

נושא:

Fast image dehazing using guided joint bilateral filter

Recent Advances in Image Dehazing

מרצה:

אמיר הנדלמן

מגיש:

רוסלאן אוסמןוב - 327480026

Ruslan31bar@gmail.com

Git : [LINK](#)

(ראה אופן שימוש עמ' 12) [GUI Fast image Dehazing](#)

(ראה אופן שימוש עמ' 18) [GUI Homomorphic Dehazing](#)

תוכן עניינים

Contents

3	Fast image dehazing using guided joint bilateral filter.
3	מבוא
3	רקע כללי
4	הסרת ערפל על בסיס סינון
5	אלגוריתם
5	Guided joint bilateral filter.
7	הערכת אור אטמוספירה
7	Scene radiance recovering
7	תוספת של (Adaptive Histogram Equalization
8	תוצאות
12	GUI - אופן שימוש
13	Recent Advances in Image Dehazing
13	מבוא
13	Homomorphic filtering
16	תוצאות השוונות בין שני שיטות לניקוי אובי:
18	מסקנות:
18	GUI - אופן שימוש
19	Appendix A
28	Appendix B

Fast image dehazing using guided joint bilateral filter.

מבוא

תופעת אובך או ערפל הנגרמת על ידי חלקיקים מרחפים באוויר, מה שmphcit את הראות ביישומי ראייה ממוחשבת. שיטות שונות הוצעו להסרת אובך מתמונות, אך עדין נותרו אתגרים בגלגול אופי הבעיה. שיטות מסוימות מידיע עומק מסוימן במספר תמונות או משתמשות במידע נוסף, אך הן אין מתאימות לישומים בזמן אמיתי. מחקרים אחרים התמקדו בהסרת אובך מתמונה בודדת, תוך שימוש בטכניות כמו מיקסום הנגדיות המקומית, הערכת זוהר הסצנה ויישום ה-(DCP). עם זאת, לשיטות אלו עשויה להיות מוגבלות, כגון הפקת צבעים רווים מדי או דרישת לתהליכי אינטנסיביים מבחינה חיונית. כדי להתמודד עם חששות הייעילות, חוקרים הציעו טכניקות שחרור מואצות, כגון מסננים מונחים ואלגוריתמים מהירים המשמשים במסננים ציוניים. בשיטה המוצגת, מוצע אלגוריתם חדש להסרת ערפל בתמונה אחת, אשר משוחרר צעיף אווירה מדויק יותר על ידי מידע עמוק סצנה. האלגוריתם משתמש במידע קצר מתמונות הקלט כדי לסנן את צעיף האטמוספירה הראשוני, וכתוכאה מכח תוצאות משופרות של חוסר ערוב כאשר העומק משתנה באופן פתאומי.

רקע כללי

בראייה ממוחשבת, היוצרות תמונות אובי מותוארת בדרך כלל על ידי תהליך הנחתת האטמוספירה:

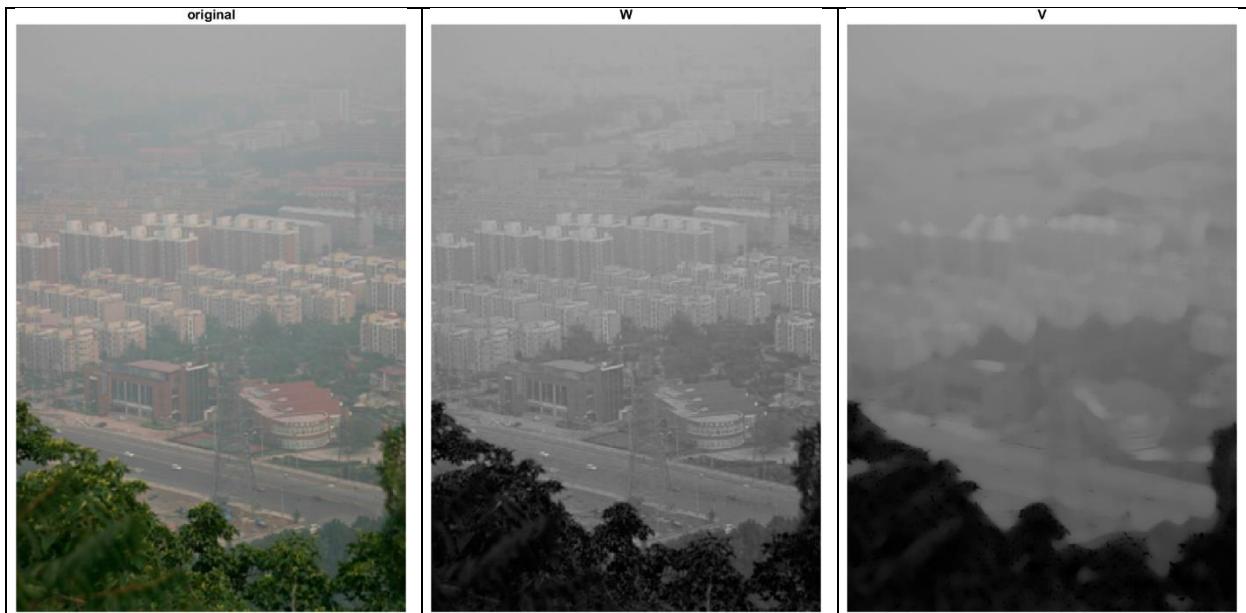
$$I(x) = J(x)t(x) + A(1 - t(x))$$

כאשר (x) הוא תמונה אובי, (x) היא התמונה נטולת אובי, A הוא אור האטמוספירה הכלול, ו- $t(x)$ הוא תוווד, כאשר האטמוספירה הומוגנית, $(x) = \exp(-\beta d)$. כאן, β הוא מוקדם הפיזור של האטמוספירה, ו- d הוא עומק הסצנה. המטרה של הסרת אובי היא לשחרר את A , (x) ו- $t(x)$. המונח (x) הוא הנחתה ישירה, מה שמצויב על כך שעורפל יגרום לו Zahar הסצנה להיחלש באופן אקספוננציאלי עם עומק הסצנה d בתוווד. המונח $(x) - 1$ הוא הצעיף האטמוספרי (פיזור או אטמוספרי), הגורם לטשטוש, שינוי צבע ועיוות בסצנה.

SHIPOR נראות התמונה על ידי מיקסום הנגדיות של התמונה המתבקשת, וניסוח מחדש של הבעיה כמיקסום צעיף האווירה $(x) = A(1 - t(x))$ בהנחה ש- $(x) = V$ חלק רוב הזמן, כמעט לאורך הקצוות עם קפיצות עמוק גדולות. מכיוון שפונקציית האופטימיזציה הייתה מסובכת מבחינה חישובית, הצעו אלגוריתם שחוור נראות מהיר על ידי שימוש בגישה סינון לחישוב צעיף האטמוספירה (x) . הם הניחו ש- $(x) = V$ רצוי צורך לעמוד בשני האילוצים הבאים: (1) הערך $(x) = V$ חיובי ($0 < (x) < V$) בכל פיקסל; (2) הערך של $(x) = V$ אינו גבוה מה-רכיבי של (x) , I , כאשר הסימון $(x) \leq W$, $W = \min_{c \in \{r, g, b\}} \{I_c(x)\}$. עם שני האילוצים והतכתיות הללו, השתמשו בסיכון חיזוי כדי להניב את הפונקציה הרצiosa $(x) = V$. תחילת הם סיננו את $(x) = W$ באמצעות מסנן חיזוי כדי לקבל $(x) = B$ ולהקל על ההשפעה של מרכיב מנוגד להסרת העורפל, הם גם יישמו את ההבדל של המוצע המקומיי $(x) = B$ וסתירות התקן המקומית של $(x) = W$. לבסוף, הם הcapsilo את $(x) = C$ בגורם קנה מידה [0, 1], כדי לשלוט בעוצמת שחוור הראות. פיזור האור באטמוספירה $(x) = V$ חושב לפי השלבים הבאים:

- (1) $W(x) = \min_{c \in \{r, g, b\}} (I^c(x))$
- (2) $B(x) = \text{median}_{\Omega}(W(x))$
- (3) $C(x) = B(x) - \text{median}_{\Omega}(|W - B|)(x)$
- (4) $V(x) = \max(\min(pC(x), W(x)), 0)$

כאשר Ω הוא חלון מרובע של מסנן חצויוני. מכיוון שהמסנן החצויוני עצמו אינו משמר קצוות ואינו קוונטורי, מידע רב על הקצה בתמונה (x) V המתקבלת אובד לאחר סינון חצויוני פערמיים, הכולל את הקצוות שבמסגרת ערך העומק משתנה באופן פתאומי. יחד עם זאת, קפיצה גדוֹלה בערך האטמוספרה חשובה מאוד לשחזור תמונה. אם מידע הקצה חסר, האלגוריתם לא יוכל ליזוח את האובך במיקומים אלה, וכטזאה מכך הסרת אובך לא מלאה. את מוצא של(x) W ואת (x) V ניתן לראות באירור הבא :

