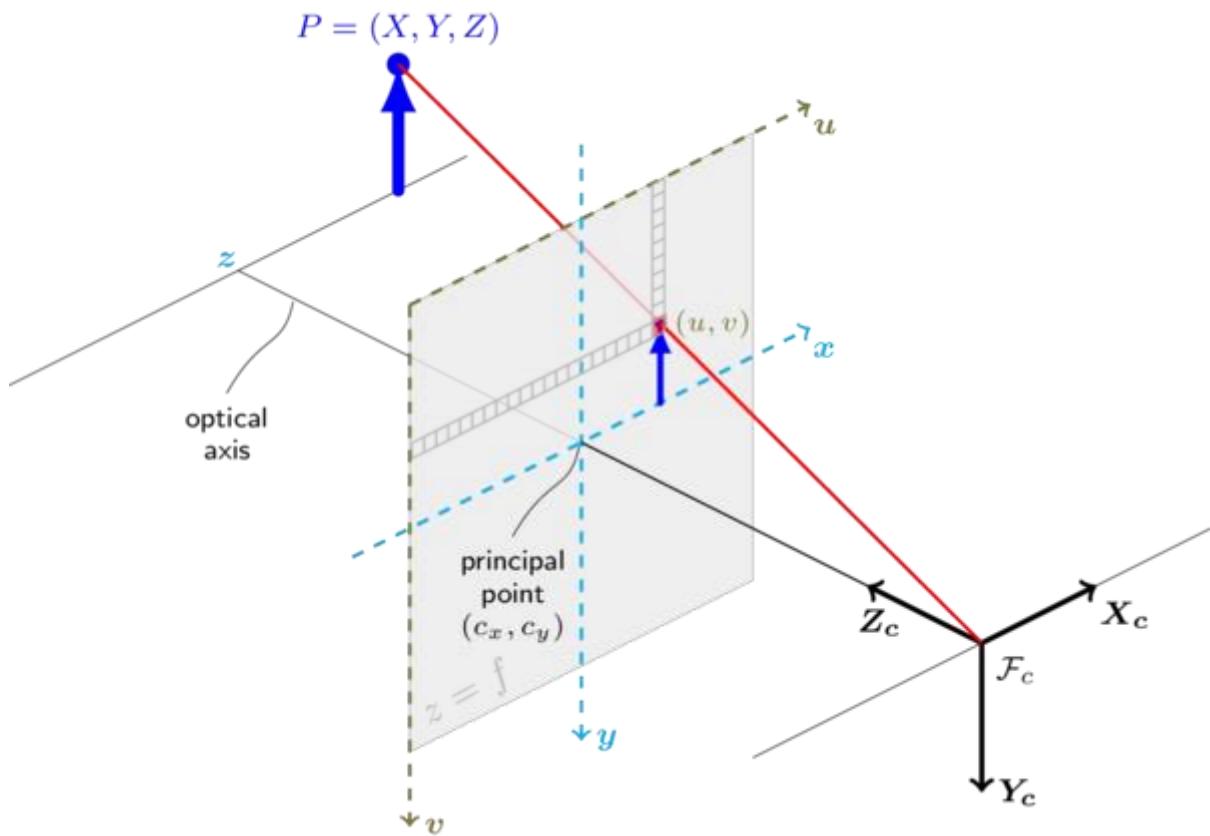
עדשות מרכזות את כל הנקודות מציר אנכי מסוים אבל אם הקרןיהם יוצאים מציר אחר אז לא כל הקרןיהם יתפקסו על מקום בודד בסרט ואז יהיה עיגול של אי-בהירות/בלבול.



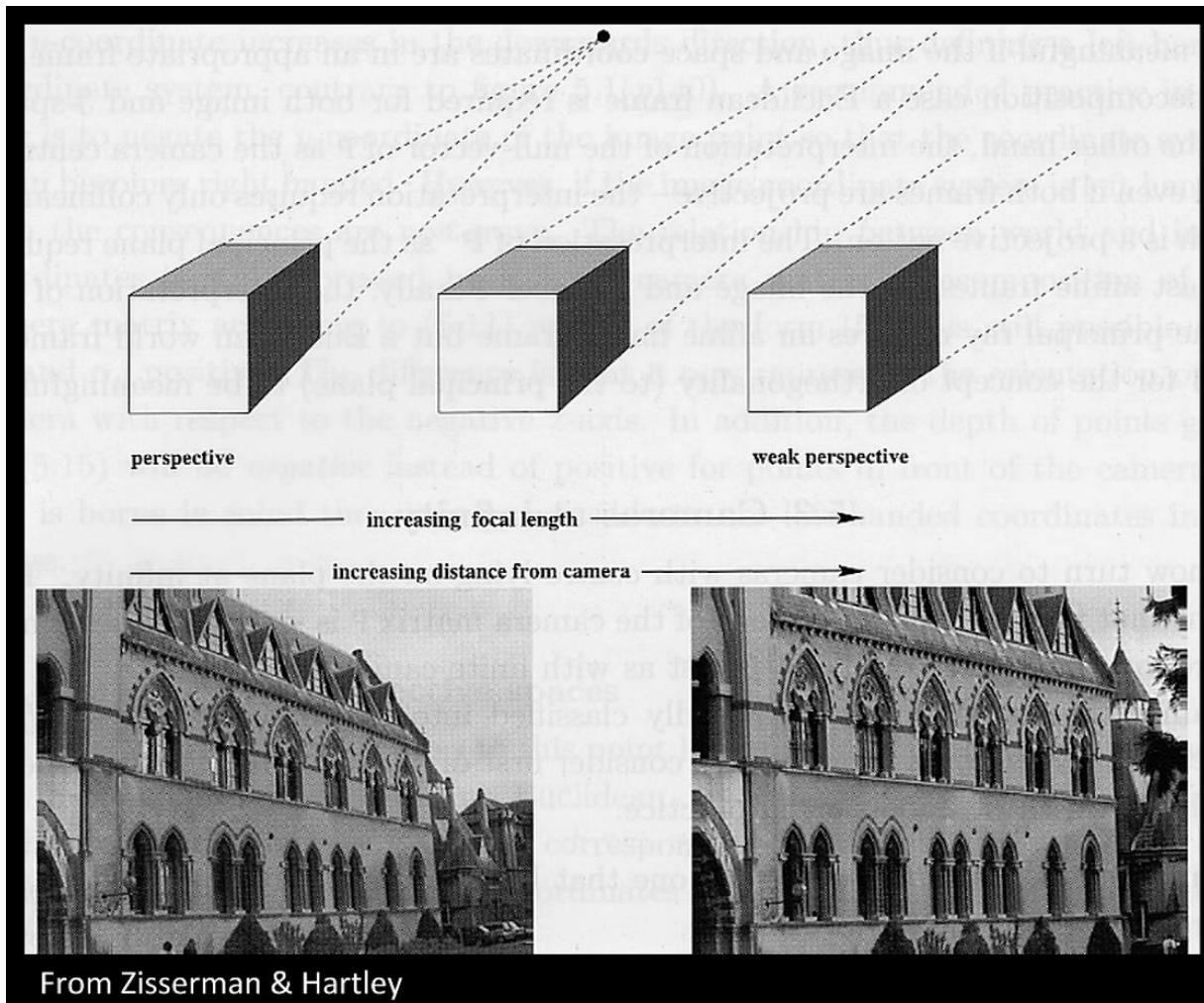
באופן כללי המודל של המצלמה הוא כמתואר באירוע הבא:



- הציר האופטי – ציר שאינו נשרב דרך העדשה וعليו יושבת נקודת המרכז.
- נקודות המרכז היא הנקודה אליה מתרכזות כל הקרןים המקבילות לציר האופטי.
- הצלם Aperture – חסם את כניסה של חלק מהקרןים על מנת למנוע תש透ש.
- מרחק הפוקוס הוא המרחק בין העדשה לנקודת המרכז.



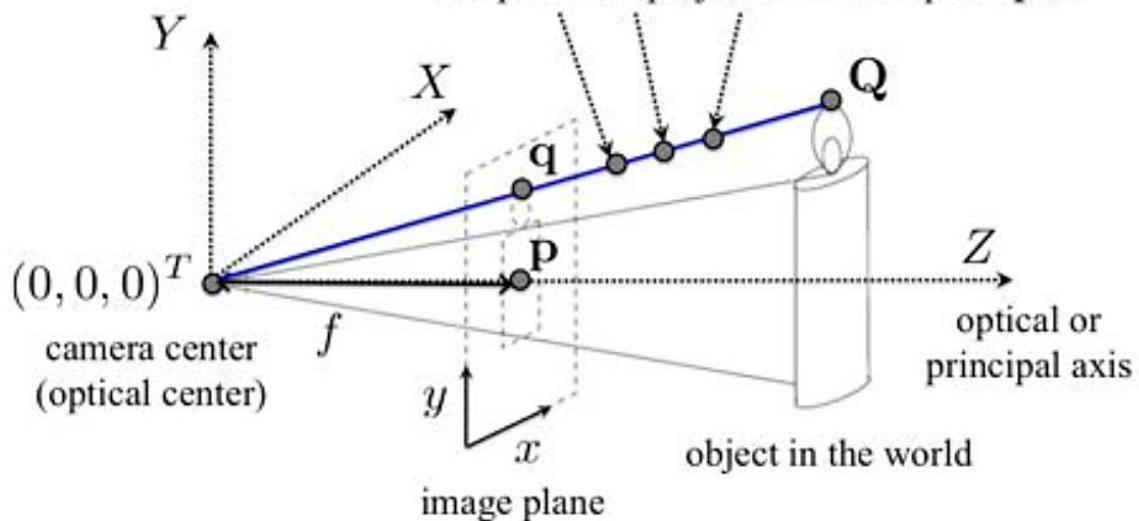
חשוב לשיכר שיש הבדל בין ללקחת זום לתמונה לבין להתקerb לתמונה:



בנוסף נשים לב שככל הנקודות על הישר Q ממופות לאותה נקודה זהה בעצם אומר שאין לנו יכולת לדעת את העומק של אובייקט בתמונה באמצעות תמונה בודדת.

camera coordinate system in 3D

all points on projection line map to \mathbf{q} !!!



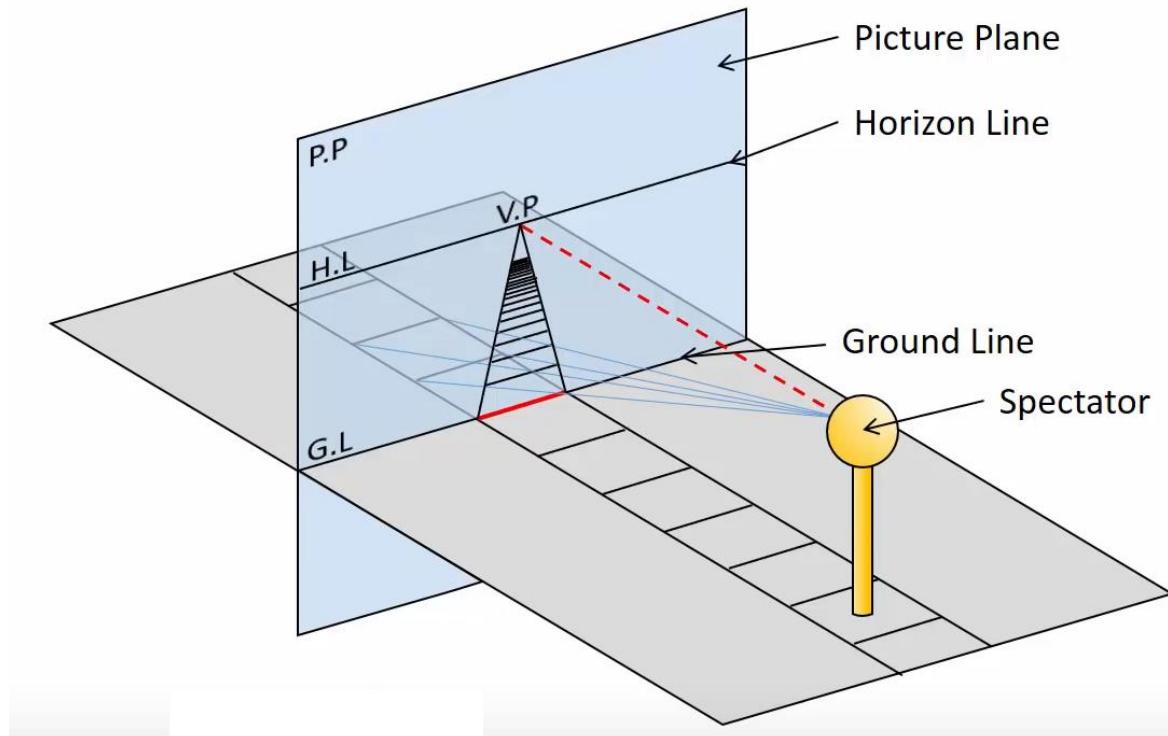
שם הפעולה	צורה מטריצית
scaling	$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$
In-Plane Rotation	$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos \Theta & -\sin \Theta & 0 \\ \sin \Theta & \cos \Theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$
Translation	$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$
Shear	$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & a & 0 \\ b & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$
Affine – שילוב של כל הנ"ל	$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$
Homography (projective)	$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$

הטלה אפינית שנייתנת להצגה כהרכבה של **Translation, Scale, Rotation** תיקרא **SIMILARITY TRANSFORMATION**. בנוסף, אם $a, e \neq \pm 1$, אז היא תיקרא הטלה איזומטרית **ISOMETRIC TRANSFORMATION**, וזה אומר שהיא שומרת על מרחקים בין שני נקודות, זוויות ועוד.