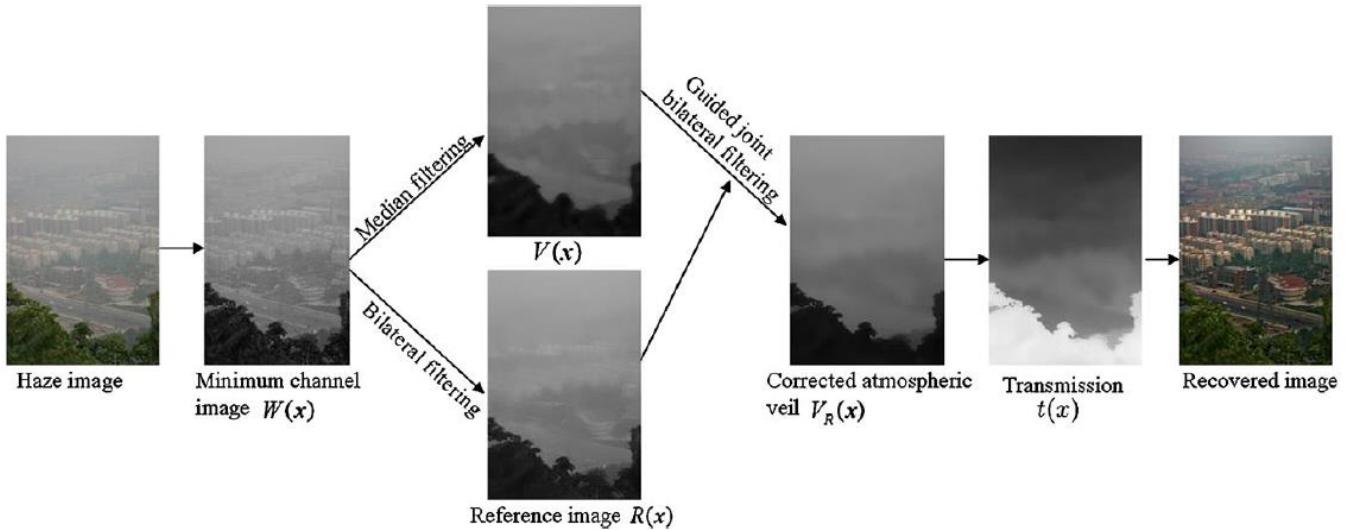


איור 1

הסרת ערפל על בסיס סינון

אנו רואים שהשונות של צעיף האטמוספירה (x)-1=A = (x) V תלוי בעיקר בעומק d של הסנה, כלומר במרקח של האובייקטים לצופה. לפיכך, צעיף האטמוספירה הרצוי צריך להיות חלק וועוצמתו צריכה להשנות בהדרגה בהתאם לעומק הסנה, ובאזורים בעלי אותו עומק צריכה להיות לעוצמתה -(x) V ערך דומה. כאשר אנו מסיקים את -(x) V מה-(x) W, נותר מידע מרקם רב כמו מידע קצה -(x) W. מידע הקצה חשוב מכיוון שהם מניחים קפיצות עומק גדוֹלות בין העצים, כגון הקצוות בין העלים והקירות לפיכך, יש צורך לשזר מידע קצה. אנו חושבים ששיתוט סינון אחת מתאימה לצריכה לשומר על צעיף האווירה חלק, לשמר את קפיצות העומק ולהסיר את מידע המרקם המיותר. כדי להגיע למטרה לעיל, אנו מנסים לתקן את צעיף האטמוספירה (x) V. מטרת אופרטור תיקון זה היא לשזר את מידע הקצה -(x) V, ולהקטין את פרטיה הטקסטוריה -(x) V, מכיוון שמידע מרקם אינו מرمז על שינויים בעומק, אנו לוקחים תמונה המכילה את מידע הקצה של תמונות המקור כתמונה היעוס, ומציגים מסנן מונחה משותף דו-צדדי כדי לשפר את צעיף האטמוספירה (x) V שהושג באמצעות סינון חצויוני, כפי ש�示 באיור 1.

אלגוריתם



איור 2 תרשيم הזרימה של אלגוריתם

Guided joint bilateral filter.

הMSN הדו-צדדי, שהוצע במקור, נמצא בשימוש נרחב בקהילות גרפייה ממוחשבת וראייה ממוחשבת, המSEN הדו-צדדי מחשב את פלט המSEN בפיקסל כממוצע משוקל של פיקסלים שכנים והוא מסוגל לשמר את הקצוות של תמונות מעובדות. עבור כל פיקסל בתמונה I, תן ל-(x)Ω להיות התיקון המקומי שמרכזו ב-x, (x)I ו-(y)I יהיו ערך העוצמה המתאים של פיקסלים x ו-y, ואז ערך העוצמה המסוננת של x הינה:

$$I^B(x) = \frac{\sum_{y \in \Omega(x)} f(x-y) \cdot g(I(x) - I(y)) \cdot I(y)}{\sum_{y \in \Omega(x)} f(x-y) \cdot g(I(x) - I(y))}$$

כאשר f ו-g הם גרעיני המSEN המרחביים והטוח, בהתאם.

הMSN הדו-צדדי מוככל לסינון הדו-צדדי המשותף, שבו גרעין הטוח מחושב על סמך תמונה הדרכה אחרת D, ואז ערך העוצמה המסוננת של x הינה:

$$I_D^B(x) = \frac{\sum_{y \in \Omega(x)} f(x-y) \cdot g(D(x) - D(y)) \cdot I(y)}{\sum_{y \in \Omega(x)} f(x-y) \cdot g(D(x) - D(y))}$$

לפיכך, ה-(x)I_D^B של התמונה המסוננת קיבל את מידע הקצה של תמונה היחס D. המSEN הדו-צדדי המשותף מועדף במיוחד כאשר תמונה הקלט אינה אמינה לספק מידע קצה, למשל, כאשר היא רועשת מאוד או היא תוצאה בינויים בעיבוד תמונה.

על מנת לשמור את הקצה ולתקן את פרטיה הקצה הלא מושלמים, אנחנו לא רק צריים לשקלול את ההבדל בתמונה ההפענה, אלא גם לדרכו לשקלול עוד יותר את ההבדל בין התמונה שיש לשנן לבין תמונה ההפענה, לתת משקלים גדולים יותר לפיקסלים עם סטייה קטנה יותר ומשקלות קטנות יותר לפיקסלים עם סטייה גדולה יותר. באמצעות פועלות הסינוון הזה, יוכל לתקן את התמונה לכיוון תמונה ההפענה.MSN זה נקראMSN משותף דו-צדדי מודרך, וערך העוצמה המסוננת של x הוא:

$$I_D^G(x) = \frac{1}{k} \sum_{y \in \Omega(x)} f(\|y\|) \cdot g(\|D(y)\|) \\ \cdot h(I(y) - D(y)) \cdot I(y)$$

כאשר,

$$h = e^{-x^2/2\sigma_t^2} \quad k = \sum_{y \in \Omega(x)} f(\|y\|) \cdot g(\|D(y)\|) \cdot h(I(y) - D(y)) \\ f(\|y\|) = e^{-(x-y)^2/2\sigma_s^2} \quad g(\|D(y)\|) = e^{-(D(x)-D(y))^2/2\sigma_r^2}$$

לאחר חישוב צעיף האטמוספירה (\hat{V}), שיטה זו מабדلت מידע רב. אנו מננסים להוסיף מידע קצה לתמונה המקורית של (x) עד (\hat{x}), ובינתיים מצמצמים את פרטי המרkers של (\hat{x}). מכיוון שפתחת עירוץ הבחירה המינימלית (x) W מכילה את תכונות הקצה ופרטי המarkers של תמונה הקלט, אנו מציעים להשתמש ב- (x) בرمז לשיפור מידע הקצה של (x) V סביר האзорים עם קפיצות עמוק פתאומיות. אנו משתמשים תחילה בMSN דו-צדדי ב- (x) W כדי לسان כמה פרטי מarker, בעוד שנותן לשמר היטב את תכונות הקצה:

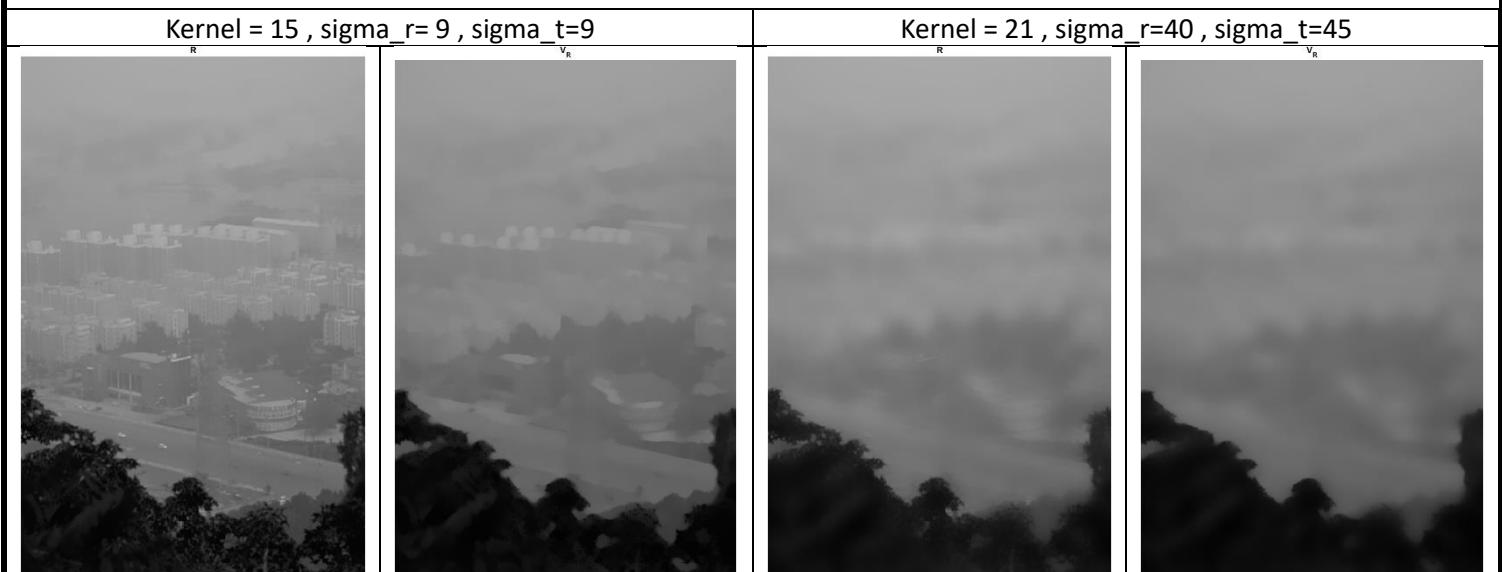
$$R(x) = \frac{\sum_{y \in \Omega(x)} f(x-y) \cdot g(W(x) - W(y)) \cdot W(y)}{\sum_{y \in \Omega(x)} f(x-y) \cdot g(W(x) - W(y))}$$

לאחר מכן ניקח את התמונה(msn) (R) כתמונה התייחסות לsns צעיף האטמוספירה (\hat{V} באמצעות מסנן המונחה המשותף:

כאשר k הוא הגורם המנורמל, ליבתMSN הטווח:

$$g(\|R(y)\|) = e^{-(R(x)-R(y))^2/2\sigma_r^2}.$$

$$V_R(x) = \frac{1}{k} \sum_{y \in \Omega(x)} f(\|y\|) \cdot g(\|R(y)\|) \\ \cdot h(V(y) - R(y)) \cdot V(y)$$



איור 3 . השפעת גודל גרעין וסטיות תקן של גausian

הערכת אור אטמוספירה

אור האטמוספירה הכללית A מוערך בדרך כלל מפיקסלים עם האובך הצפוף ביותר, לדוגמה, לוקחים את ערך הבاهירות הגבוהה ביותר של התמונה כאור הסביבה. אבל הפיקסל הבהיר ביותר עשוי להיות האובייקטים הלבנים. אנו משתמשים בערוץ האפל (W) כדי לשפר את הדיוק של אור האטמוספירה: ראשית, בחרית הפיקסלים הבהירים ביותר (0.2% בערוץ האפל), המותאים לאזורי המ לאורפלים ביותר; לאחר מכן, בחרית הפיקסל הבהיר ביותר ביותר מ בין הפיקסלים באותו מקום בתמונה הקלט המאורפלת כמו אור האווירה הכללי A .

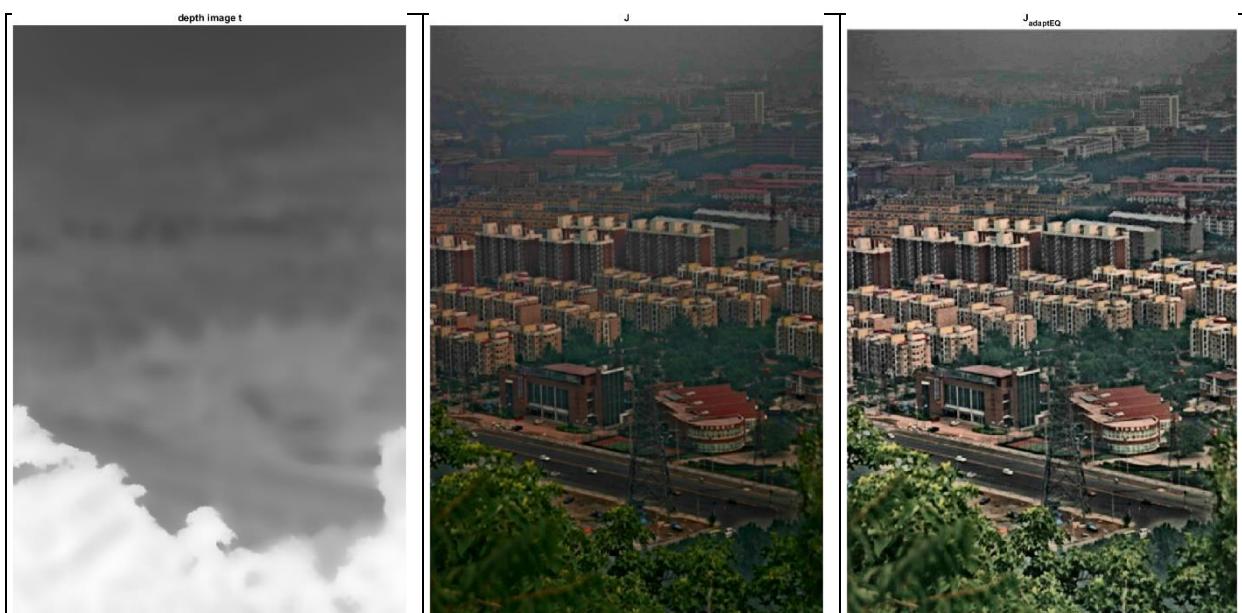
Scene radiance recovering

בהתנן $(x) V_R$ ואור האטמוספירה הכללי A , ניתן לקבל את $(x) t$ על ידי: $t = A/(V_R(x) - \omega)$, כאשר הפרמטר $\omega \in [0, 1]$ משמש לשימור מעט אוביך בסצנה המורחפת והופך את התמונה המשוחזרת לטבעית יותר. ברוב הניסויים זה, הערך של ω הוא 0.95 . ניתן לחשב את הקרגנט הסצנה הסופית (התמונה הברורה נטולת האוביך) $C = J + A \frac{(I(x) - A)}{\max(t(x), t_0)}$ כדי למנוע רעשים באזורי אוביך צפופים מאוד.

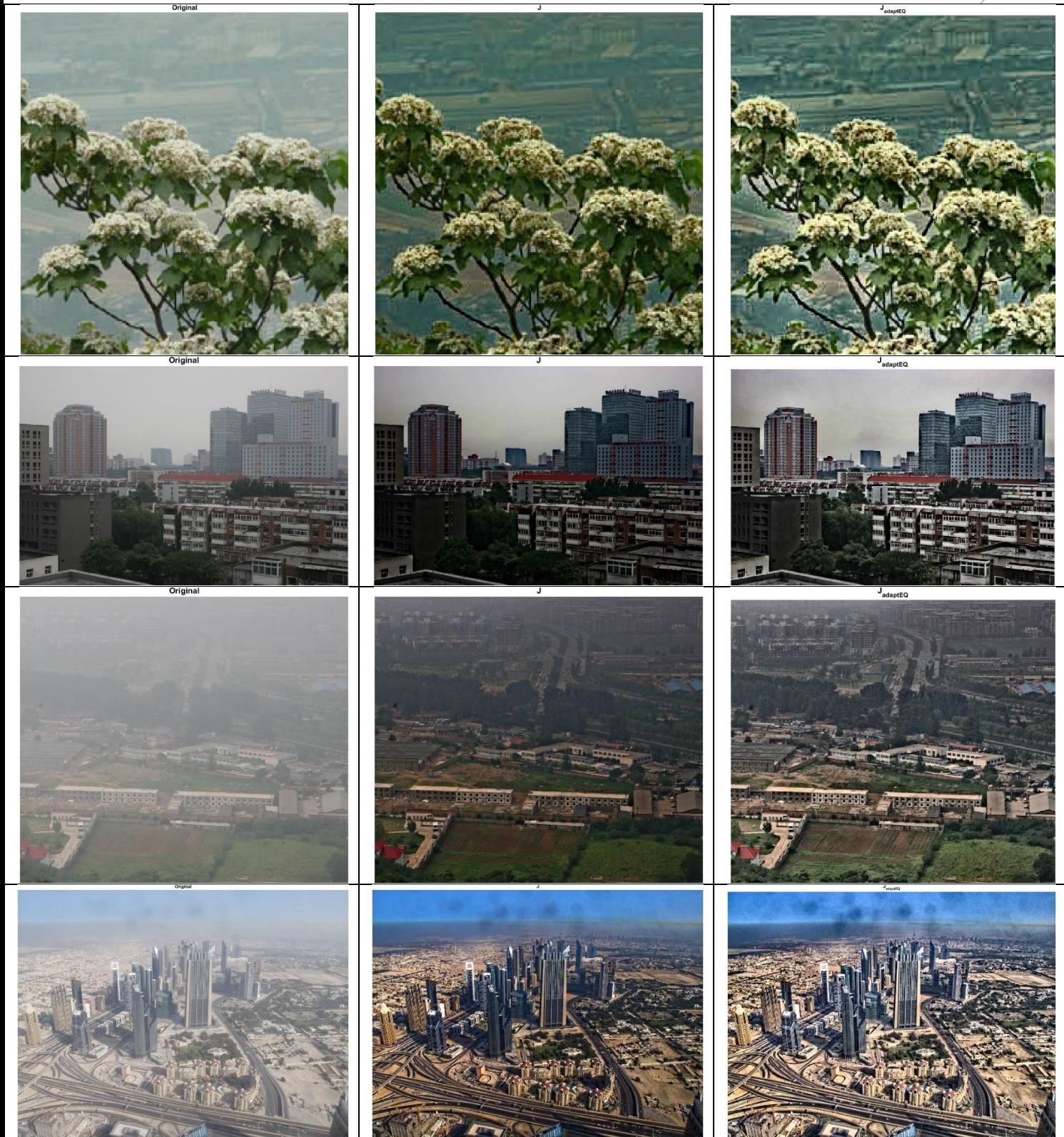
* * * תוספת של *

השווota היסטוגרמאות אדפטטיבית היא שיטה המשמשת לשיפור הניגודיות של תמונה על ידי חלוקה מחדש של עוצמות הפיקסלים. בעוד השוואות היסטוגרמאות מסורתית פועלות על כל התמונה, מחלקת את התמונה לאזורי קטנים יותר ומבצעת השוואות היסטוגרמאות באופן עצמאי בכל אזור. זה מאפשר שיפור ניגודיות טוב יותר באזורים מקומיים.

- המרת תמונה הצבע מ-RGB למרחב צבע LAB. מרחב הצבעים של LAB מפריד בין רכיב הבاهירות (L), המציג את הבاهירות, לבין מידע הצבע (רכיבי a ו- b).
- השוואות היסטוגרמאות על ערוץ L . בשלב זה משפר את הניגודיות ברכיב הבاهירות.
- שילוב ערוץ L המתוקן עם מידע הצבע המקורי בערוצי a ו- b כדי לקבל את תמונה המעבדה המתוקנת.
- המרת תמונה LAB מתוקנת חוזה למרחב הצבע RGB.







Original



J



J adaptEQ



Original



J



J adaptEQ



Original



J



J adaptEQ



Original



J

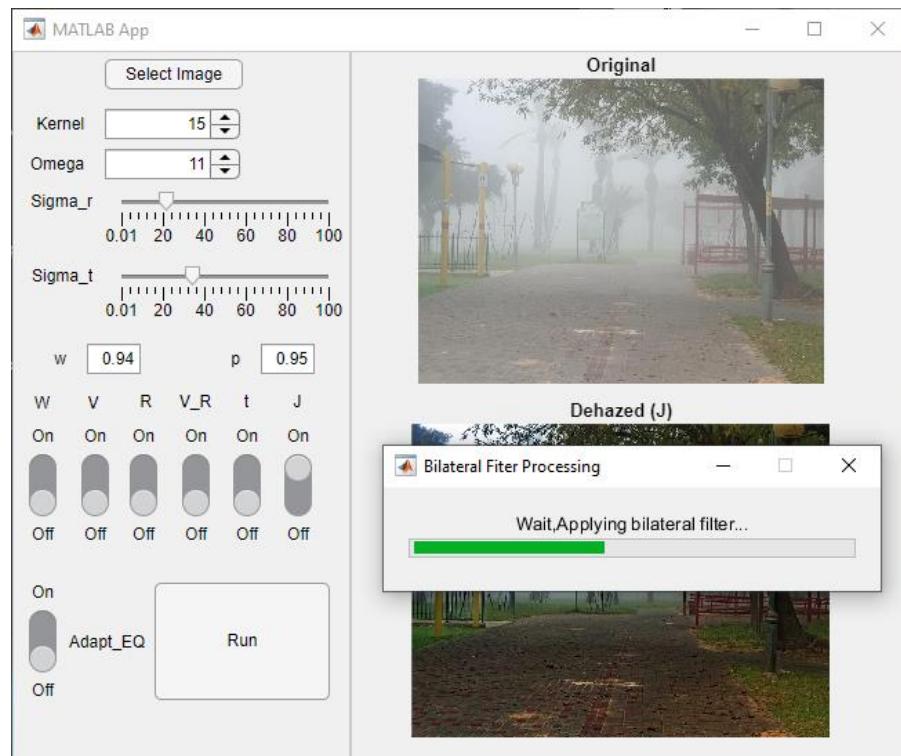


J adaptEQ



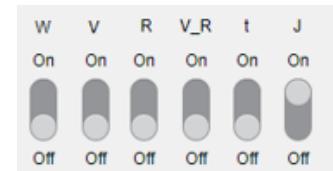


GUI - אופן שימוש
GUI רשום בשפת Matlab



- – Select Image
- – גודל של גרעין עבור מסנן בילטרלי (R, V_R)
- – גודל של גרעין טשטוש במסנן חצויוני (V)
- – סטיית תקן של גרעני גאוסיאני (R, V_R, Sigma_r, Sigma_t)
- – משקל כדי לשולוט בעוצמת שחזור הראות
- – משקל לשימור מעט אובייקט בסצנה המרוחקת
- – Run – כפטור הרצה של אלגוריתם.