Projective	Affine	Linear	
מעבר ל-(c, f)	מעבר ל-(c, f)	נשאר (0,0)	ראשית הצירים
לא נשמר	נשמרים	नשמרים	Ratios
לא נשמרות	רק אם היא איזומטרית	נשמרות	זוויות
מרובע	מקבילית	מלבן	מיפוי של מלבן
לא מקבילים	מקבילים	מקבילים	קוויים מקבילים
לא סגור	סגור	סגור	סגירות להרכבה
7	6	4	דרגות חופש

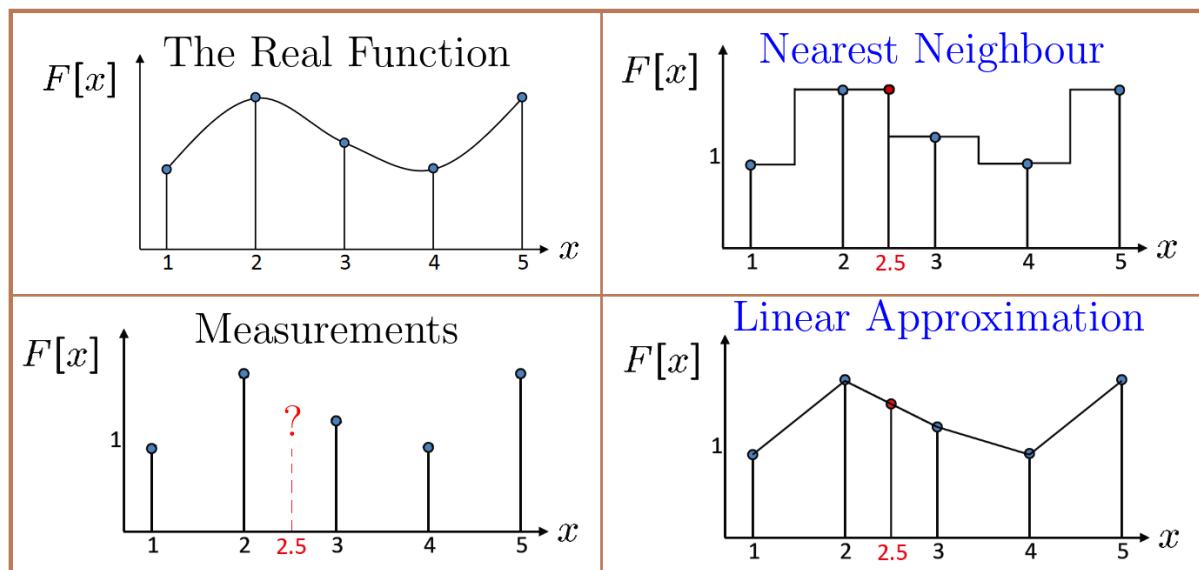
באופן כללי בהטלה פרספקטיבית קווים מקבילים ישארו מקבילים רק אם ציר העומק של המצלמה מאונך למישור של הישרים המקבילים.

כתוצאה מהטלה פרספקטיבית של ושרים מקבילים עם אותו וקטור כיוון ניתן להבחן שכיוון מקבילים בעולם האמתי מופיעים לישרים נחたちים במישור התמונה (כפי שניתן לראות באירוע הבא).

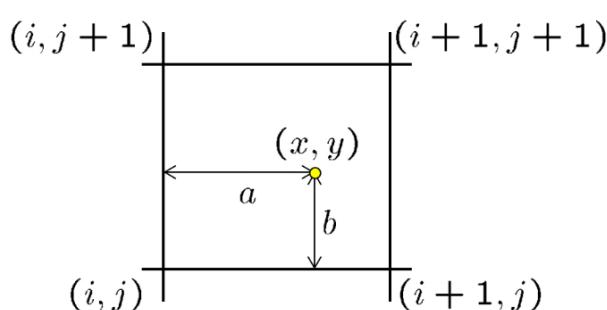
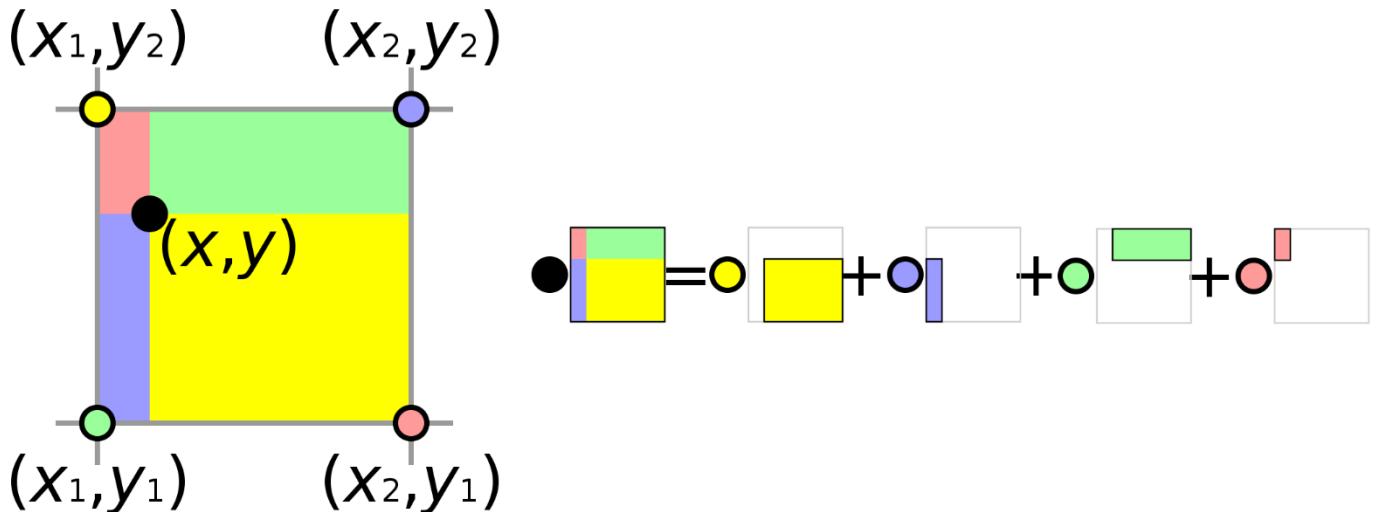


נניח שאנו רוצים להפעיל טרנספורמציה T על תמונה, ישנו שני דרכים לעשות זאת:

- FORWARD WRAPPING – לחשב את התמונה החדשה $(x', y') = I(T(x, y))$, הביעיותות בשיטה זאת היא שנקבל لأن כל פיקסל ממופה אבל יהיו חורים בתמונה. לפיכך, נדרש לפחות פרג את הצבע של החורים בעזרת השכנים, פעולה זאת נקראת splatting.
- BACKWARD WARPING – עטיפה לאחר – מה שנעשה הוא שנגדיר $(y, x) \leftarrow T^{-1}(x', y')$ ואמ הערך שמתקיים לא יהיה שלם נבצע אינטראפולציה (מעין ממוצע משוכל בין השכנים), יש מספר סוג אינטראפולציה: חיפוש שכן קרוב ביותר, אינטראפולציה לינארית, אינטראפולציה בי-לינארית, אינטראפולציה ריבועית, אינטראפולציה בי-ריבועית, וכו'ב. אנחנו נסקרו חלק מהאינטראפולציות.



כעת נציג אינטראפולציה בילינארית שהוצגה בכיתה:



אצלנו יהיה המצב הבא:

$$I'(x, y) \leftarrow ab + (1 - a)b + (1 - b)a + (1 - a)(1 - b)$$

ונקבל

מציאת הפרמטרים המתאים לטרנספורמציה

ראינו טרנספורמציות שונות וכן איך להפעיל אותן על תמונות. כעת נניח שיש לנו שתי תמונות ויש לנו התאמות של נקודות בהם איזו נרצה למצוא את הפרמטרים של הטרנספורמציה. בעצם נרצה למצוא את הטרנספורמציה A כך שהמפרק בין כל זוג נקודות מתאימים (x'_i, x_i) יהיה מינימלי בnormה האוקלידית, ככלمر נחפש את A שסatisfies את

$$\sum_{i=1}^n \|Ax_i - x'_i\|^2$$

טרנספורמציה אפינית. המטריצה A תהיה מהצורה הבאה:

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_2 & y_2 & 1 \\ \vdots & & & & & \\ x_n & y_n & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_n & y_n & 1 \end{bmatrix} = \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix} = \begin{bmatrix} x'_1 \\ y'_1 \\ x'_2 \\ y'_2 \\ \vdots \\ x'_n \\ y'_n \end{bmatrix}$$

$$\mathbf{A}_{2n \times 6} \quad \mathbf{t}_{6 \times 1} = \mathbf{b}_{2n \times 1}$$

בעם אנחנו רוצים לפתור $\mathbf{b} = A\mathbf{t}$ אם המטריצה A הייתה ריבועית היינו מכפילים את שני האגפים משמאלי במטריצה ההופכית ומקבילים פתרון. אנחנו ניצור מטריצה הופכית באופן הבא:

$$A^T A \mathbf{t} = A^T \mathbf{b}$$

עתה נשים לב ש- $A^T A$ היא מטריצה ריבועית, ולכן היא הפיכה נקבע:

$$\boxed{\mathbf{t} = (A^T A)^{-1} A^T \mathbf{b}}$$

פתרון להומוגרפיה.

$$\begin{pmatrix} x'_i \\ y'_i \\ 1 \end{pmatrix} = \begin{pmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix}$$

כלומר אם נפתח את המטריצה נקבל:

$$x'_i = \frac{h_{00}x_i + h_{01}y_i + h_{02}}{h_{20}x_i + h_{21}y_i + h_{22}}, \quad y'_i = \frac{h_{10}x_i + h_{11}y_i + h_{12}}{h_{20}x_i + h_{21}y_i + h_{22}}$$

נרשום זאת בצורה לינארית כתלות ב- h_{00}, \dots, h_{22} באופן הבא:

$$\begin{aligned} x'_i(h_{20}x_i + h_{21}y_i + h_{22}) &= h_{00}x_i + h_{01}y_i + h_{02} \\ y'_i(h_{20}x_i + h_{21}y_i + h_{22}) &= h_{10}x_i + h_{11}y_i + h_{12} \end{aligned}$$

ולפיהם נקבל

$$\begin{aligned} x'_i \cdot (h_{20}x_i + h_{21}y_i + h_{22}) - h_{00}x_i + h_{01}y_i + h_{02} &= 0 \\ y'_i \cdot (h_{20}x_i + h_{21}y_i + h_{22}) - h_{10}x_i + h_{11}y_i + h_{12} &= 0 \end{aligned}$$

עתה נרשום זאת בצורה מטריציונית:

$$\begin{pmatrix} x_i & y_i & 1 & 0 & 0 & 0 & -x'_i x_i & -x'_i y_i & -x'_i \\ 0 & 0 & 0 & x_i & y_i & 1 & -y'_i x_i & -y'_i y_i & -y'_i \end{pmatrix} \cdot \begin{pmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \\ h_{22} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

ובאופן כללי אנחנו צריכים לפחות 4 התאמות (כי יש 8 דרגות חופש), ונקבל את המטריצה

$$\left[\begin{array}{ccccccccc} x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1 x_1 & -x'_1 y_1 & -x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1 x_1 & -y'_1 y_1 & -y'_1 \\ & & & & & & \vdots & & \\ x_n & y_n & 1 & 0 & 0 & 0 & -x'_n x_n & -x'_n y_n & -x'_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -y'_n x_n & -y'_n y_n & -y'_n \end{array} \right] \begin{pmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \\ h_{22} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

A
 $2n \times 9$

h
9

0
 $2n$

על מנת לפתור את זה נשתמש בפירוק ה-SVD של המטריצה:

$$\mathbf{A} = \mathbf{U} \Sigma \mathbf{V}^T = \sum_{i=1}^9 \sigma_i \mathbf{u}_i \mathbf{v}_i^T$$

ortho-normal diagonal ortho-normal
n x m n x n n x m m x m

n x 1 1 x m

ונזכיר שככל

ווקטור عمودה ב- V^T הוא פתרון למשוואה $A\mathbf{h} = 0$, כאשר ה-singular value מתייחס את ה-reprojection error שזה בעצם מידת כלשה למדוד את הטעות لكن נרצה למצוא את הסינגול ווקטור שתואם לעורר הסינגול הכי קטן.

Objective

Given $n \geq 4$ 2D to 2D point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$, determine the 2D homography matrix \mathbf{H} such that $\mathbf{x}'_i = \mathbf{H}\mathbf{x}_i$.

Algorithm

- For each correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ compute the matrix \mathbf{A}_i from (4.1). Only the first two rows need be used in general.
- Assemble the $n 2 \times 9$ matrices \mathbf{A}_i into a single $2n \times 9$ matrix \mathbf{A} .
- Obtain the SVD of \mathbf{A} (section A4.4(p585)). The unit singular vector corresponding to the smallest singular value is the solution \mathbf{h} . Specifically, if $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ with $\mathbf{\Sigma}$ diagonal with positive diagonal entries, arranged in descending order down the diagonal, then \mathbf{h} is the last column of \mathbf{V} .
- The matrix \mathbf{H} is determined from \mathbf{h} as in (4.2).

בספרות ניתן לראות שורותים למנמם את הערך העצמי של $\mathbf{A}^T\mathbf{A}$, כמוון שהוא שקול כפי שניתן להראות:

General form of total least squares

(Warning: change of notation. \mathbf{x} is a vector of parameters!)

$$\begin{aligned} E_{\text{TLS}} &= \sum_i (\mathbf{a}_i \mathbf{x})^2 \\ &= \|\mathbf{Ax}\|^2 \quad (\text{matrix form}) \\ \|\mathbf{x}\|^2 &= 1 \quad \text{constraint} \end{aligned}$$

$$\begin{aligned} \text{minimize } & \|\mathbf{Ax}\|^2 \\ \text{subject to } & \|\mathbf{x}\|^2 = 1 \end{aligned}$$

Solution is the eigenvector corresponding to smallest eigenvalue of $\mathbf{A}^T \mathbf{A}$

(equivalent)



$$\text{minimize } \frac{\|\mathbf{Ax}\|^2}{\|\mathbf{x}\|^2}$$

(Rayleigh quotient)

Solution is the column of \mathbf{V} corresponding to smallest singular value

$$\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T$$

התאמות של פיצ'רים

בහינת שני תמונות אנחנו נרצה לדעת להדביק תמונה אחת על השנייה על מנת לקבל תמונה כוללית יותר. לשם כך נדרש לזוזה פיצ'רים (שהם נקודות חשובות) ולאחר מכן למצוא את ההתאמה בין הנקודות החשובות, ככלומר איזה נקודה בתמונה א' שיכת לאיזה נקודה בתמונה ב'. בסיום אנחנו משתמשים בזוגות של ההתאמות על מנת ליישר את התמונות ולהרכיב את התמונה המלאה.



אייר 3. דוגמה ליישור של שתי תמונות בשבלן לקבל תמונה פנורמתה.

כאמור, אנחנו נדרש שנקודות העניין שלנו יופיעו בשני הנקודות אחרות או בכל מה לדבר על ההתאמה ובטע שלא על יישור של תמונה. בנוסף אנחנו נרצה שיהיה קל להבדיל בין הנקודות שמשמעותן אותן. בפרט אנחנו נשיר לכל נקודה מתאר (descriptor) ואנו נרצה שלנקודות מתאימות יהיה מתאר דומה, ושבין נקודות שלא מתאימות המתאר יהיה שונה.

פיצ'רים טובים

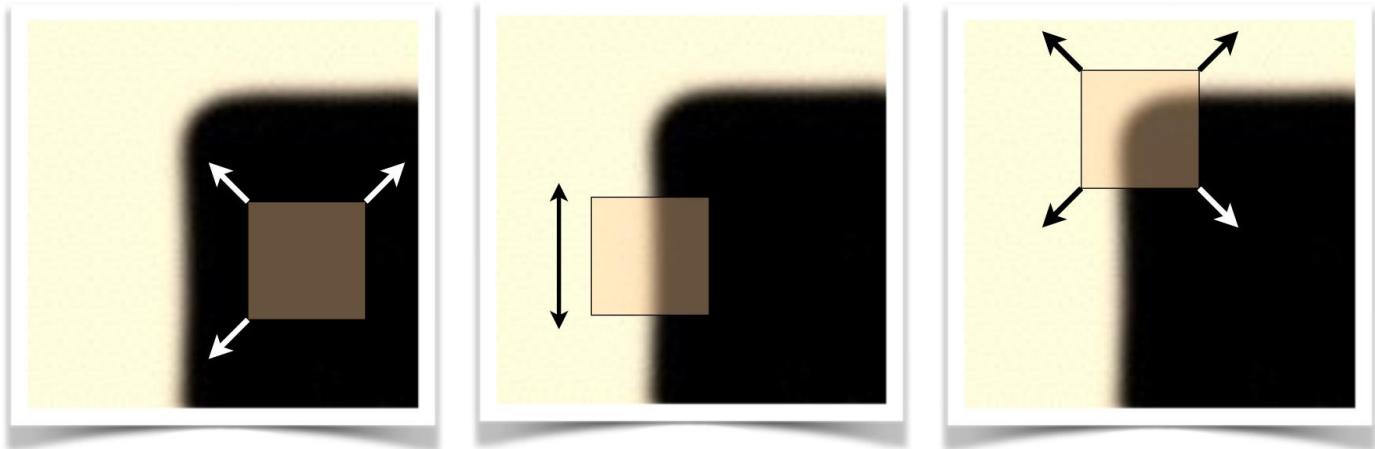
כל אצבע פיצ'ר יקרא טוב אם הוא מכיל שני Edges שונים. פורמלית, פיצ'ר טוב בעל התכונות הבאות:

- ניתן למצוא בדיק איפה הפיצ'ר נמצא בשני התמונות.
- יחידיות - הפיצ'ר חייב להיות ייחודי.
- локאלי - הפיצ'ר צריך להיות קטן בשטחו (בתמונה).

ו ככל שהשטח של הפיצ'ר גדול יותר העייפות שיוצג לו כתוצאה מהטרנספורמציה יהיה גדול יותר וזה יפגע בזיהוי שפיצ'ר הוא זהה בשתי התמונות, لكن אנחנו רוצים סביבה קטנה.

בנוסף, אנחנו נרצה מספר הפיצ'רים יהיה קטן באופן משמעותי ממספר הפיקסלים בתמונה.

על מנת לזוזה באופן חד-משמעות נקבע צורך צרי ששינוי בכיוון מסוים יגרום שינוי בעוצמה של הצבע. אחרת, כל האзор באותו גוון ולא ניתן להבדיל בין שום שני נקודות באזור. נזכיר שינוי בעוצמה יוצר קשת (Edge) אבל קשת אינה מספיקה כי שינוי בכיוון הקשת לא ייצור שינוי בעוצמה. לפיכך אנחנו נרצה שינוי בכל הכוונים ככלומר אנחנו נרצה פינה.



"flat" region:
no change in all
directions

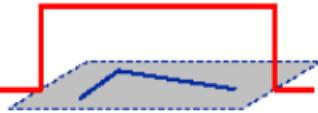
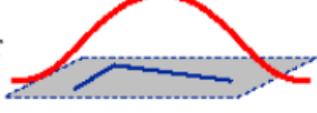
"edge":
no change along the edge
direction

"corner":
significant change in all
directions

Harris Corner Detector

אנו נגיד את הטעות של הזזה ב- u פיקסלים בציר ה- x ו- v פיקסלים בציר ה- y בחלון מסוים Ω באופן הבא:

$$\mathcal{E}(u, v) := \sum_{x, y \in \Omega} w(x, y) \cdot (I[x + u, y + v] - I[x, y])^2$$

Windows function $w(x, y) =$  or 
1 in window, 0 outside Gaussian

כאשר I מציין את התמונה $-(-\cdot, \cdot)$ w מציין את המשקל שאנו נותנים לחלון:

אנו נניח שהתזזה בשני הצללים היא קטנה כלומר u, v סמוכים ל-(0,0). נוכל לקרב את $(I(x + u, y + v) - I(x, y))$ באמצעות פיתוח טילור באופן הבא:

$$I(x + u, y + v) \approx I(x, y) + x \cdot I_x + y \cdot I_y + u \cdot v$$

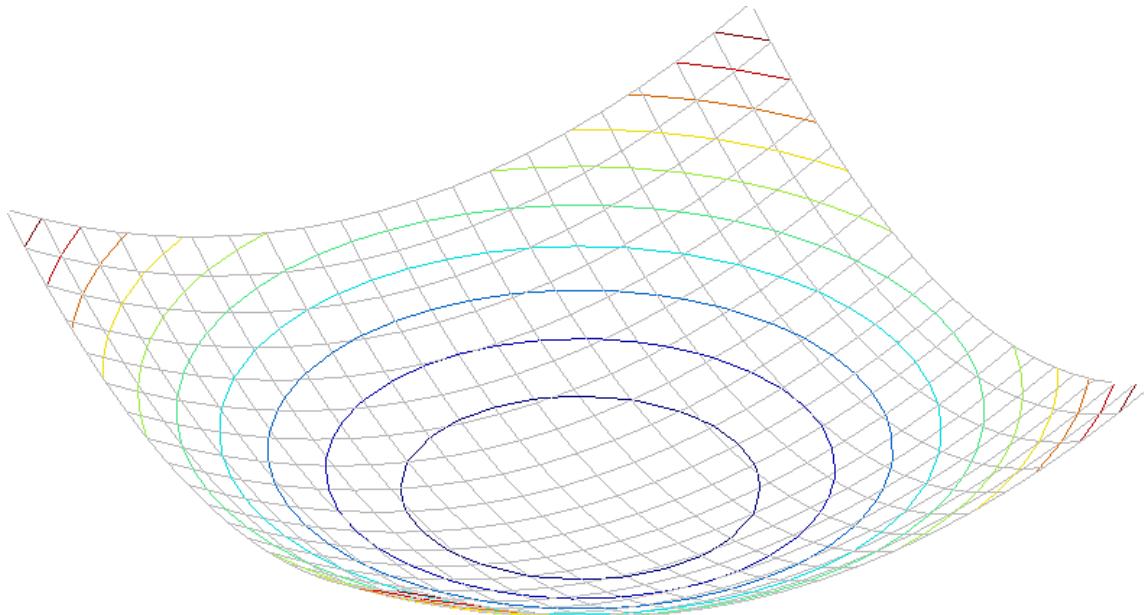
ולכן נקבל

$$\begin{aligned} \mathcal{E}(u, v) &\approx w(x, y) \cdot \sum_{x, y \in \Omega} (I_x \cdot u + I_y \cdot v)^2 \\ &= w(x, y) \cdot \sum_{x, y \in \Omega} I_x^2 \cdot u^2 + 2uv \cdot I_x I_y + I_y^2 \cdot v^2 \\ &= w(x, y) \cdot \sum_{x, y \in \Omega} [u \quad v] \cdot \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \cdot \begin{bmatrix} u \\ v \end{bmatrix} \\ &= [u \quad v] \cdot \underbrace{\left(w(x, y) \cdot \sum_{x, y \in \Omega} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \right)}_M \cdot \begin{bmatrix} u \\ v \end{bmatrix} \end{aligned}$$

נשים לב שאם נפתח את המטריצה $(u, v) \mathcal{E}$ נקבל:

$$\mathcal{E}(u, v) = u^2 \sum_{x,y \in \Omega} I_x^2 + \left(2 \sum_{x,y \in \Omega} I_x I_y \right) \cdot uv + v^2 \cdot \sum_{x,y \in \Omega} I_y^2$$

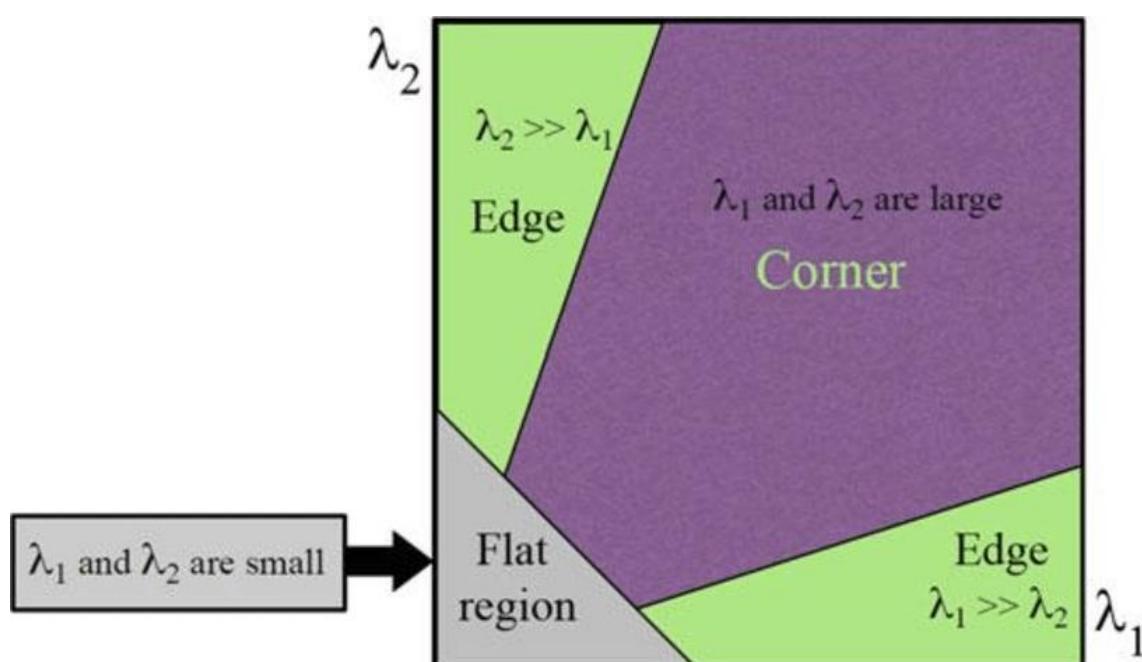
ועבור כל ערך קבוע k שעבורו $k = (u, v)$ קיבל את משווהת האליפסה. כלומר אם נצייר את התמונה בחתלה מימד כאשר ציר ה- z הוא k , נקבל:



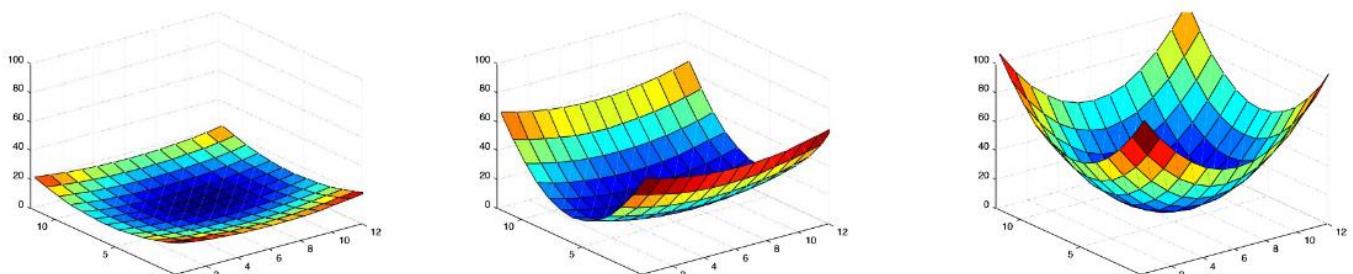
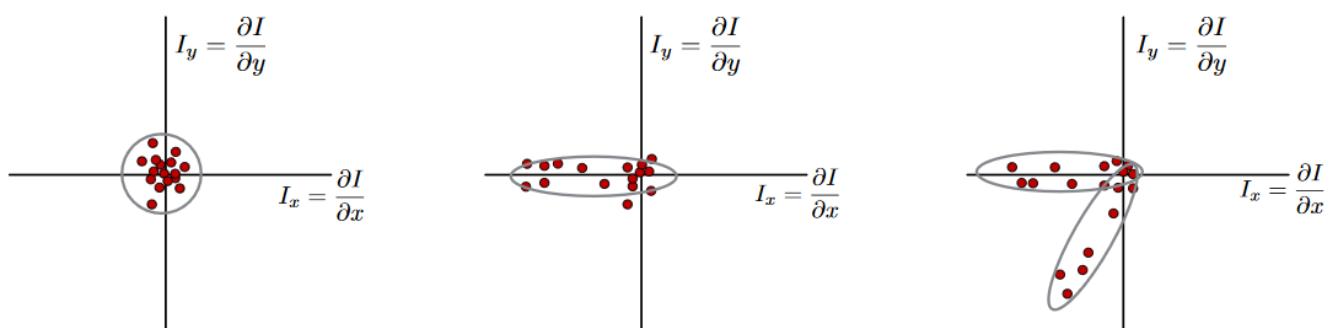
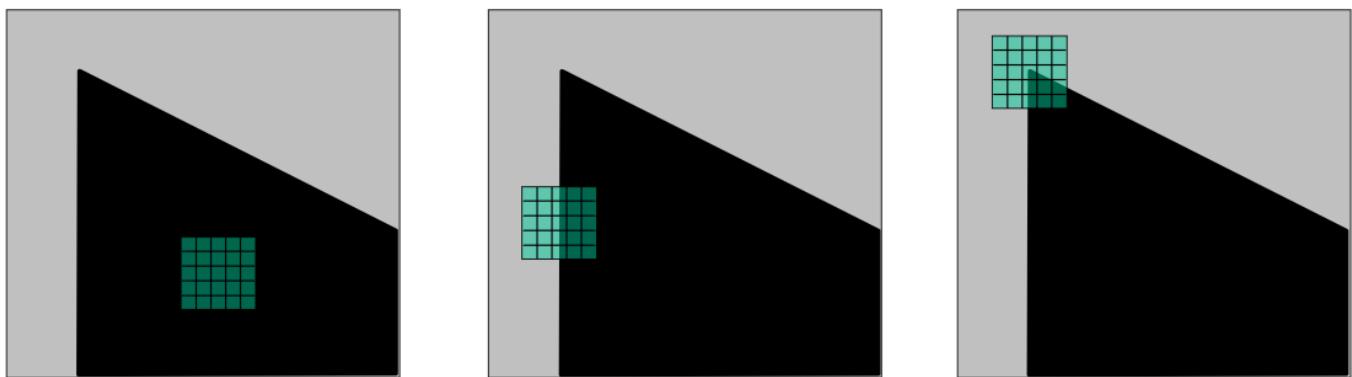
. $a = \lambda_1^{-1/2}$, $b = \lambda_2^{-1/2}$, ניתן להראות ש- $\frac{x^2}{a^2} - \frac{y^2}{b^2} = k$