- בוררים לצורך הצגה של תמונה
בשלבי עיבוד שונים

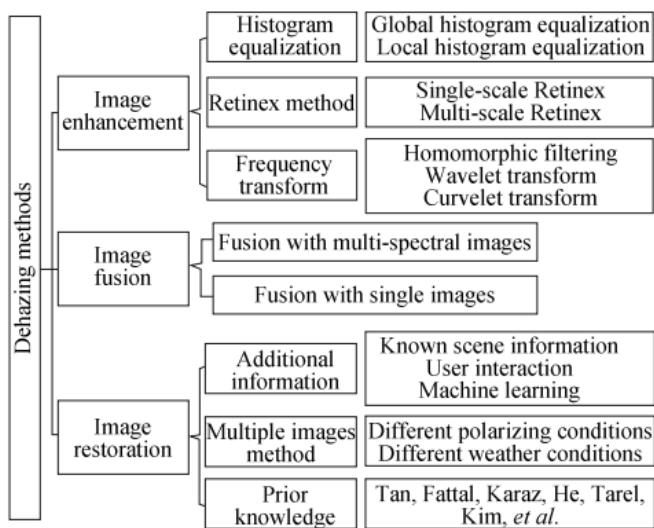


Recent Advances in Image Dehazing

מבוא

בתנאי מזג אוויר כמו ערפל ואובך, איקות התמונות יורדת מאוד עקב השפעת החלקיים באטמוספירה. החלקיים מרחפים יפזרו אור ויגרמו להפחיתה האור המוחרם מהסצנה והאור האטמוספרי המפזר יתערבב גם עם האור המתkeletal במצלמה וישנה את הניגודיות והצבע של התמונה.

לכן, יש צורך במערכות ראייה ממוחשבת כדי לשפר את האפקטים החזותיים של התמונה ולהציג את תוכנות התמונה. טכניקת הסרת אובך של תמונה, הידועה גם בשם "הסרת אובך" או "הסרת ערפל" היא רק הטכניקה להפחיתה או אפילו להסיר הפגיעה עקב אובך על מנת להשיג אפקטים חזותיים מספקים ולקבל מידע שימושי יותר. בתיאוריה, ניקוי אובך של תמונה מסיר אפקטים חזותיים לא רצויים ולעתים קרובות נחשב כטכניקה לשיפור תמונה. עם זאת, היא נבדلت מושיטות מסווגות להסרת רעשים ושיפור ניגודיות שכן השפה של פיקסלים תמונה שנגרמה על ידי נוכחות אובך תלויות במרחק בין האובייקט להתקן הרכישה ובצפיפות האזוריית של האובך. ההשפעה של ערפל על פיקסלים של תמונה גם מדכאת את הטווח הדינמי של הצבעים.



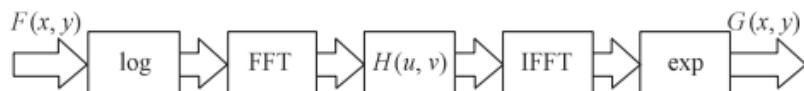
בהתבסס על הבדלים בעקרונות שחרור ערפל, ניתן לחלק את השיטות הנוכחיות לשולש קטגוריות: שיטות מבוססות שיפור תמונה, שיטות מבוססות היתוך תמונה ושיטות מבוססותഴוזו תמונה. שיטות המבוססות על שיפור תמונה אין לוקחות בחשבון את הגורם להידרדרות התמונה, אלא משתמשות בעיקר בשיטות עיבוד תמונה מוקדמות לשיפור הניגודיות והפרטים, שיטות האפקטים החזותיים של התמונה. שיטות מבוססות היתוך תמונה ממקסימות את המידע המועיל מעורציו מקור מרובים כדי ליצור תמונה באיכות גבוהה, ללא צורך במודל פיזי. שיטות מבוססותഴוזו תמונה מבוססות מודל השפה של תמונה מעורפלת על ידי לימוד המנגנוןים הפיזיים של הדמיה אופטית, היפוך התלילי השפה ופיצוי על עיות הנגרם מתחליני השפה אלו על מנת לקבל תמונות ברורות ללא אובך.

Homomorphic filtering

בתנאי ערפל, רכיב התדר הנמוך של התמונה משופרים, כך שנitinן להשתמש במסנן מעביר גבוהים לסינון תמונות כדי לדכא תדרים נמנחים ולשפר תדרים גבוהים. שיפור תחום התדר משתמש תמיד תמיד בניתוח פוריה ובשיטות אחרות כדי להמיר תמונה בתחום התדר. לאחר השלמת פעולת הסינון, מתבצעת טרנספורמציה הפוכה חוזרת בתחום המרחב. שיטות טיפוסיות המבוססות על תחום התדר כוללות סינון הומו-מורפי, והתרמת wavelets.

העיקרונות של סינון הומו-מורפי הוא חלוקת התמונה לרכיב קרינה ורכיב השתקפות. מרכיב הקרינה של התמונה המעורפלת מאופיין בשונות איטית במרחב, ורכיב ההשתקפות מקשר לרוב הפרטיו הסצנה. שיפור התמונה מושג על ידי הסרת רכיב הקרינה. על ידי שילוב של סינון תדרים עם טרנספורמציה בתחום האפור, ניתן להשתמש בטווח הדינמי של התמונה הדחוסה כדי לשפר את איקות התמונה.

לפיכך, העיקרונות הבסיסיים של סינון הומו-מורפי ניקוי אובך עדין מבוסס על מודל ההארה. תרשימים הזורימה של אלגוריתם זה מוצג באירוע 1. כאשר \log הינה התמרת הלוגריתמית, FFT הוא התמרת פוריה, (v, u) הינה פונקציית סינון התדרים, IFFT הינה התמרת פוריה הפוכה ו- exp היא הפעולה הפכית ל \log .



איור 1

אלגוריתם הסינו הומו-מורפי יכול להשיר אזורים לא אחידים שנוצרו על ידי אוור תוך שימוש קומי המתאר של התמונה. עם זאת, הוא זוקק לשתי טרנספורמציות פורייה, פעולה אקספוננציאלית אחת ופעולה לוגריתמית אחת לכל פיקסל של התמונה.

(v) **H פונקציית מסנן הומו-מורפי**, (גauss) :

$$H(u, v) = (\gamma_H - \gamma_L)[1 - e^{-c(D^2(u,v)/D_0^2)}] + \gamma_L$$

כאשר D הוא תדר הקטוען של המסנן, γ_L ו- γ_H הם ערכי המינימום והמקסימום של המסנן, וקבוע c משמש בעיקר כדי לשלוט על ההטיה של פונקציית העברת המסנן. (u, v) הוא המרחק מנקודה (u, v) למקור התדר, המוגדר כדלקמן.

$$D(u, v) = \left[\left(u - \frac{M}{2} \right)^2 + \left(v - \frac{N}{2} \right)^2 \right]^{1/2}$$

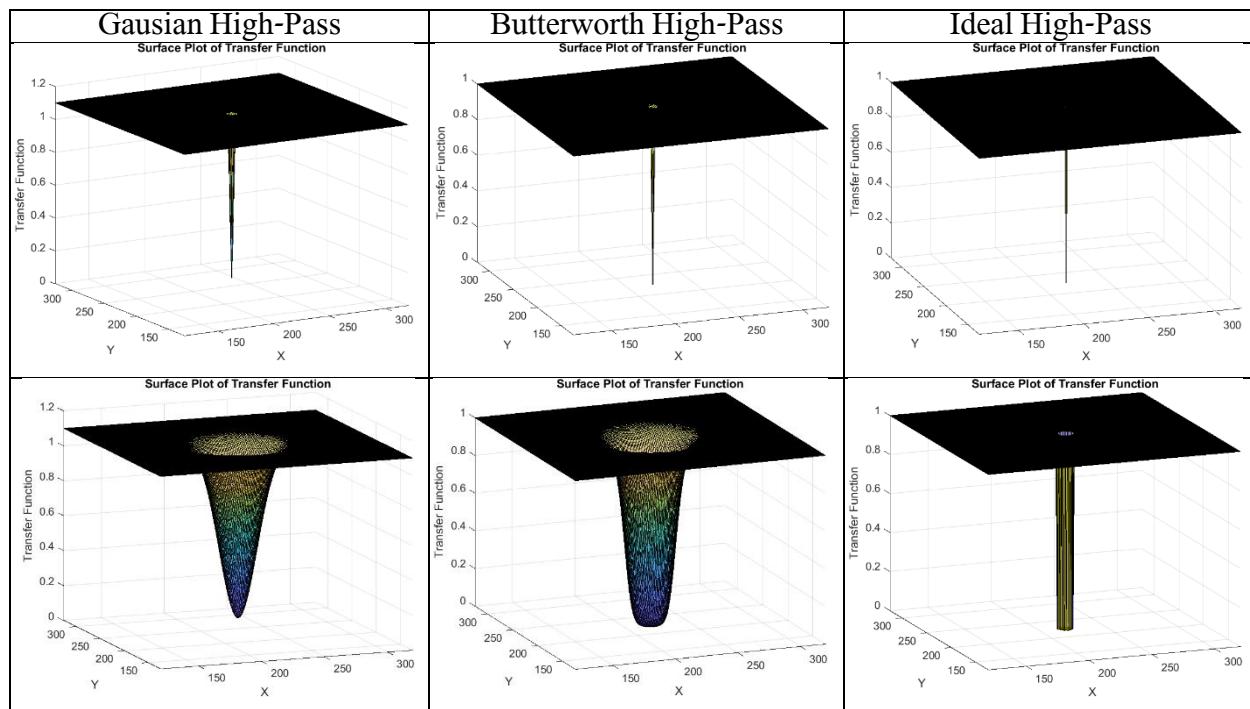
בנוסף הוספה אופצית למסנן BUTTERWORTH ומסנן אידיאלי :

מסנן מעביר גבויים של Butterworth שומר על תדרים מחוץ לרדיויס D_0 . יש לו מעבר הדרוגתי מ-0-ל-1 כדי להפחית חפצי צלול. מסנן (BHPF) של Butterworth בסדר n ותדר חיתוך D_0 .