ומכך נסיק את החלוקה הבאה למקרים:

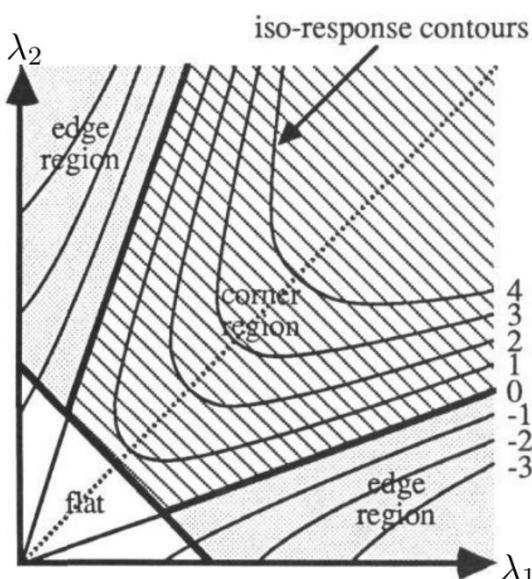
- אם λ_1 ו- λ_2 גדולים אזי השנו בשני הצירים משמעותם כולם זהווינו פינה.
- אם $\lambda_2 \gg \lambda_1$ אזי התזוזה לאורך ציר x הוא המשמעותי ביותר. היהות והגרדיינט מאונך לכיוון ה-*Edge*.
- נסיק שיש *Edge* לאורך ציר y .
- אם $\lambda_1 \gg \lambda_2$ אזי מסובות דומות יש *Edge* לאורך ציר x .
- אחרת, ככלומר λ_1 ו- λ_2 קטנים מאוד איזור שטוח כי השינוי בכל הכוונים הוא כמעט 0.



כעת נראה מספר דוגמאות על מנת להמחיש את הנטואציה ולראות איך המצב נראה בכל אחד מהמקרים.



flat

edge
'line'corner
'dot'

כעת נסקר שני מקרים שונים להשתמש בהם במקום הערכיים העצמיים. הסיבה שאנו מציגים מקרים מודדים אלו נובעת מהעובה שחשבון ערכים עצמיים היא פעולה בעלת סיבוכיות חישוב יקרה יותר מהמודדים שנציגו.

$$1. \quad \alpha \in [0.4, 0.6] \text{ עבור } R = \det(M) - \alpha \cdot \text{trace}^2(M)$$

כאשר הערך של α נמצא באופן היוריסטי. נשים לב,

$$\lambda_1^2 + \lambda_2^2 - \alpha(\lambda_1 + \lambda_2) = R \text{ וכן נקבל:}$$

- עבור פינה נקבל ש- R - גדול.

- עבור צלע נקבל ש- R -שלילי אבל $|R|$ גדול.

- עבור אזור שטוח נקבל ש- $|R|$ קטן.

Kanade & Tomasi (1994) .2

$$R = \min(\lambda_1, \lambda_2)$$

Nobel (1998) .3

$$R = \frac{\det(M)}{\text{trace}(M)} = \frac{\lambda_1 \cdot \lambda_2}{\lambda_1 + \lambda_2}$$

האלגוריתם

Algorithm Harris Detector

- 1: **procedure** HARRIS DETECTOR(I)
- 2: Compute Gaussian derivatives at each pixel.
- 3: Compute second moment Matrix

$$M = w(x, y) \sum_{x, y \in \Omega} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}.$$

- 4: Compute corner response function

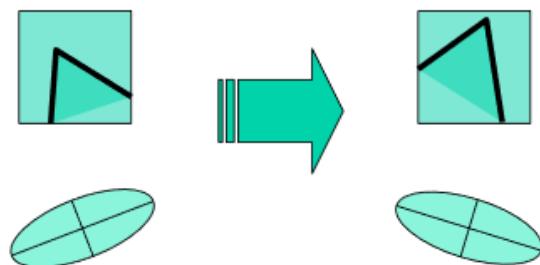
$$R = \det(M) - \alpha \text{trace}^2(M).$$

- 5: Threshold R
- 6: Non-maximum suppression. ▷ Remove all surroundings non-maxima.
- 7: **end procedure**

תכונות של אלגוריתם Harris Detector

Harris Corner Detector Resilient For •

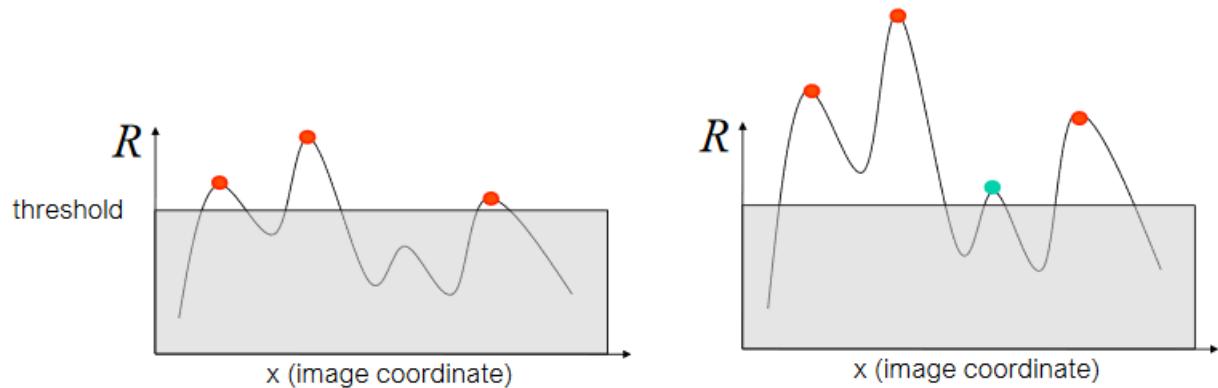
- **רוטציה** – מטריצת ההזזה משתנה אבל הערכים העצמיים לא ולקן הריס עמיד לרטוציה.



- **הוספת סקלר:** $I + b \leftarrow I$

זה נכון כי רק הנגזרות משפיעות ולכן קבועים חסרי חשיבות עבור מציאת פינות.

- כפל העוצמה בסקלר: $I \leftarrow a \cdot I$
- יש לשנות את הספ' למה נחשב Corner

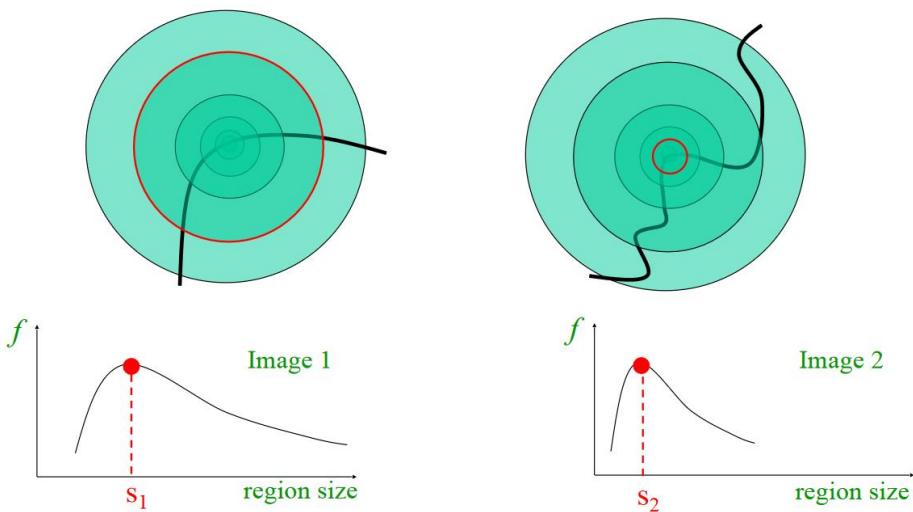


Harris Corner Detector **Do NOT Resilient For** •
.Image Scaling – Geometrical Scale ○



אלגוריתם Scale Invariant

אנחנו נרצה למצוא פונקציה f שלא תהיה תלולה ב-Scale-ב- f כלומר אם נגדיל או נקטין את התמונה הערך שנתקבל יהיה זהה. דוגמה לפונקציה כזאת היא הממוצע; לאוטו אובייקט יהיה את אותו ממוצע בלי קשר לגודל שבוא הוא צולם. לדוגמה: אם אתה מצלם בניין מ-10 מטר או 20 מטר, הממוצע על הבניין ישאר זהה. היות ואנחנו מעוניינים למצוא שינויים אנחנו בעצם מ Chapman פונקציה כזאת שתיתן את הערך המקסימלי תחת האזור שמרתיך כי טוב את הפינה.

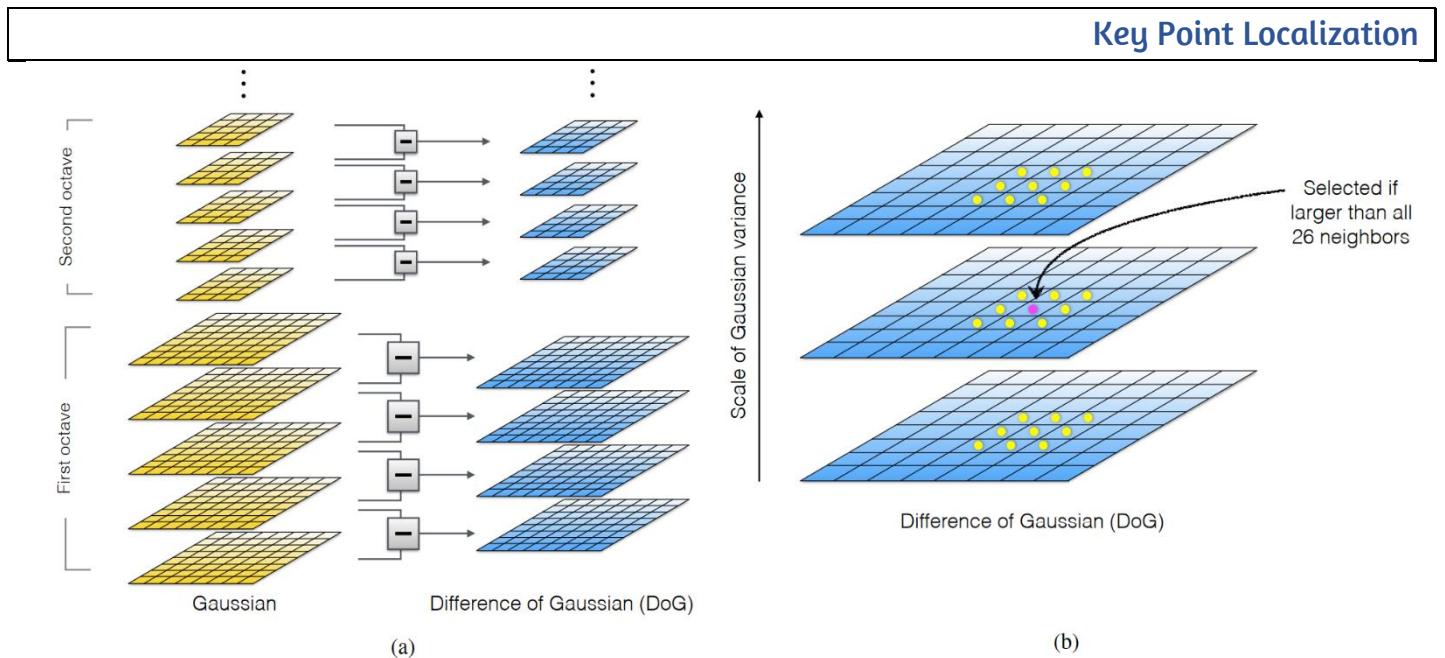


Lindeberg הראה (2) שחתת תנאים די כלליים שהKERNEL הגרפי היחיד לניטות של אנליזת קנה המידה (scale-space). אינטואטיבית, החלקה היא פעולה המדמה הקטנה של התמונה והגדלת בחרזה. שובייקט מוקטן בעצם כל פיקסל מייצג את הסביבה שלו וזה בדיק מה שנחנו עושים בהחלקה. כיוון שהמטרה שלנו למצוא פינות אנו ב עצמי נרצה את ה-Gaussian, ולכן נקבל את ה-LoG.

באופן מעשי משתמשים בדרך"כ ב-(Gaussian DoG)Difference of Gaussian (DoG) כקירוב ל-LoG, זאת ממשום שה-LoG מערב נגרות שנייה שהוא פעולה יתירה מחישור. כיצד, שני הfonקציות מוגדרות באופן הבא:

$$\text{LoG} := \sigma^2(G_{xx}(x, y, \sigma) - G_{yy}(x, y, \sigma))$$

$$\text{DoG} := G(x, y, k\sigma) - G(x, y, \sigma)$$

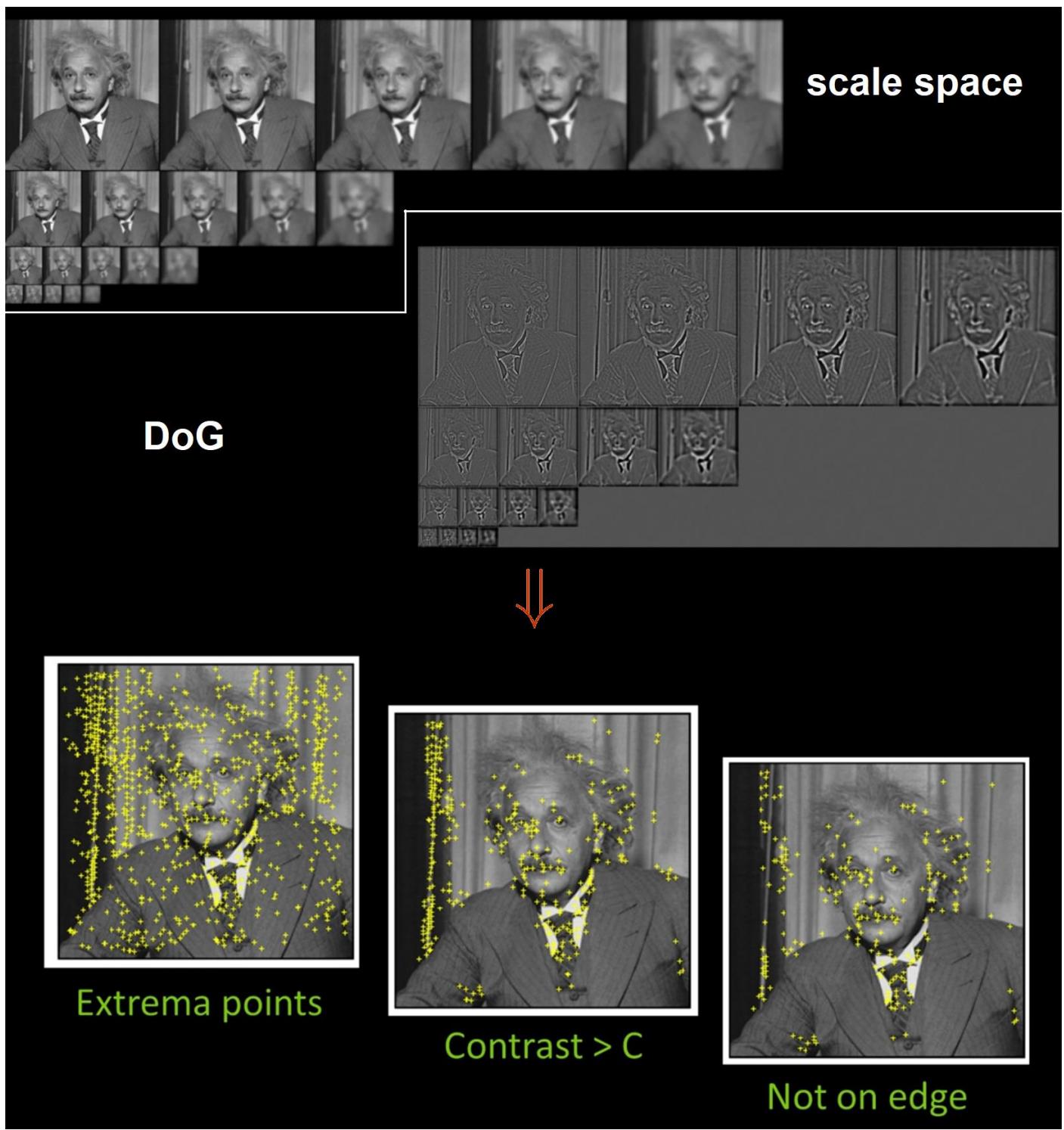


אייר 5. (a) ה-scale First scale הוא הרמה הראשונה של הפירמידה וה-second Scale היא הרמה השנייה וכן הלאה. בכל scale אנחנו מחשבים גaussians עם $\sigma_1 \dots \sigma_n$, כאשר σ הוא סטיית תקן ברמה ה-n. החישור בין שני רכומות עוקבות נoston את DoG (שכחושבת ביעילות). (b) פיקסל נבחר אם הוא גדול יותר מכל השכנים שלו בmäßig ה-Gaussian ובסיגור ה-Gaussian.

לאחר שביצענו את הפעולה לעיל אנו נבצע שני סינונים:

1. נקבע סני k , ומבחן הנקודות שנבחרו נשאיר רק את אלו שהערך שלהם (שזה ה-Contrast בתמונה) גדול מ- k .
2. נשמייט את כל ה-edge – ככלומר נמחק את כל הנקודות שהשינו שלהם מתבצע רק לאורך ציר אחד.

נראה שלבים אלו בדוגמה.



התאמת של פיצ'רים

התחלנו את הפרק במטרה לפתח אלגוריתם להתאמת של פיצ'רים. עד כה הגדרנו מה הם נקודות מעניינות וair למצוותם. ראיינו דרכיהם למצואו אותם שהם עקביהם תחת שינויים עצמאים של הפיקסלים בכיוון של הפיקסלים ואפילו בגודל של האובייקט (הפיצ'ר). כתעת אחרי שמצאנו פיצ'רים בשני התמונות נרצה להתאים אותם. לשם כך אנחנו נרצה להגדיר מתאר של הפיצ'ר (descriptor) שבעצם יהיה וקטור שייתאר את הפיצ'ר כך שפיצ'רים דומים יקבלו וקטוריים קרובים.

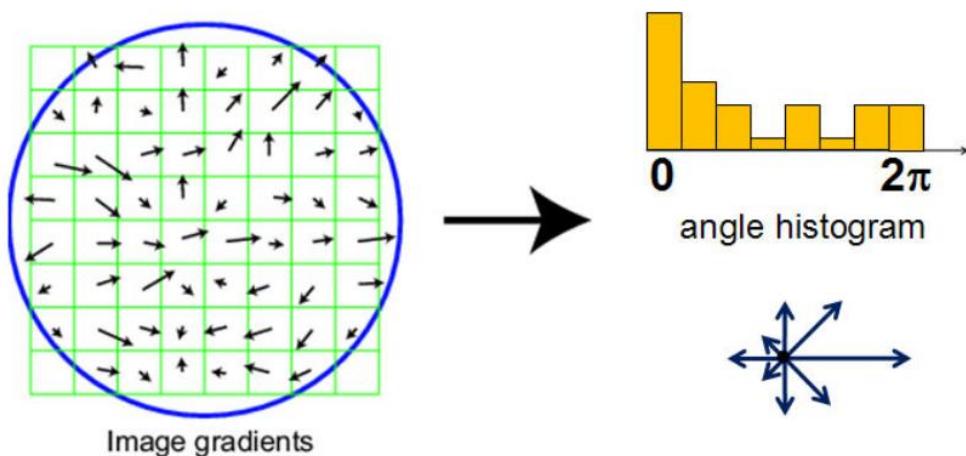
SIFT – Scale Invariant Feature Transform

ישנים 4 שלבים באlgorigthm, את שני השלבים הראשונים כבר הצגנו ונותר רק לדבר על השלישי והרביעי:

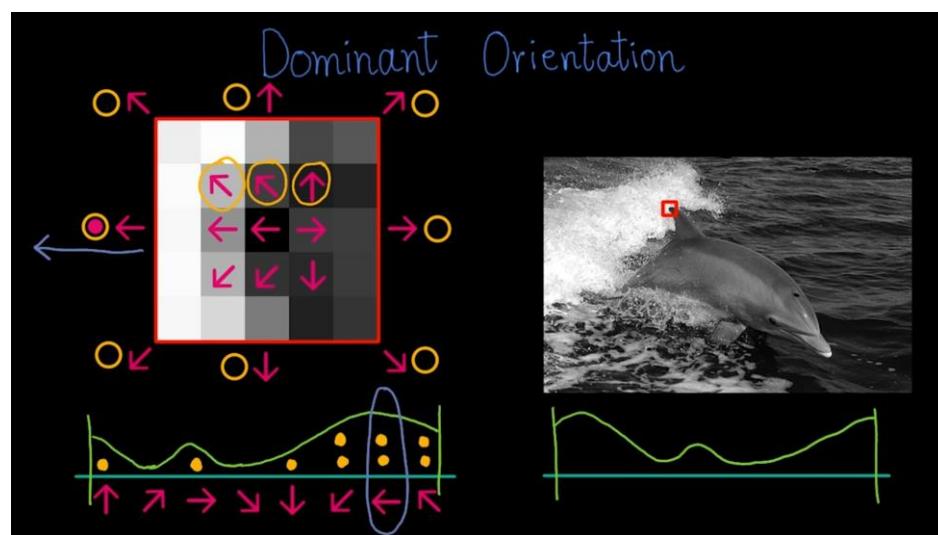
1. Multi-scale extrema detection
2. Keypoint localization
3. Orientation assignment
4. Keypoint descriptor

Orientation Assignment

על מנת שהמזהה שלנו יהיה עקבי תחת רוטציה נרצה ליזיר את הפיצ'רים. אנחנו נחשב את הגרדיינט המנורמל בכל פיקסל, נعتبر גאוסיאן כך שהמרכז יקבל את החשיבות הגבוהה ביותר ולאחר מכן נעשה חלוקה של חזויות ל-36 חלקים. ניקח חזית שהופיעה בחלק עם הכוי הרובה התאמות ונסתכל עליו Caino הוא הצפון החדש של הפיצ'ר. ככה פיצ'רים דומים יקבלו את אותו כיוון ובעצם יהיו מושרים בכיוון אחד.



דוגמה. נסתכל על הפיצ'ר שופיעה על הסנפיר של הדולפין, ונשקל את כיווני הגרדיינט בסביבה בגודל 3×3 , תוצאות הכוון מופיעות באIOR הבא.



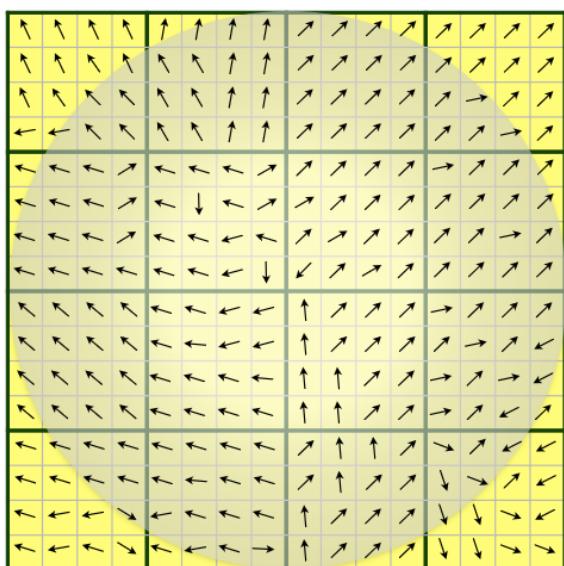
نبצע היסטוגרמה ונקבל את הגרף המתווך בחלק השמאלי התחתון. שנחליק את הגרף נקבל את הגרף הירוק ונוכל להסיק ש← הוא הכוון השולט.

Key-Points Descriptor

כל פיצ'ר הוא רכיב בגודל 16×16 ש切成 לחלקים שמרכיבים אותו בגודל 4×4 , סה"כ יש 16 חלקים. מעברים גאומטריים על ציוויל הגרדיינט של הפיצ'ר כך שהפיקסלים שקרובים יותר למרקם הפיצ'ר יקבלו חשיבות גבוהה יותר. בכל חלק מבצעים היטוגרממה על ציוויל הגרדיינט המשוקלים ל-8 תאים. סה"כ מקבלים וקטור בגודל $128 = 8 \times 16$ מימדים.

Image Gradients

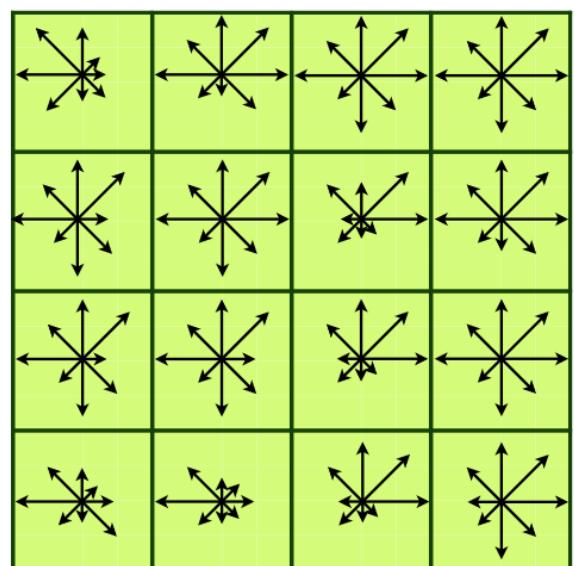
(4×4 pixel per cell, 4×4 cells)



Gaussian weighting
(sigma = half width)

SIFT descriptor

(16 cells \times 8 directions = 128 dims)



התאמת של פיצ'רים

יש מספר דרכים לבצע ההתאמת של פיצ'רים:

1. חיפוש שכן קרוב ביותר. הבועה היא שביצוע של ההתאמת לעבור על כל הקומבינציות של זוגות.
2. שיטה שנקראה Best-Bin-First – שוחזגה ע"י Beis & Lowe ב-97. זהה ווריאציה של עצי-d-k שמאפשרת חיפוש שכן קרוב בזמן של $O(\log n)$. התוצאה מתאפשרת בפקטור של 1000 – 100 מהמהירות של חיפוש שכן קרוב אבל לבדוק של כ-95%. מכובן שזה מספיק טוב כי גם במקרה לא צריכים הרבה ההתאמות.

3. Haar "Wavelet-based hashing" – חשב מתאר קצר יותר מהסביבה באמצעות שימוש ב-"Wavelet".

4. Locally Sensitive Hashing – נרצה לבנות פונקציית Hash כך שההסתברות ששתי נקודות קרובות יפללו באותו מצב/אזור/חלק תהיה גבוהה וההסתברות שתשתי נקודות רחוקות יפללו באותו אזור תהיה נמוכה. גישה זאת הוצגה ע"י Kulis & Grauman הוצג ב-2009, ראה (3).