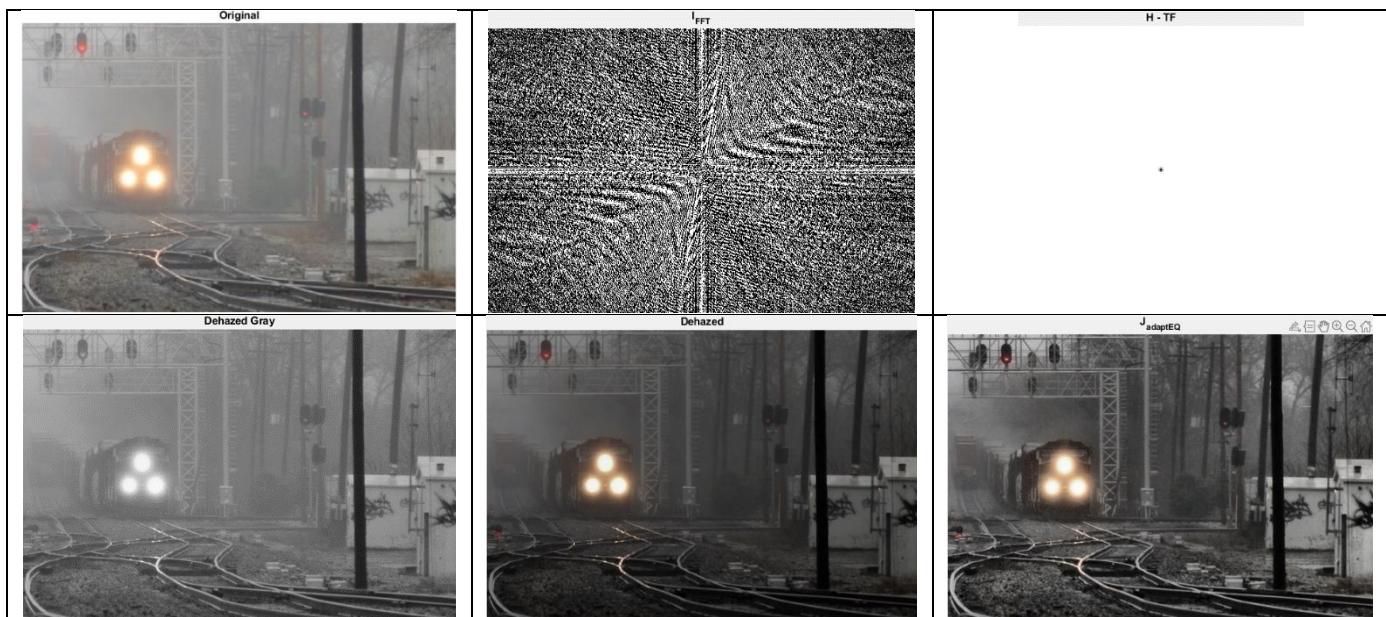
$$H(u, v) = \frac{1}{1 + [D_0/D(u, v)]^{2n}}$$

מסנן מעביר גבויים אידיאלי

$$H(u, v) = \begin{cases} 0 & \text{if } D(u, v) \leq D_0 \\ 1 & \text{if } D(u, v) > D_0 \end{cases}$$



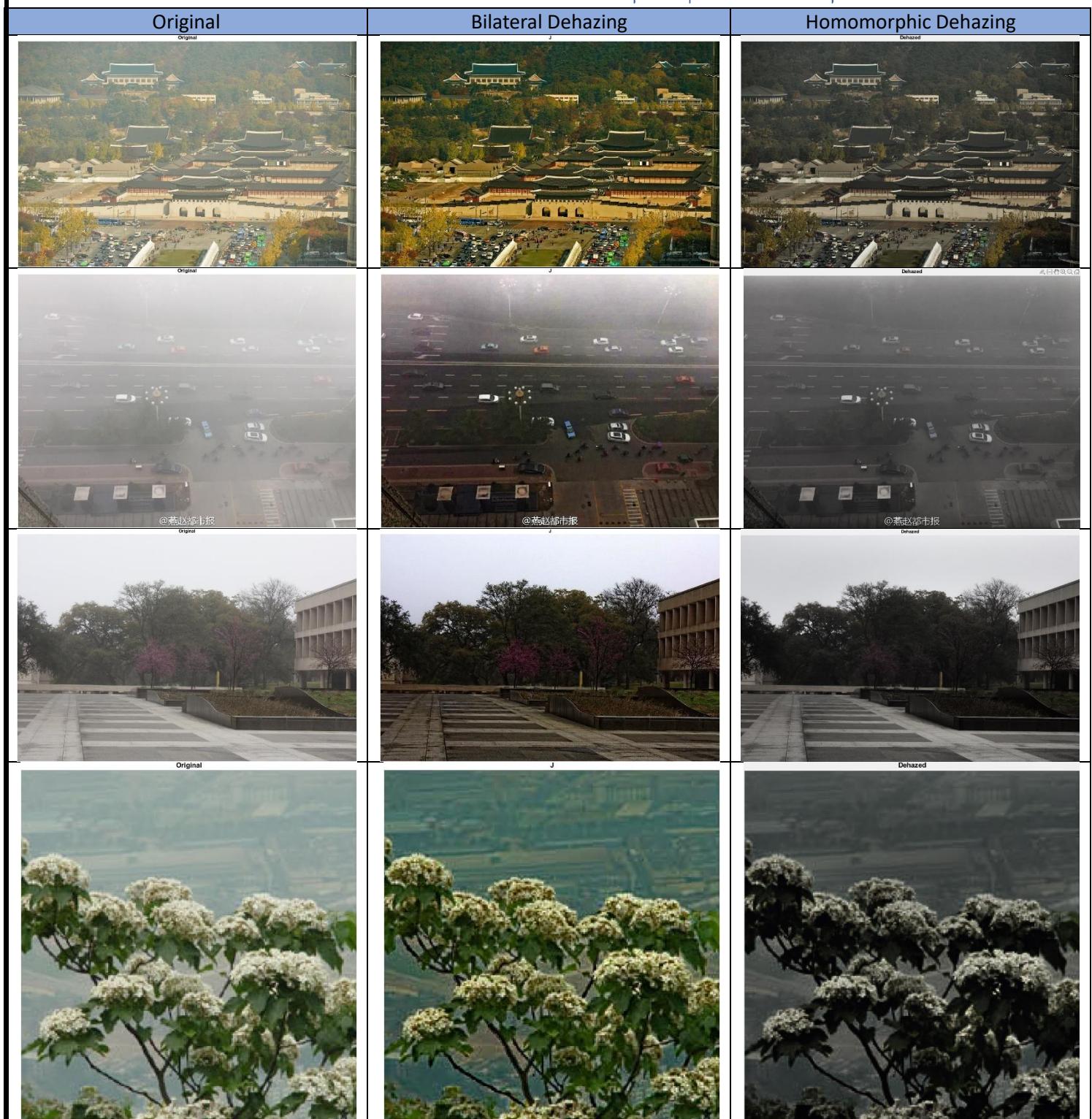
נציג את מוצא לאחר כל שלב באלגוריתם שבאיור 1 בנוסח הוסף אומסציה לתקן היסטוגרמה אדפטיבית כמו בחלק הראשון של הפרויקט זה (נציג רק בעזרת מסנן גאוסיאני):

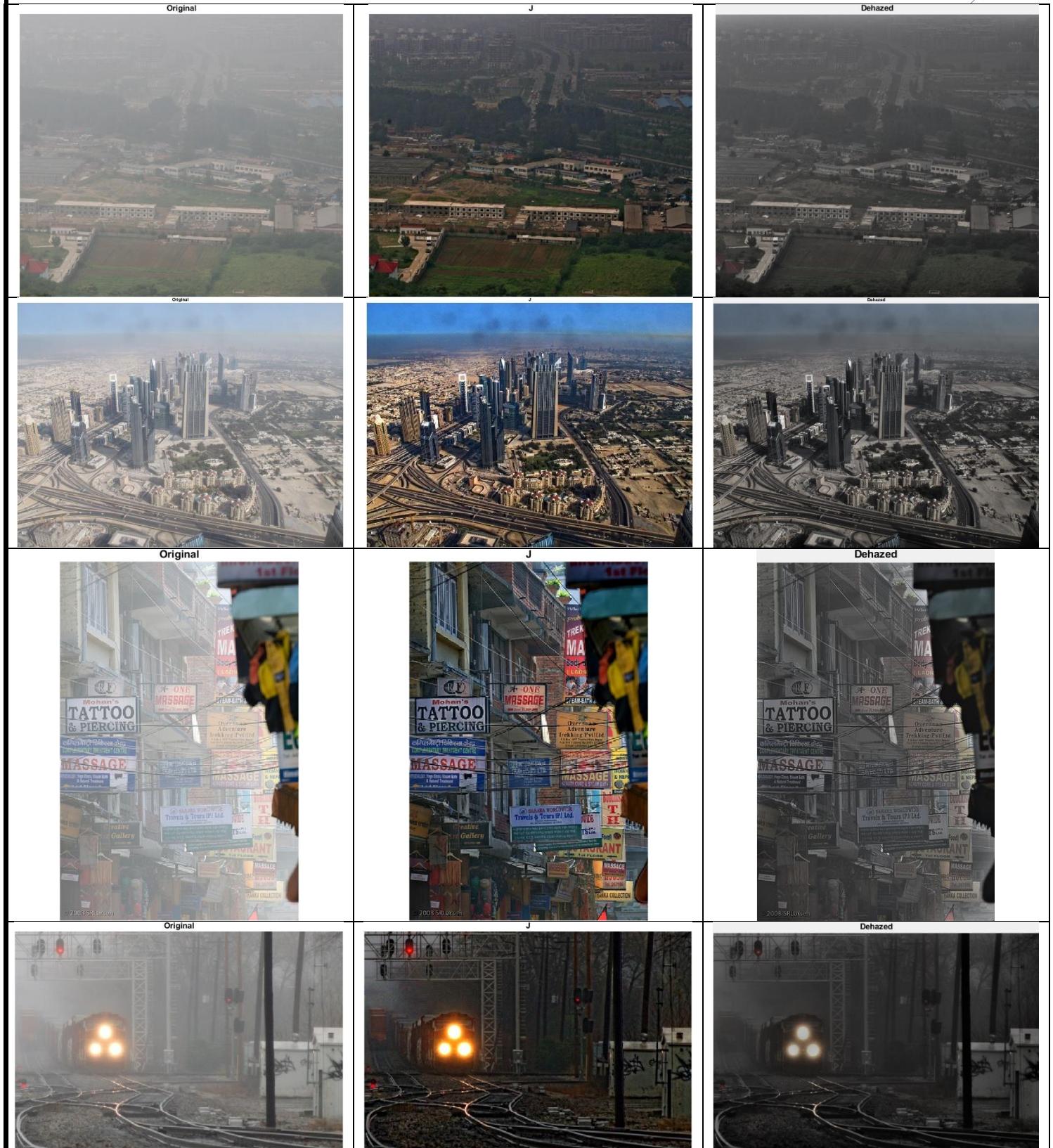


מסקנות:

היתרון של שיטה זו היא ביקר זמן חישוב נמוך מאוד ופשטות של השיטה, אך ככל שמדוברים את רדיוס של מסנן כך יורדת גם הבחרות ככלי של התמונה.