.5

Algorithm 3 RANSAC - Random Sample Consensus

```

1: procedure RANSAC( $\mathcal{I}, N$ )
2:   Let  $s$  be the minimal number of minimal matches need to compute the model.
3:   for  $i = 1, \dots, N$  do
4:     Select uniformly at random  $s$  matches.
5:     Compute the model.
6:     Get consensus set  $C_i$ .
7:   end for
8:   return  $C_i$  with maximum size.
9: end procedure

```

- 1.** s – number of points to compute solution
- 2.** p – probability of success
- 3.** e – proportion outliers, so % inliers = $(1 - e)$
- 4.** $P(\text{sample set with all inliers}) = (1 - e)^s$
- 5.** $P(\text{sample set will have at least one outlier}) =$
$$(1 - (1 - e)^s)$$
- 6.** $P(\text{all } N \text{ samples have outlier}) = (1 - (1 - e)^s)^N$
- 7.** We want $P(\text{all } N \text{ samples have outlier}) < (1 - p)$
- 8.** So $(1 - (1 - e)^s)^N < (1 - p)$

| 17 |

$$N > \frac{\log(1 - p)}{\log(1 - (1 - e)^s)}$$

Algorithm 4 RANSAC For Estimating Homography

```

1: procedure RANSAC FOR HOMOGRAPHY( $\mathcal{I}, N, \mathcal{S}, \varepsilon$ )
2:    $\mathcal{H} = \emptyset$ 
3:   for  $i = 1, \dots, N$  do
4:     Select uniformly at random 4 matches.
5:     Compute the Homography matrix  $H$ .
6:     if  $\text{SSD}_{p \in \mathcal{S}}(p', Hp) < \varepsilon$  then
7:        $\mathcal{H} \leftarrow \mathcal{H} \cup \{H\}$ 
8:     end if
9:   Let  $\mathcal{H}_{\max}$  be the homography  $H \in \mathcal{H}$  with largest set of inliers.
10:  Recompute least-square  $\mathcal{H}_{\text{final}}$  estimate on all of the inliers.
11: end for
12: return  $C_i$  with maximum size.
13: end procedure

```

Algorithm 5 Adaptive RANSAC For Estimating Homography

```

1: procedure FIND  $N$  ADAPTIVE APPROACH( $\mathcal{I}, \mathcal{S}$ )
2:    $N \leftarrow \infty$ , sample_counter  $\leftarrow 0$ ,  $e \leftarrow 1$ 
3:   while  $N > \text{sample\_counter}$  do
4:     Pick 4 matches and count the number of inliers  $I$ .
5:      $e' \leftarrow 1 - I/|\mathcal{S}|$ .
6:     if  $e' < e$  then
7:        $e \leftarrow e'$ 
8:        $N \leftarrow \frac{\log(1-p)}{\log(1-(1-e)^s)}$ 
9:     end if
10:    sample_counter  $\leftarrow \text{sample\_counter} + 1$ 
11:  end while
12:  return  $N$ .
13: end procedure

```

יתרונות וחסרונות של RANSAC

• יתרונות:

- עמיד למספר רב של outliers.outliers.
- קל לבחור את הפרמטרים בניגוד ל-Hough transform.
- ישומי למספר רב יותר של פרמטרים לעומת הומוגניטם.Hough transform

• חסרונות:

- סיבוכיות הזמן גדלה במהירות עם מספר הפרמטרים של המודל.s.
- מתקשה להתמודד עם מספר של התאמות, לדוגמה אתה מנסה להתאים מספר מושתחים.
- המודל חייב לתאר בצורה מהימנה את הממציאות.

Object Recognition

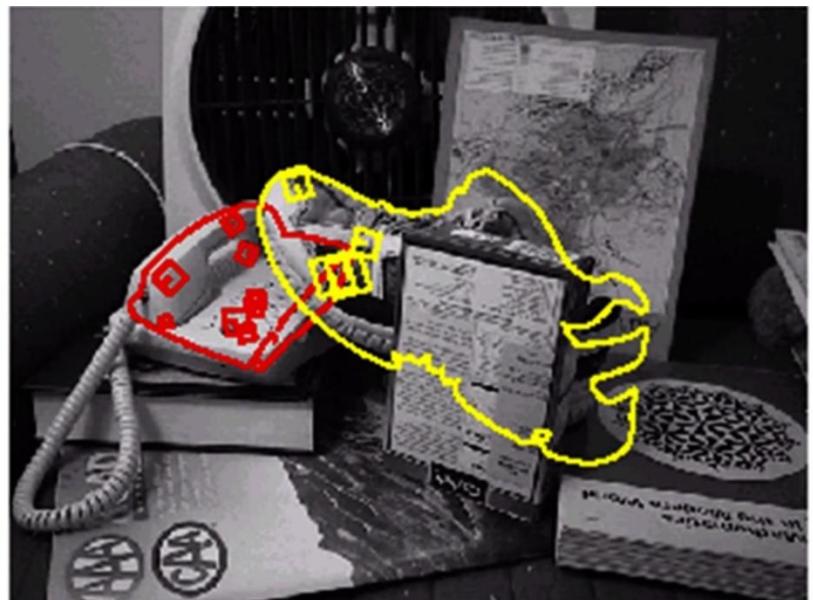
בזמן האימון אנחנו ניצור תכונות שביהם האובייקטים אותם נרצה לזהות מבודדים, ככלומר יש את האובייקט בלבד בתמונה. נ賴ץ על SIFT detector על מנת לזהות נקודות מעניינות וכל זה יבוצע בשלב ה-training.

בזמן ה-test ניקח את key points מהתמונה שעלייה מרכיבים את ה-test ונשווה אותם מול כל ה-key

points באובייקט שאותו אנחנו מנסים לזהות בתמונה. כעת נמצא את ה-affine transformation המתאימה לאובייקט (כזכור, ל-affine transformation יש 6 משתנים لكن אנחנו צריכים למצוא 3 התאמות, כי כל נקודה יש לה שני רכיבים y, x).



Training



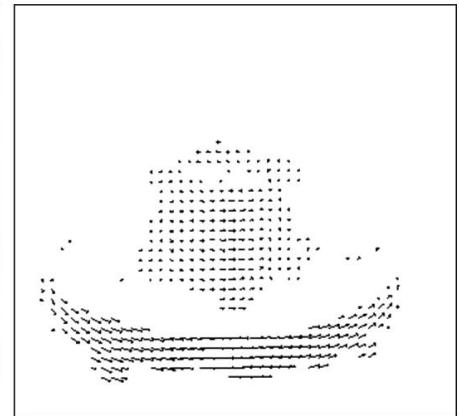
Testing

OPTICAL FLOW

הגדולה (זרימה אופטית).

זרימה אופטית היא הזרימה הנראית של אובייקטים או משטחים.

הערה. שדה הזרימה הוא הזרימה הכללית (הנראית והבלתי-נראית).



הבעיה שנפתרה בחלק זה היא כיצד לשערר את התנועה האופטית, לשם כך נניח מספר הנחות:

- **COLOR CONSTANCY.** A feature remains in the same colour.
- **BRIGHTNESS CONSTANCY.** The brightness of the feature preserves.
- **SMALL MOTION.** Points do not move too far, i.e., more than a pixel.

:
בזכות ההנחה השנייה נוכל להשתמש בפיתוח טיילור (TAYLOR'S EXPANSION)
 $I(x + \Delta x, y + \Delta y) \approx I(x, y) + I_x \cdot \Delta x + I_y \cdot \Delta y$

אנחנו נרצה למזער את ה-SSD :

$$\begin{aligned} \mathcal{E}(\Delta x, \Delta y) &:= \sum_{x,y \in \Omega} (I_2(x + \Delta x, y + \Delta y) - I_1(x, y))^2 \\ &= \sum_{x,y \in \Omega} (I_2(x, y) + I_x \cdot \Delta x + I_y \cdot \Delta y - I_1(x, y))^2 \quad (\text{Taylor's Expansion}) \\ &= \sum_{x,y \in \Omega} ([I_2(x, y) - I_1(x, y)] + I_x \cdot \Delta x + I_y \cdot \Delta y)^2 \\ &= \sum_{x,y \in \Omega} (I_x \cdot \Delta x + I_y \cdot \Delta y + I_t)^2 \end{aligned}$$

נגזר לפיא Δx ולפי Δy ונקבל:

$$\frac{\partial \mathcal{E}}{\partial (\Delta x)} = 0 \implies -2 \sum_{x,y \in \Omega} I_x \cdot (I_x \cdot \Delta x + I_y \cdot \Delta y + I_t) = 0$$

$$\frac{\partial \mathcal{E}}{\partial (\Delta y)} = 0 \implies -2 \sum_{x,y \in \Omega} I_y \cdot (I_x \cdot \Delta x + I_y \cdot \Delta y + I_t) = 0$$

בכתב מטריציוני נקבל

$$\begin{pmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_y I_x & \sum I_y^2 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} \sum I_x I_t \\ \sum I_y I_t \end{pmatrix}$$

נוסאים חשובים שלא דיברתי עליהם

Normalized Cross-Correlation •

- שיטה זאת מאפשרת למצוא התאמות של תבניות מסוימות בתמונה ע"י לקיחת הערך המקסימלי המתאים מהערכת הפילטר של התבנית על כל התמונה.
- השיטה **invariant** (לא עקבית) ל**רטציה** ול**Scaling** של התבונה.
- מוחשיים u, v כך $\mathcal{E}(u, v) := \sum_{x,y \in \Omega} (I_2[x+u, y+v] - I_1[x, y])^2$ קטן ככל הניתן.

מתקיים:

$$\begin{aligned}\mathcal{E}(u, v) &:= \sum_{x,y \in \Omega} (I_2[x+u, y+v] - I_1[x, y])^2 \\ &= \sum_{x,y \in \Omega} I_1^2(x, y) - 2I_1(x, y) \cdot I_2(x+u, y+v) + I_2^2(x+u, y+v)\end{aligned}$$

נשים לב שהביטוי הראשון קבוע וכן הביטוי האמצעי גדול משני הביטויים השניים ע"פ א-שוויון קושי שוורץ:

$$\begin{aligned}\sum_{x,y \in \Omega} I_1(x, y) \cdot I_2(x+u, y+v) &\leq \left(\sum_{x,y \in \Omega} I_1(x, y) \right)^2 \cdot \left(\sum_{x,y \in \Omega} I_2(x+u, y+v) \right)^2 \\ &\leq \sum_{x,y \in \Omega} (I_1(x, y))^2 \cdot \sum_{x,y \in \Omega} (I_2(x+u, y+v))^2\end{aligned}$$

לכן נגדיר את הקורלציה:

$$\mathfrak{C}_{I_1, I_2}(u, v) := \mathfrak{C}(u, v) := \sum_{x,y \in \Omega} I_1(x, y) \cdot I_2(x+u, y+v)$$

ונשים לב שאנו רוצים למקסם את הקורלציה.

- הקורלציה המנורמלת נשמרת תחת הכפלת העוצמה בקבוע וחתה הוספה של קבועה

לעוצמה כלומר אם נגדיר $b = I_2^{(\star)} \cdot a$ עבור $\mathbb{R} \in a, b$ נקבל

$$\arg \max N\mathfrak{C}_{I_1, I_2}(u, v) = \arg \max N\mathfrak{C}_{I_1, I_2^{(\star)}}(u, v)$$

בהינתן משתנה מקרי X נגדיר את המשתנה המנורמל

$$X := \frac{X - \mathbb{E}(X)}{\sigma_X} = \frac{X - \mathbb{E}(X)}{\sqrt{(X - \mathbb{E}(X))^2}}$$

אנחנו נגדיר את הקורלציה המנורמלת באמצעות הקורלציה על התמונות המנורמלות כאשר כל איבר בתמונה נבחר באופן אחיד. כלומר:

$$N\mathfrak{C}_{I_1, I_2}(u, v) = \mathfrak{C}_{\hat{I}_1, \hat{I}_2}(u, v) := \sum_{x,y \in \Omega} \frac{I_1(x, y) - \mathbb{E}(I_1)}{\sqrt{(I_1(x, y) - \mathbb{E}(I_1))^2}} \cdot \frac{I_2(x, y) - \mathbb{E}(I_2)}{\sqrt{(I_2(x, y) - \mathbb{E}(I_2))^2}}$$

- נזכיר שנית שהקורלציה המנורמלת נשמרת תחת הכפלת העוצמה בסקלר והוספה סקלר לעוצמה. ונזכיר שאנו מוחשיים את הערך שמקסם את הקורלציה המנורמלת.

גסאודו קוד NCC, template matching, עם

קלט: מטריצה דו ממדית של תמונה NxN, תבנית דו ממדית KxK.

פלט: מיקום התבנית בתמונה

אלגוריתם:

- עבור כל חלון KxK אפשרי בתמונה- חשב את ה NCC בין התבנית לחלק בתמונה
- הערך הגבוה ביותר הוא מיקום התבנית המשוער
- החזר את מיקום התבנית המשוער

- ניתן להכליל שיטה זאת למצוא לא רק Translation אלא גם Rotation עם עוד פרמטר של זווית, ואילו ניתן למצוא את ה-Template תחת הטלה אפינית זהה יתן עם 6 פרמטרים.
- הבעה היא זמן הריצה:

Type Of Transformation	Parameters	Runtime Complexity
Translation	<ul style="list-style-type: none"> • Δx • Δy 	$O(N^2)$
Translation + Rotation	<ul style="list-style-type: none"> • Δx • Δy • Θ – Angle 	$O(N^3)$
Affine Transformation	<ul style="list-style-type: none"> • 6 – parameters 	$O(N^6)$

Bilateral filter •

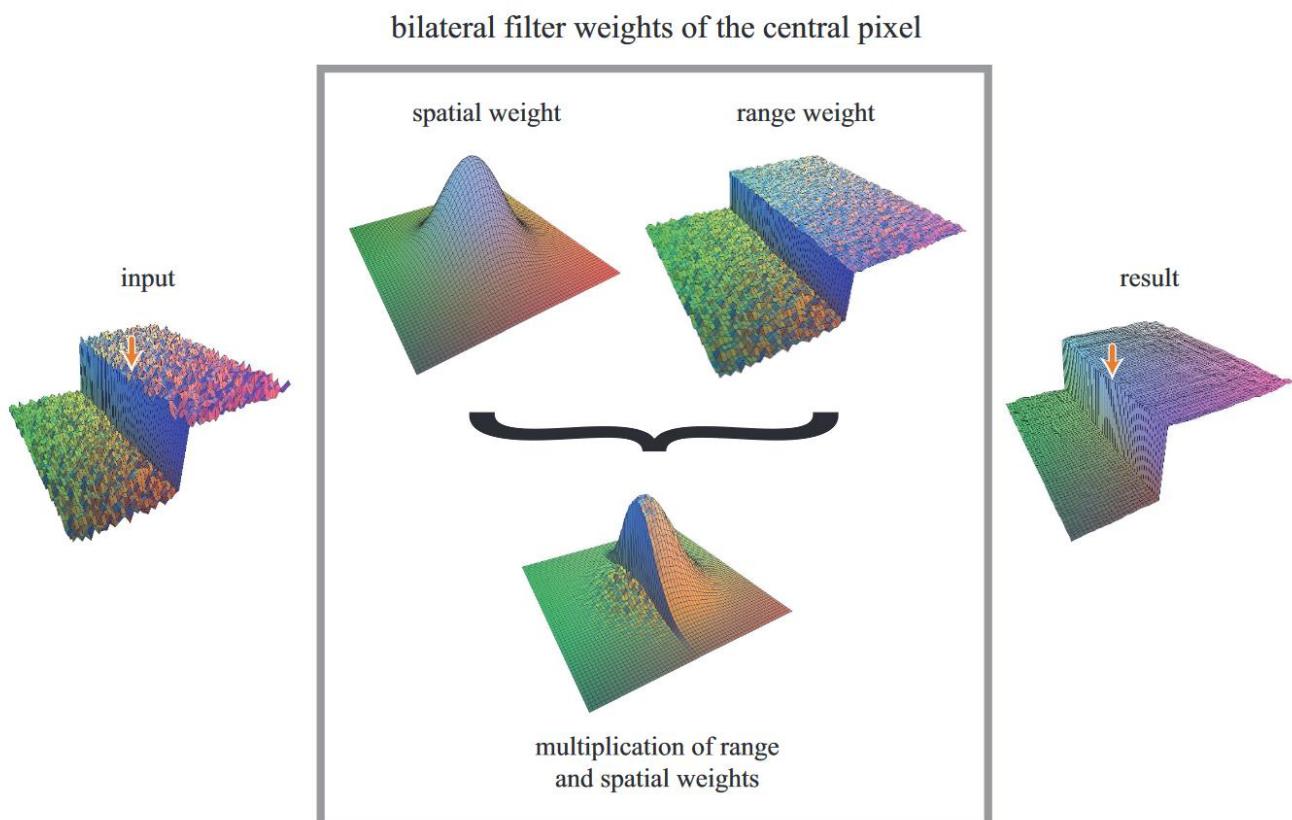
- זהו פילטר שמטרתו להחליק את התמונה אבל יחד עם זאת לשמר על השפות של התמונה, לשם כך הוא מביא בחשבון גם את המרחק האוקלידי בין הפיקסלים וגם את הפרשי העוצמה.