תוצאות השוותיות בין שני שיטות לנקיוי אובך:





מסקנות:

לסיום, המשנן הדו-צדדי המשותף והפילטר ההומומורפי לדי-ערפל של תמונות מציע פתרון מבטיח שהמתמודד ביעילות עם האתגרים שמציבים תנאים מעורפלים. גישה זו לוקחת בחשבון הן את יעילות זמן הריצה והן את השמירה על תכונות חיוניות כגון חדות, דיק צבע ואיכות חזותית כוללת.

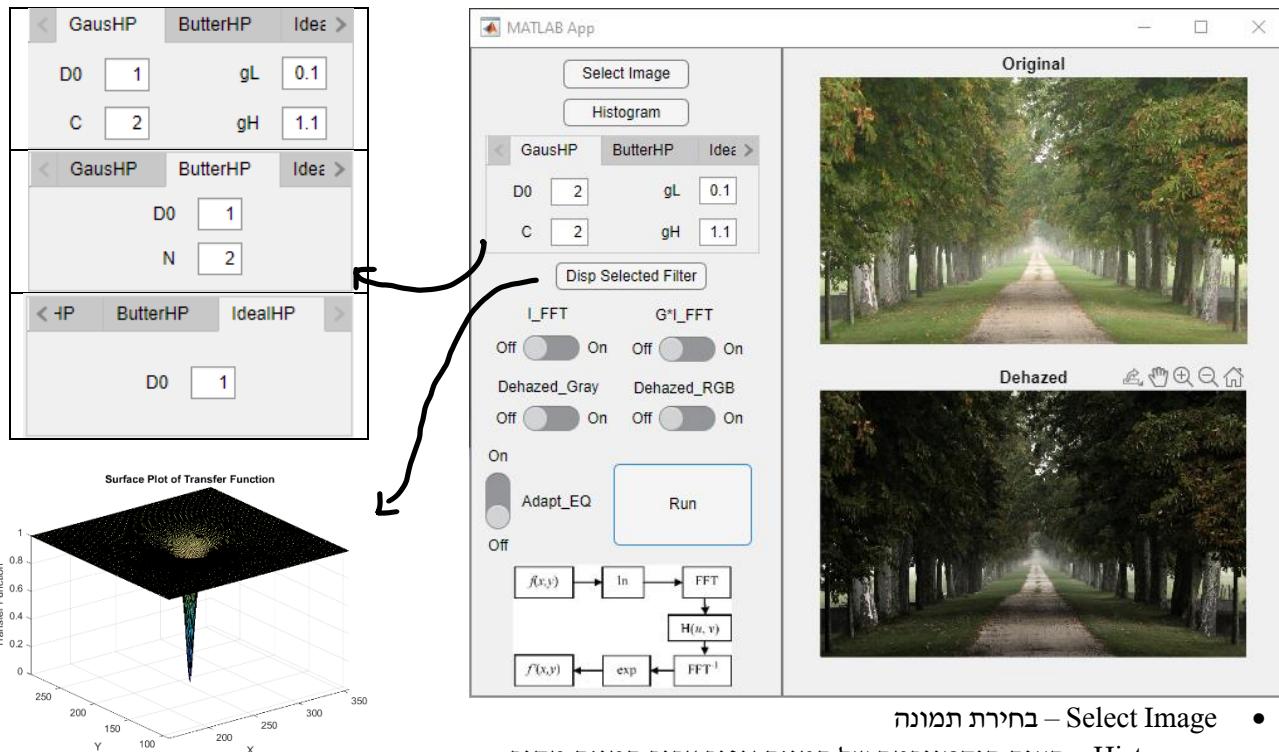
גישה עם המשנן הדו-צדדי המשותף, על ידי התחשבות בדמיון במרחב ובטוחה, מפחית בהצלחה את האובך תוך שמירה על פרטים חשובים בתמונה. היכולת שלו לשמר על חדות הקצה והמרקם הופכת אותו לבחירה אידיאלית למשימות של ניקוי אוביך. יתר על כן, משנן זה מבטיח שזמן הריצה יישאר יעיל, ומאפשר עיבוד כמעט בזמן אמיתי של תמונות מעורפלות.

גישה עם הפילטר ההומורפי, לעומת זאת, ייעיל במיוחד בשיפור הניגודיות בתמונות מעורפלות. על ידי פירוק התמונה לרכיבים התדר, זה מקל על הסרת אוביך וбо זמינות משפר את הנראות של פרטים עדים. משנן זה מבטיח שזמן הריצה יישאר יעיל, ומאפשר עיבוד בזמן אמיתי של תמונות מעורפלות.

הבדל הוא בזורת גישה עם משנן הדו-צדדי משותף ניתן לשחזר ולשמור על צבעי של התמונה

GUI - אופן שימוש

GUI רשום בשפת Matlab

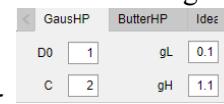


– Select Image

– Histogram

- בחרית פילטר מעביר גבויים

- הצגת פונק תמסורת של משנן הנבחר.



•

•

•

•

- בחרית אופצייה להציג תמונות בכל שלב

- כפטור הריצה של אלגוריתם.

Git ו Appendix B - Script

•

•

•

•

Appendix A

```

classdef FastDehazeImage_exported < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure           matlab.ui.Figure
        GridLayout         matlab.ui.container.GridLayout
        LeftPanel          matlab.ui.container.Panel
        SelectImageButton  matlab.ui.control.Button
        KernelSpinnerLabel matlab.ui.control.Label
        KernelSpinner      matlab.ui.control.Spinner
        OmegaSpinnerLabel  matlab.ui.control.Label
        OmegaSpinner       matlab.ui.control.Spinner
        wEditFieldLabel   matlab.ui.control.Label
        wEditField         matlab.ui.control.NumericEditField
        pEditFieldLabel   matlab.ui.control.Label
        pEditField         matlab.ui.control.NumericEditField
        pEditField         matlab.ui.control.NumericEditField
        WSwitchLabel      matlab.ui.control.Label
        WSwitch            matlab.ui.control.Switch
        VSwitchLabel      matlab.ui.control.Label
        VSwitch            matlab.ui.control.Switch
        V_RSwitchLabel    matlab.ui.control.Label
        V_RSwitch          matlab.ui.control.Switch
        RunButton          matlab.ui.control.Button
        Sigma_tSliderLabel matlab.ui.control.Label
        Sigma_t            matlab.ui.control.Slider
        Sigma_rSlider_2Label matlab.ui.control.Label
        Sigma_r            matlab.ui.control.Slider
        RSwitchLabel      matlab.ui.control.Label
        RSwitch            matlab.ui.control.Switch
        tSwitchLabel      matlab.ui.control.Label
        tSwitch            matlab.ui.control.Switch
        Adapt_EQSwitchLabel matlab.ui.control.Label
        Adapt_EQSwitch    matlab.ui.control.Switch
        JSwitchLabel      matlab.ui.control.Label
        JSwitch            matlab.ui.control.Switch
        RightPanel         matlab.ui.container.Panel
        UIAxes_2           matlab.ui.control.UIAxes
        UIAxes             matlab.ui.control.UIAxes
    end

    % Properties that correspond to apps with auto-reflow
    properties (Access = private)
        onePanelWidth = 576;
    end

    properties (Access = private)
        imageData = []
        imageW = []
        imageV = []
        imageV_R = []
        imageR = []
        imaget = []
        imageJ = []
        imageJ_adapt = []
    end

    % Callbacks that handle component events
    methods (Access = private)

        % Button pushed function: SelectImageButton
        function SelectImageButtonPushed(app, event)

            close all;

            [file ,path] = uigetfile({'*.jpg;*.jpeg;*.png;*.gif;*.tif';'*.*'}, 'File Selector');
            selectedfile = fullfile(path,file);
        

```

```
% Check if a file was selected
if isequal(file, 0) % No file selected
    msgbox('There is NO image selected');
else
    % Read the image file
    app.imageData=imread(selectedfile);
    imshow(app.imageData, 'Parent', app.UIAxes);
    [height, width, ~] = size(app.imageData);
    app.UIAxes.XLim = [0 width];
    app.UIAxes.YLim = [0 height];
    app.KernelSpinner.Enable='on';app.KernelSpinnerLabel.Enable='on';
    app.OmegaSpinner.Enable='on';app.OmegaSpinnerLabel.Enable='on';
    app.Sigma_r.Enable='on';app.Sigma_rSlider_2Label.Enable='on';
    app.Sigma_t.Enable='on';app.Sigma_tSliderLabel.Enable='on';
    app.pEditField.Enable='on';app.pEditFieldLabel.Enable='on';
    app.wEditField.Enable='on';app.wEditFieldLabel.Enable='on';
    app.tSwitch.Enable='on';app.tSwitchLabel.Enable='on';
    app.V_RSwitch.Enable='on';app.V_RSwitchLabel.Enable='on';
    app.RSwitch.Enable='on';app.RSwitchLabel.Enable='on';
    app.VSwitch.Enable='on';app.VSwitchLabel.Enable='on';
    app.WSwitch.Enable='on';app.WSwitchLabel.Enable='on';
    app.JSwitch.Enable='on';app.JSwitchLabel.Enable='on';
    app.Adapt_EQSwitch.Enable='on';app.Adapt_EQSwitchLabel.Enable='on';
    app.RunButton.Enable='on';
end

% Button pushed function: RunButton
function RunButtonPushed(app, event)
close all;
[height, width, ~] = size(app.imageData);
sigma_s=0.03*min(height,width);
sigma_r=app.Sigma_r.Value;
sigma_t=app.Sigma_t.Value;
p=app.pEditField.Value;
w=app.wEditField.Value;
t0=0.3;
radius = app.KernelSpinner.Value;
omega(1:2) = app.OmegaSpinner.Value;
app.imageW = double(min(app.imageData,[],3));
if strcmp(app.WSwitch.Value,'On')
    figure,imshow(uint8(app.imageW)),title('W');
end
B = medfilt2(app.imageW,omega,'symmetric');
C=B-medfilt2(abs(app.imageW-B),omega,'symmetric');
app.imageV=max(min(p.*C,app.imageW),0);
if strcmp(app.VSwitch.Value,'On')
    figure,imshow(uint8(app.imageV)),title('V');
end
app.imageR=bilat_filter(app.imageW,radius,sigma_s,sigma_r);
if strcmp(app.RSwitch.Value,'On')
    figure,imshow(uint8(app.imageR)),title('R');
end

app.imageV_R=bilat_filter_joint(app.imageV,app.imageR,radius,sigma_s,sigma_r,sigma_t);
if strcmp(app.V_RSwitch.Value,'On')
    figure,imshow(uint8(app.imageV_R)),title('V_{R}');
end
A = min([estimateAtmosphericLight(app.imageW), max(max(255-app.imageW))]);
app.imaget=ones(height,width)-w*app.imageV_R/A;
if strcmp(app.tSwitch.Value,'On')
    figure,imshow(app.imaget),title('depth image t');
end
image_double=double(app.imageData);
app.imageJ=zeros(size(app.imageData));
app.imageJ(:,:1)=(image_double(:,:1)-A)./max(app.imaget,t0)+A;
app.imageJ(:,:2)=(image_double(:,:2)-A)./max(app.imaget,t0)+A;
app.imageJ(:,:3)=(image_double(:,:3)-A)./max(app.imaget,t0)+A;
```

```

imshow(uint8(app.imageJ), 'Parent', app.UIAxes_2);
[height, width, ~] = size(app.imageJ);
app.UIAxes_2.XLim = [0 width];
app.UIAxes_2.YLim = [0 height];
if strcmp(app.JSwitch.Value,'On')
    figure,imshow(uint8(app.imageJ)),title('J');
    figure,imshow(uint8(app.imageData)),title('Original');
end
if strcmp(app.Adapt_EQSwitch.Value,'On')
    LAB = rgb2lab(uint8(app.imageJ));
    L = LAB(:,:,1)/100;
    L = adapthisteq(L,'NumTiles',[8 8],'ClipLimit',0.005);
    LAB(:,:,1) = L*100;
    app.imageJ_adapt = lab2rgb(LAB);
    figure,imshow(app.imageJ_adapt),title('J_{adaptEQ}');
end
end

% Changes arrangement of the app based on UIFigure width
function updateAppLayout(app, event)
currentFigureWidth = app.UIFigure.Position(3);
if(currentFigureWidth <= app.onePanelWidth)
    % Change to a 2x1 grid
    app.GridLayout.RowHeight = {516, 516};
    app.GridLayout.ColumnWidth = {'1x'};
    app.RightPanel.Layout.Row = 2;
    app.RightPanel.Layout.Column = 1;
else
    % Change to a 1x2 grid
    app.GridLayout.RowHeight = {'1x'};
    app.GridLayout.ColumnWidth = {245, '1x'};
    app.RightPanel.Layout.Row = 1;
    app.RightPanel.Layout.Column = 2;
end
end
end

% Component initialization
methods (Access = private)

% Create UIFigure and components
function createComponents(app)

    % Create UIFigure and hide until all components are created
    app.UIFigure = uifigure('Visible', 'off');
    app.UIFigure.AutoScaleChildren = 'off';
    app.UIFigure.Position = [100 100 650 516];
    app.UIFigure.Name = 'MATLAB App';
    app.UIFigure.SizeChangedFcn = createCallbackFcn(app, @updateAppLayout, true);

    % Create GridLayout
    app.GridLayout = uigridlayout(app.UIFigure);
    app.GridLayout.ColumnWidth = {245, '1x'};
    app.GridLayout.RowHeight = {'1x'};
    app.GridLayout.ColumnSpacing = 0;
    app.GridLayout.RowSpacing = 0;
    app.GridLayout.Padding = [0 0 0 0];
    app.GridLayout.Scrollable = 'on';

    % Create LeftPanel
    app.LeftPanel = uipanel(app.GridLayout);
    app.LeftPanel.Layout.Row = 1;
    app.LeftPanel.Layout.Column = 1;

    % Create SelectImageButton
    app.SelectImageButton = uibutton(app.LeftPanel, 'push');
    app.SelectImageButton.ButtonPushedFcn = createCallbackFcn(app,
@SelectImageButtonPushed, true);
    app.SelectImageButton.Position = [67 488 100 22];
    app.SelectImageButton.Text = 'Select Image';

```

```
% Create KernelSpinnerLabel
app.KernelSpinnerLabel = uilabel(app.LeftPanel);
app.KernelSpinnerLabel.HorizontalAlignment = 'right';
app.KernelSpinnerLabel.Enable = 'off';
app.KernelSpinnerLabel.Position = [12 452 40 22];
app.KernelSpinnerLabel.Text = 'Kernel';

% Create KernelSpinner
app.KernelSpinner = uislider(app.LeftPanel);
app.KernelSpinner.Step = 2;
app.KernelSpinner.Limits = [1 50];
app.KernelSpinner.RoundFractionalValues = 'on';
app.KernelSpinner.Enable = 'off';
app.KernelSpinner.Position = [67 452 100 22];
app.KernelSpinner.Value = 15;

% Create OmegaSpinnerLabel
app.OmegaSpinnerLabel = uilabel(app.LeftPanel);
app.OmegaSpinnerLabel.HorizontalAlignment = 'right';
app.OmegaSpinnerLabel.Enable = 'off';
app.OmegaSpinnerLabel.Position = [7 423 45 22];
app.OmegaSpinnerLabel.Text = 'Omega';

% Create OmegaSpinner
app.OmegaSpinner = uislider(app.LeftPanel);
app.OmegaSpinner.Step = 2;
app.OmegaSpinner.Limits = [1 50];
app.OmegaSpinner.RoundFractionalValues = 'on';
app.OmegaSpinner.Enable = 'off';
app.OmegaSpinner.Position = [67 423 100 22];
app.OmegaSpinner.Value = 15;

% Create wEditFieldLabel
app.wEditFieldLabel = uilabel(app.LeftPanel);
app.wEditFieldLabel.HorizontalAlignment = 'right';
app.wEditFieldLabel.Enable = 'off';
app.wEditFieldLabel.Position = [14 282 25 22];
app.wEditFieldLabel.Text = 'w';

% Create wEditField
app.wEditField = uieditfield(app.LeftPanel, 'numeric');
app.wEditField.Limits = [0.5 1];
app.wEditField.Enable = 'off';
app.wEditField.Position = [54 282 39 22];
app.wEditField.Value = 0.95;

% Create pEditFieldLabel
app.pEditFieldLabel = uilabel(app.LeftPanel);
app.pEditFieldLabel.HorizontalAlignment = 'right';
app.pEditFieldLabel.Enable = 'off';
app.pEditFieldLabel.Position = [140 282 25 22];
app.pEditFieldLabel.Text = 'p';

% Create pEditField
app.pEditField = uieditfield(app.LeftPanel, 'numeric');
app.pEditField.Limits = [0.5 1];
app.pEditField.Enable = 'off';
app.pEditField.Position = [180 282 39 22];
app.pEditField.Value = 0.95;

% Create WSwitchLabel
app.WSwitchLabel = uilabel(app.LeftPanel);
app.WSwitchLabel.HorizontalAlignment = 'center';
app.WSwitchLabel.Enable = 'off';
app.WSwitchLabel.Position = [9 250 25 22];
app.WSwitchLabel.Text = 'W';

% Create WSwitch
app.WSwitch = uiswitch(app.LeftPanel, 'slider');
app.WSwitch.Orientation = 'vertical';
app.WSwitch.Enable = 'off';
```

```

app.WSwitch.Position = [12 179 20 45];

% Create VSwitchLabel
app.VSwitchLabel = uilabel(app.LeftPanel);
app.VSwitchLabel.HorizontalAlignment = 'center';
app.VSwitchLabel.Enable = 'off';
app.VSwitchLabel.Position = [46 250 25 22];
app.VSwitchLabel.Text = 'V';

% Create VSwitch
app.VSwitch = uiswitch(app.LeftPanel, 'slider');
app.VSwitch.Orientation = 'vertical';
app.VSwitch.Enable = 'off';
app.VSwitch.Position = [50 179 20 45];

% Create V_RSwitchLabel
app.V_RSwitchLabel = uilabel(app.LeftPanel);
app.V_RSwitchLabel.HorizontalAlignment = 'center';
app.V_RSwitchLabel.Enable = 'off';
app.V_RSwitchLabel.Position = [118 251 29 22];
app.V_RSwitchLabel.Text = 'V_R';

% Create V_RSwitch
app.V_RSwitch = uiswitch(app.LeftPanel, 'slider');
app.V_RSwitch.Orientation = 'vertical';
app.V_RSwitch.Enable = 'off';
app.V_RSwitch.Position = [123 179 20 45];

% Create RunButton
app.RunButton = uibutton(app.LeftPanel, 'push');
app.RunButton.ButtonPushedFcn = createCallbackFcn(app, @RunButtonPushed, true);
app.RunButton.Enable = 'off';
app.RunButton.Position = [103 46 127 85];
app.RunButton.Text = 'Run';

% Create Sigma_tSliderLabel
app.Sigma_tSliderLabel = uilabel(app.LeftPanel);
app.Sigma_tSliderLabel.HorizontalAlignment = 'right';
app.Sigma_tSliderLabel.Enable = 'off';
app.Sigma_tSliderLabel.Position = [8 342 50 22];
app.Sigma_tSliderLabel.Text = 'Sigma_t';

% Create Sigma_t
app.Sigma_t = uislider(app.LeftPanel);
app.Sigma_t.Limits = [0.01 100];
app.Sigma_t.MajorTicks = [0.01 20 40 60 80 100];
app.Sigma_t.Enable = 'off';
app.Sigma_t.Position = [79 351 150 3];
app.Sigma_t.Value = 20;

% Create Sigma_rSlider_2Label
app.Sigma_rSlider_2Label = uilabel(app.LeftPanel);
app.Sigma_rSlider_2Label.HorizontalAlignment = 'right';
app.Sigma_rSlider_2Label.Enable = 'off';
app.Sigma_rSlider_2Label.Position = [8 396 50 22];
app.Sigma_rSlider_2Label.Text = 'Sigma_r';

% Create Sigma_r
app.Sigma_r = uislider(app.LeftPanel);
app.Sigma_r.Limits = [0.01 100];
app.Sigma_r.MajorTicks = [0.01 20 40 60 80 100];
app.Sigma_r.Enable = 'off';
app.Sigma_r.Position = [79 405 150 3];
app.Sigma_r.Value = 20;

% Create RSwitchLabel
app.RSwitchLabel = uilabel(app.LeftPanel);
app.RSwitchLabel.HorizontalAlignment = 'center';
app.RSwitchLabel.Enable = 'off';
app.RSwitchLabel.Position = [84 251 25 22];
app.RSwitchLabel.Text = 'R';