$$BF[I]_p = \frac{1}{W_p} \sum_{q \in S} G_{\sigma_s}(\|p - q\|) G_{\sigma_r}(|I_p - I_q|) I_q$$

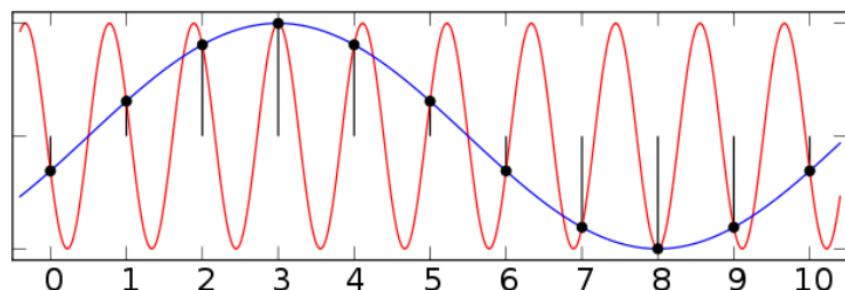
↓ ↓ ↓
 Normalization Factor Space Weight Range Weight

$$W_p = \sum_{q \in S} G_{\sigma_s}(\|p - q\|) G_{\sigma_r}(I_p - I_q)$$

באיור מתרגם תוצאת הפעלה של הפילטר:



Aliasing Signals "traveling in disguise" as other frequencies.



• נגזרות – Derivative

צריך לזכור שבנושא הנגזרות יש Trade-Off בין הлокאליות של ה-edges לבין מידת ההחלהה שבא נשתמש. ככל שנחליק יותר הישרים יהיו עבים יותר, וככל שנחליק פחות נקבל יותר רעש.

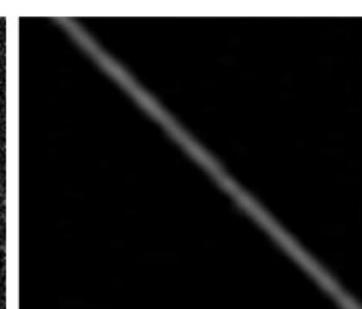
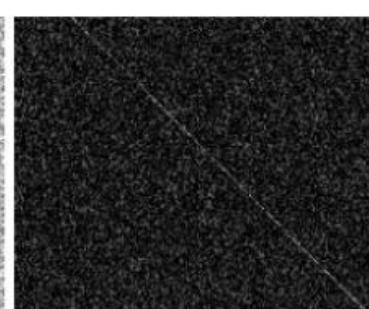
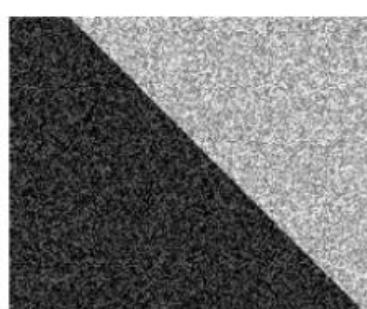
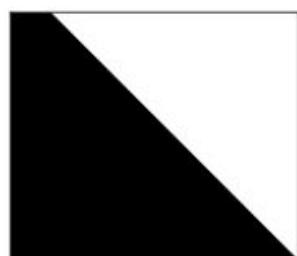
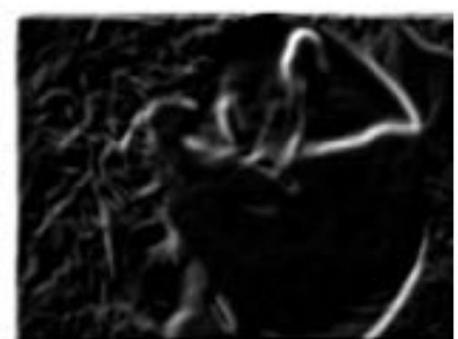
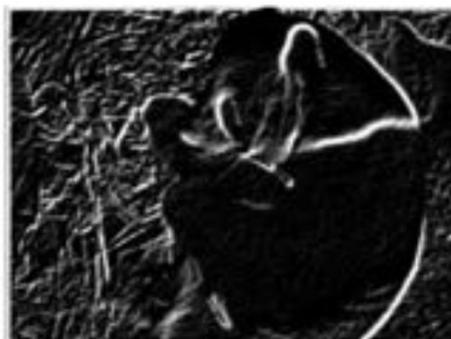


Image + Noise

Derivatives detect edge and noise

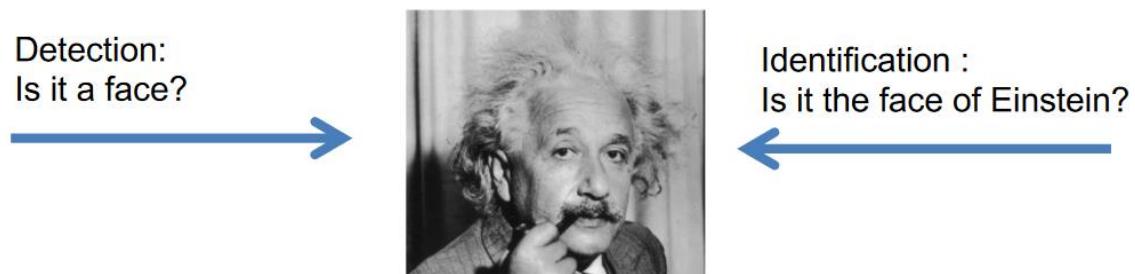
Smoothed derivative removes noise, but blurs edge

 $\sigma = 1 \text{ pixel}$ $\sigma = 3 \text{ pixels}$

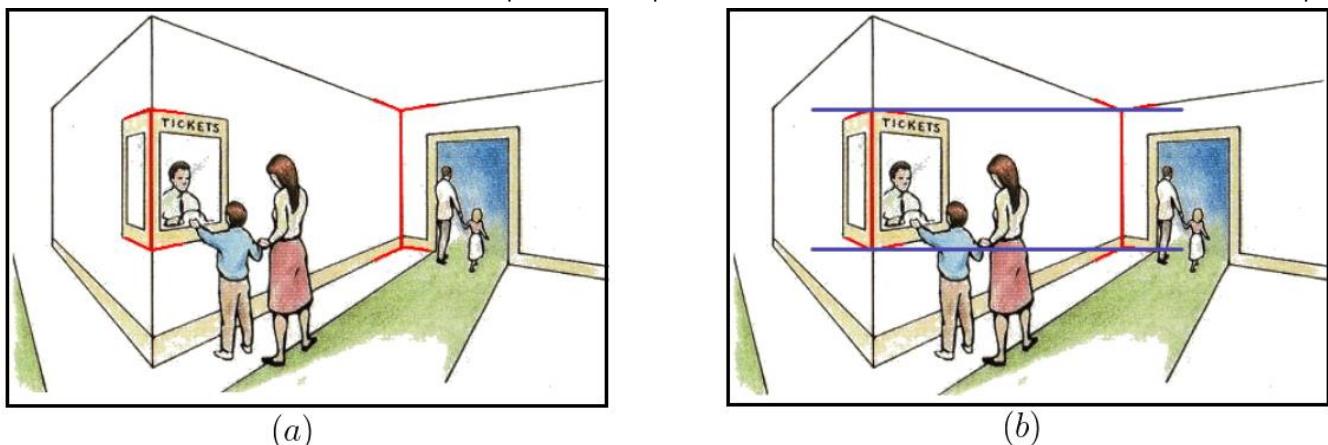
.....	INPUT
.....	MEDIAN
.....	MEAN

- חציון הוא Edge-Preserving filter ב Gegod למכמצע (box-filter) ולגאומיאן:

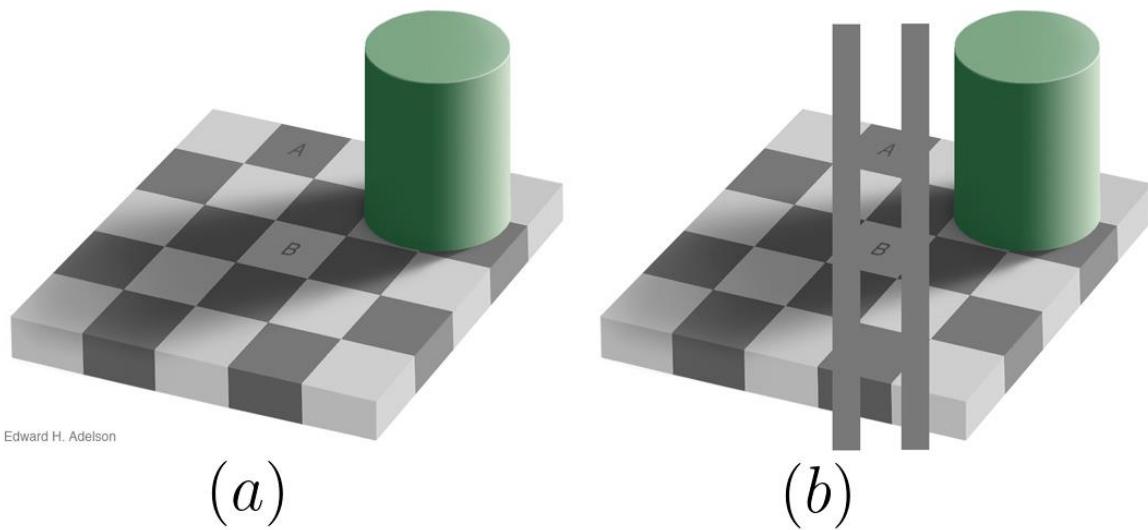
- ההבדל בין ראייה ממוחשבת לעיבוד תמונה הוא שבuibוד תמונה אנחנו עושים מניפולציות על התמונה תוך כדי שימוש בידע מתוכן התמונה ולא רק מעצם היotaה התמונה אלא ממש מנתחים מה יש בפנים ועל פי זה פועלם. הפעולות בראייה ממוחשבת לא מתעסקות בהסקת וניתוח של התמונה ושל הרכיבים בתמונה.
- Recognition vs Identification – ב-Recognition נרצה לזהות פנים באופן כללי, אולם ב-Identification נרצה לזהות פנים של בן אדם ספציפי.



- יתכנו אשליות נוספות שיווצרו מנסיבות שונות, לדוגמה בתמונה הבאה ניתן לראות אשליה שנייה: הקווים האדומים באורכים שונים, אולם הם באותו אורך כפי שניתן לראות בתמונה הימנית:



אשליות נוספות יכולות להיווצר כתוצאה מ-shadawing כמתואר באיר הבא:

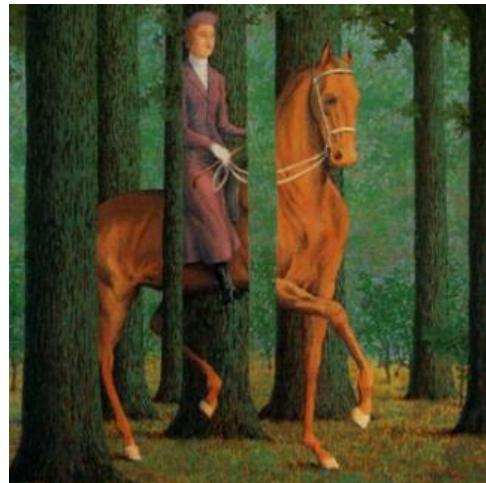


סבירות אלו ועוד מקרים علينا להבין את התמונה וממחישים מדוע להבין תמונה זה קשה.

דוגמאות לסיבות נוספות:

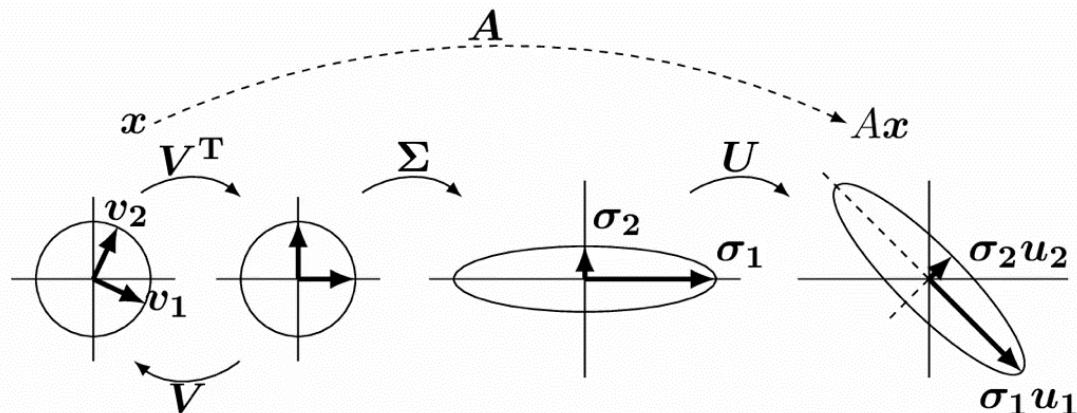
- נקודת מבט – האובייקט אותו אובייקט אומנם במצבה זה נראה שונה.

- o הירה – קמות האור הtmpונה משנה את הפיקסלים אבל הtmpונה אותה tmpונה.
- o ראיינו ב-SIFT – Scale
- o חוסרים בתמונה – Occlusion



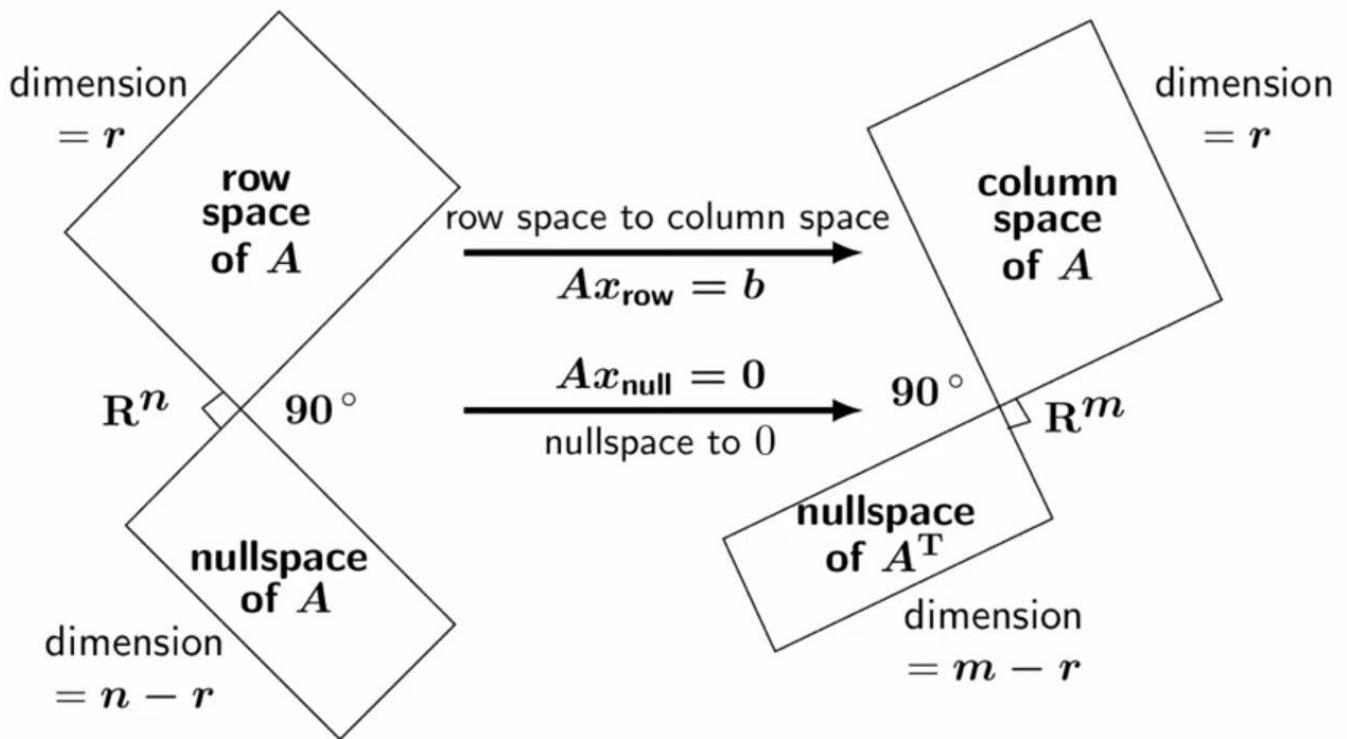
- o רכע, תנועה ועוד.

- Linear Algebra – Abridged
 - Singular Value Decomposition – SVD



U and V are rotations and possible reflections. Σ stretches circle to ellipse.

- The Four Fundamental Subspaces



This is the Big Picture—two subspaces in \mathbb{R}^n and two subspaces in \mathbb{R}^m .

From row space to column space, A is invertible.

:image stitching 9.3

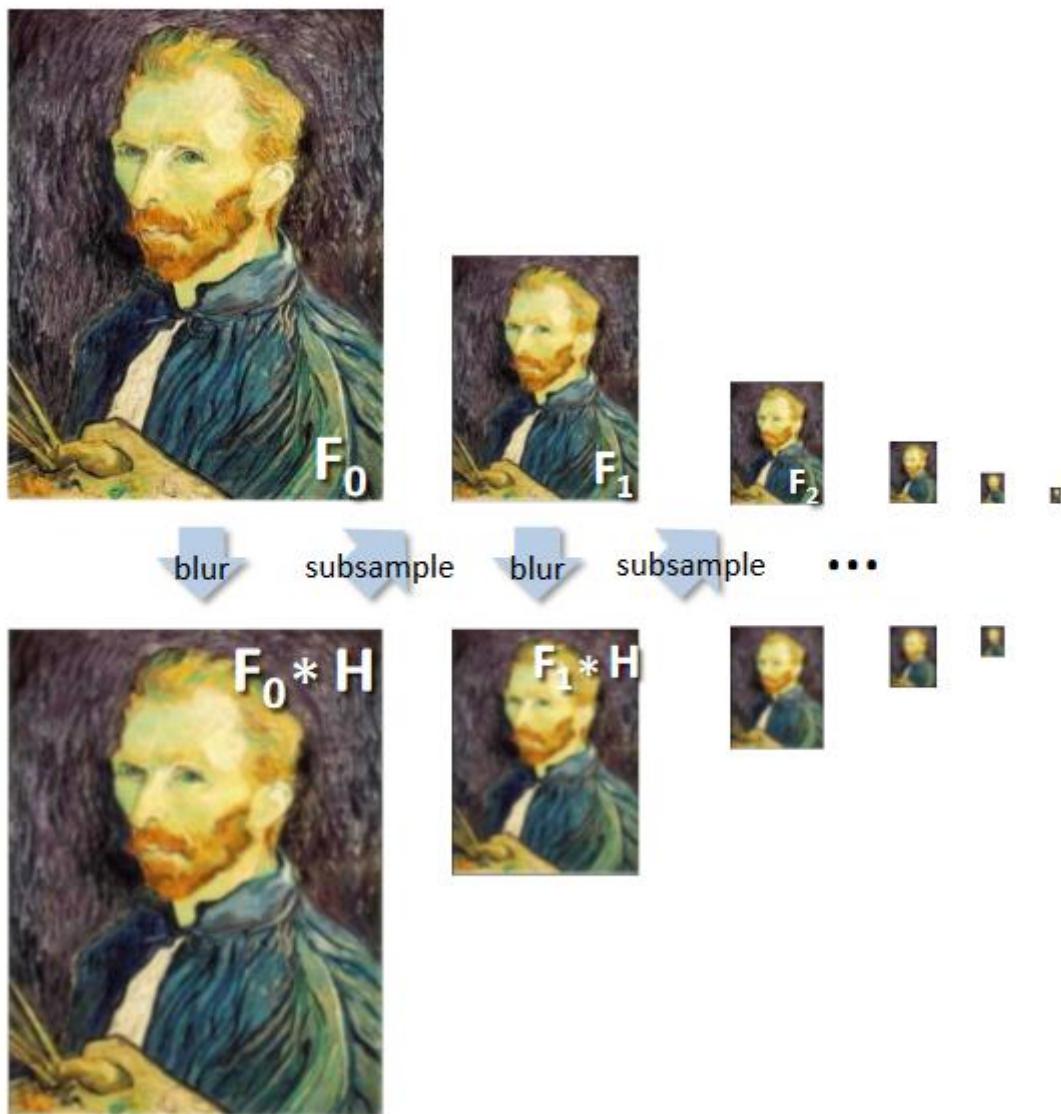
כעת לאחר שיש בידינו מספיק כלים, ניתן לתכנן אלגוריתם לחיבור תמונות (פנורמה).

מה הבעיה בלחבר תמונות? הבעיה שהתמונה הנו אמורים שונים בעולם האמיתי. צריך לדעת איזה אזורים חופפים בין התמונות במציאות - כדי לדעת איך לחבר אותן. בנוסף, צריך לעמוד כל פיקסל חוף איזה פיקסל הוא מייצג בתמונה השניה.

סדר פעולות בסיסי בהדבקת תמונות :

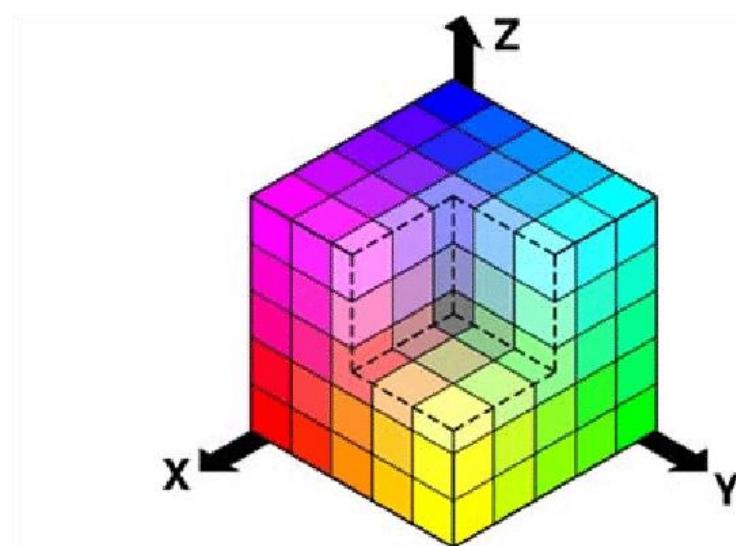
1. לקיוח סדרת תמונות מאותה פוזיציה (אולי מזiosis את המצלמה על אותו קו אופקי)
2. מחשבים טרנספורמציה דו ממדית בין התמונה השנייה לראשונה - הומוגרפיה, מעשה כך יודעים את היחסים של כל שני פיקסלים שבשתי התמונות
3. עטיפים (warp) את התמונה השנייה כדי שתחפוף עם התמונה הראשונה. (להפוך את שתי התמונות לאותו מרחב קוואורדינטוט)
4. לסייעים - כדי שהחיתוך בין שתי התמונות לא יהיה גס- עושים ערבול (blending)
5. חוזרים כך עבר כל סדרת התמונות

- Image Pyramids – Gaussian Pyramid:



- Color Spaces

- RGB – Problem. If you move a little you change the entire color



YI

QYI הוא מרחב הצבעים המשמש את מערכת הטלוויזיה הצבעונית NTSC, אשר בשימוש בעיקר בצפון ומרכז אמריקה, וביפן.

הערוץ הראשון (Y) הוא ערוץ העוצמה (סולם הזרה) (luminance scale).

שלא כמו ה-RGB, שם הבהירות מפוצלת בין שלושת הערוצים.

The other two contain the chrominance information.

Orange <-> Blue I:

Purple <-> Green Q:

ה-I וה-Q הם אורתוגונליים ובכך משתרעים על הצבעים האפשריים. מערכת הראייה האנושית רגישה הרבה יותר לשינויים בציר ה-I מאשר בציר ה-Q,

ומאפשרת להעביר את ציר ה-Q בפחות נאמנות, וכך לשמר על רוחב הפס.

QYI שימש לשידור בטלוויזיות ישנות בשחור לבן וטלוויזיות צבעוניות חדשות בו זמנית, הישנות פשוט השתמשו בערוץ Y, בעוד שהחדשנות שילבו את השניים האחרים כדי לקבל תצוגת צבע.

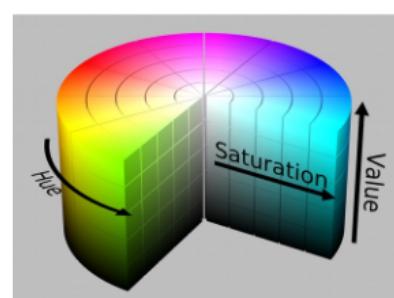
HSV

מרחב צבע נוסף הוא HSV, הפרמטרים שבו הם:

Hue – Color, Range $[0, 360)$ – גוון

Saturation – Strength of the color, $[0, 1]$ – רוחיה

Value – Brightness, $[0, 1]$ – ערך בהירות



HSV מאוד שימושי כשאנו צריכים לקבל פיקסלים בטווח צבע

מסויים, שלא כמו RGB או YIQ ואחרים, בהם הצבע נמצא

בספקטרום רציף.



ביבליוגרפיה – Bibliography

1. What is a Convolution? [Online] LeiosOS. <https://youtu.be/8fm8wXa2x5k?t=89>.
2. *Scale-Space Theory: A Basic Tool for Analysing Structures at Different Scales*. Lindeberg, Tony. 1994, Journal of Applied Statistics, Vol. 21, pp. 224-270.
3. *Kernelized locality-sensitive hashing for scalable image search*. Brian Kulis, Kristen Grauman. s.l. : IEEE, 2009, pp. 2130-2137.
4. Rafael C. Gonzalez, Richard E. Woods. *Digital Image Processing*. s.l. : Addison-Wesley, 2007.
5. Wilhelm Burger, Mark J. Burge. *Digital Image Processing*. s.l. : Springer, 2016.
6. McAndrew, Alasdair. *A Computational Introduction to Digital Image Processing*. s.l. : Chapman and Hall/CRC, 2015.
7. Maria M. P. Petrou, Costas Petrou. *Image Processing: The Fundamentals*. s.l. : Wiley, 2010.
8. Tyagi, Vipin. *Understanding Digital Image Processing*. s.l. : CRC Press, 2018.
9. Jain, Anil K. *Fundamentals of digital image processing*. s.l. : Prentice Hall, 1989.
10. Gradient Vector Visualization. [Online] <https://www.geogebra.org/m/sWsGNs86>.
11. Ai Shack. [Online] <https://aishack.in/category/computer-vision/>.
12. David A. Forsyth, Jean Ponce. *Computer Vision: A Modern Approach*. 2nd Edition. s.l. : Pearson Education, 2012.
13. 16-385 Computer Vision. *CMU Slides*. [Online] <http://www.cs.cmu.edu/~16385/>.
14. Computer Vision Lectures. *Udacity*. [Online]
<https://classroom.udacity.com/courses/ud810>.
15. Szeliski, Richard. *Computer Vision: Algorithms and Applications*. s.l. : Springer, 2011.
16. *Object recognition from local scale-invariant features*. Lowe, David G. 1999, Proceedings of the Seventh IEEE International Conference on Computer Vision, Vol. 2, pp. 1150-1157.

<http://www.cs.toronto.edu/~fidler/teaching/2019Fall/CSC420.html>

<http://www.cs.toronto.edu/~fidler/slides/2019/CSC420/lecture8.pdf>

<https://www.youtube.com/watch?v=fVJeJMWZcq8&list=PL1gez-2dMcljWfR3TIRkubRcqWfbsH6aN&index=9&t=1155s>