```

```
% Create RSwitch
app.RSwitch = uiswitch(app.LeftPanel, 'slider');
app.RSwitch.Orientation = 'vertical';
app.RSwitch.Enable = 'off';
app.RSwitch.Position = [85 179 23 45];

% Create tSwitchLabel
app.tSwitchLabel = uilabel(app.LeftPanel);
app.tSwitchLabel.HorizontalAlignment = 'center';
app.tSwitchLabel.Enable = 'off';
app.tSwitchLabel.Position = [157 251 25 22];
app.tSwitchLabel.Text = 't';

% Create tSwitch
app.tSwitch = uiswitch(app.LeftPanel, 'slider');
app.tSwitch.Orientation = 'vertical';
app.tSwitch.Enable = 'off';
app.tSwitch.Position = [160 179 20 45];

% Create Adapt_EQSwitchLabel
app.Adapt_EQSwitchLabel = uilabel(app.LeftPanel);
app.Adapt_EQSwitchLabel.HorizontalAlignment = 'center';
app.Adapt_EQSwitchLabel.Enable = 'off';
app.Adapt_EQSwitchLabel.Position = [38 77 61 22];
app.Adapt_EQSwitchLabel.Text = 'Adapt_EQ';

% Create Adapt_EQSwitch
app.Adapt_EQSwitch = uiswitch(app.LeftPanel, 'slider');
app.Adapt_EQSwitch.Orientation = 'vertical';
app.Adapt_EQSwitch.Enable = 'off';
app.Adapt_EQSwitch.Position = [12 66 20 45];

% Create JSwitchLabel
app.JSwitchLabel = uilabel(app.LeftPanel);
app.JSwitchLabel.HorizontalAlignment = 'center';
app.JSwitchLabel.Enable = 'off';
app.JSwitchLabel.Position = [194 251 25 22];
app.JSwitchLabel.Text = 'J';

% Create JSwitch
app.JSwitch = uiswitch(app.LeftPanel, 'slider');
app.JSwitch.Orientation = 'vertical';
app.JSwitch.Enable = 'off';
app.JSwitch.Position = [197 179 20 45];

% Create RightPanel
app.RightPanel = uipanel(app.GridLayout);
app.RightPanel.Layout.Row = 1;
app.RightPanel.Layout.Column = 2;

% Create UIAxes_2
app.UIAxes_2 = uiaxes(app.RightPanel);
title(app.UIAxes_2, 'Dehazed (J)')
app.UIAxes_2.XColor = 'none';
app.UIAxes_2.XTick = [];
app.UIAxes_2.YColor = 'none';
app.UIAxes_2.YTick = [];
app.UIAxes_2.ZColor = 'none';
app.UIAxes_2.GridColor = 'none';
app.UIAxes_2.MinorGridColor = 'none';
app.UIAxes_2.Position = [21 14 349 251];

% Create UIAxes
app.UIAxes = uiaxes(app.RightPanel);
title(app.UIAxes, 'Original')
app.UIAxes.XColor = 'none';
app.UIAxes.XTick = [];
app.UIAxes.YColor = 'none';
app.UIAxes.YTick = [];
app.UIAxes.ZColor = 'none';
```

```

app.UIAxes.GridColor = 'none';
app.UIAxes.MinorGridColor = 'none';
app.UIAxes.Position = [21 270 350 245];

% Show the figure after all components are created
app.UIFigure.Visible = 'on';
end
end

% App creation and deletion
methods (Access = public)

    % Construct app
    function app = FastDehazeImage_exported

        % Create UIFigure and components
        createComponents(app)

        % Register the app with App Designer
        registerApp(app, app.UIFigure)

        if nargout == 0
            clear app
        end
    end

    % Code that executes before app deletion
    function delete(app)

        % Delete UIFigure when app is deleted
        delete(app.UIFigure)
    end
end
end

```

```

function outputImage = medianFilter(inputImage, windowSize)
    % Get the size of the input image
    [rows, cols] = size(inputImage);

    % Calculate the padding size based on the window size
    paddingSize = floor(windowSize / 2);

    % Create a padded version of the input image
    paddedImage = padarray(inputImage, [paddingSize, paddingSize]);

    % Create an output image with the same size as the input image
    outputImage = zeros(rows, cols);

    % Apply the median filter
    for i = 1:rows
        for j = 1:cols
            % Extract the window from the padded image
            window = paddedImage(i:i+windowSize-1, j:j+windowSize-1);

            % Calculate the median value of the window
            medianValue = median(window(:));

            % Set the output pixel value to the median value
            outputImage(i, j) = medianValue;
        end
    end
    %outputImage = uint8(outputImage);
end

```

```

function [output_image] = bilat_filter(input_image,radius,sigma_s,sigma_r )
    % Compute the spatial weights
    [X,Y]=meshgrid(-radius:radius,-radius:radius);
    spatial_weights = exp(-(X.^2+Y.^2)/(2*sigma_s^2));

    % Create waitbar.
    h = waitbar(0,'Wait, Applying bilateral filter...');
    set(h,'Name','Bilateral Filter Processing');

    % Compute the size of the input and guidance images
    [rows, cols] = size(input_image);
    % Initialize the output image
    output_image = zeros(size(input_image));

    % Apply the improved joint bilateral filter
    for i = 1:rows
        for j = 1:cols
            % Compute the local region boundaries
            row_min = max(i - radius, 1);
            row_max = min(i + radius, rows);
            col_min = max(j - radius, 1);
            col_max = min(j + radius, cols);

            % Extract the local region of the input
            local_input = input_image(row_min:row_max, col_min:col_max);

            % Compute the range weights
            range_weights = exp(-(local_input-input_image(i, j)).^2 / (2 * sigma_r^2));

            %Taking the product of the range and domain filter.
            weights = spatial_weights((row_min:row_max)-i+radius+1,(col_min:col_max)-j+radius+1).*range_weights;
            output_image(i,j) = sum(weights(:).*local_input(:))/sum(weights(:));
        end
        waitbar(i/rows);
    end
    % Close waitbar.
    close(h);
end

```

```

function atmosphericLight = estimateAtmosphericLight(darkChannel)
    % Determine the number of pixels to consider (0.2% of total pixels)
    numPixels = floor(0.002 * numel(darkChannel));
    % Reshape the channel into a column vector and sort it in descending order
    darkChannelSorted = sort(darkChannel(:), 'descend');
    % Select the top pixels with the highest intensities
    topPixels = darkChannelSorted(1:numPixels);
    % Retrieve the atmospheric light from the hazy image
    atmosphericLight = double(max(topPixels));
end

```

```

function output_image = bilat_filter_joint(input_image, guidance_image, radius, sigma_s, sigma_r,
sigma_t)
    % Compute the spatial weights
    [X,Y]=meshgrid(-radius:radius,-radius:radius);
    spatial_weights = exp(-(X.^2+Y.^2)/(2*sigma_s^2));

    % Create waitbar.
    h = waitbar(0,'Wait ,Applying joint bilateral filter...');
    set(h,'Name','Joint Bilateral Filter Processing');

    % Compute the size of the input and guidance images
    [rows, cols] = size(input_image);

    % Initialize the output image
    output_image = zeros(size(input_image));

    % Apply the improved joint bilateral filter
    for i = 1:rows
        for j = 1:cols
            % Compute the local region boundaries
            row_min = max(i - radius, 1);
            row_max = min(i + radius, rows);
            col_min = max(j - radius, 1);
            col_max = min(j + radius, cols);

            % Extract the local region of the input and guidance images
            local_input = input_image(row_min:row_max, col_min:col_max);
            local_guidance = guidance_image(row_min:row_max, col_min:col_max);

            % Compute the range weights
            range_weights = exp(-(local_input-input_image(i, j)).^2 / (2 * sigma_r^2));

            % Compute the temporal weights
            temporal_weights = exp(-(local_guidance-guidance_image(i, j)).^2 / (2 * sigma_t^2));

            % Compute the combined weights
            weights = spatial_weights((row_min:row_max)-i+radius+1, (col_min:col_max)-
j+radius+1).*range_weights.*temporal_weights;

            % Apply the weighted average to filter the input image
            output_image(i, j) = sum(weights(:).*local_input(:)) / sum(weights(:));
        end
        waitbar(i/rows);
    end
    % Close waitbar.
    close(h);
end

```

Appendix B

```

classdef HomomorphicDehazeImage < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure
        GridLayout
        LeftPanel
        Image
        HistogramButton
        Dehazed_RGBSwitchLabel
        Dehazed_GraySwitch
        Dehazed_GraySwitchLabel
        GI_FFTSwitch
        GI_FFTSwitchLabel
        TabGroup
        GausHPTab
        CEditField
        CEditFieldLabel
        gHEditField
        gHEditFieldLabel
        gLEditField
        gLEditFieldLabel
        D0EditField
        D0EditFieldLabel
        ButterHPTab
        NEditField
        NEditFieldLabel
        D0EditField_2
        D0EditField_2Label
        IdealHPTab
        D0EditField_3
        D0EditField_3Label
        Adapt_EQSwitch
        Adapt_EQSwitchLabel
        RunButton
        I_FFTSwitch
        I_FFTSwitchLabel
        DispSelectedFilterButton
        SelectImageButton
        Dehazed_RGBSwitch
        RightPanel
        UIAxes
        UIAxes_2
    end

    % Properties that correspond to apps with auto-reflow
    properties (Access = private)
        onePanelWidth = 576;
    end

    properties (Access = private)
        imageData = []
        I_gray = []
        I_gray_defog = []
        H = []
        H_selected = []
        I_defog = []
        imageJ_adapt = []
    end

    methods (Access = private)

        function Update_H(app)
            app.Dehazed_GraySwitch.Enable='on';app.Dehazed_GraySwitchLabel.Enable='on';
            app.Dehased_RGBSwitch.Enable='on';app.Dehased_RGBSwitchLabel.Enable='on';
            app.Adapt_EQSwitch.Enable='on';app.Adapt_EQSwitchLabel.Enable='on';
            app.GI_FFTSwitch.Enable='on';app.GI_FFTSwitchLabel.Enable='on';
        end
    
```

```

        app.I_FFTSwitch.Enable='on';app.I_FFTSwitchLabel.Enable='on';
        app.DispSelectedFilterButton.Enable='on';
        app.RunButton.Enable='on';
    end
end

% Callbacks that handle component events
methods (Access = private)

    % Button pushed function: SelectImageButton
    function SelectImageButtonPushed(app, event)

        close all;

        [file ,path] = uigetfile({'*.jpg;*.jpeg;*.png;*.gif;*.tif;*.*'},'File Selector');
        selectedfile = fullfile(path,file);

        % Check if a file was selected
        if isEqual(file, 0) % No file selected
            msgbox('There is NO image selected');
        else
            % Read the image file
            app.imageData=imread(selectedfile);
            imshow(app.imageData, 'Parent', app.UIAxes);
            [height, width, ~] = size(app.imageData);
            app.UIAxes.XLim = [0 width];
            app.UIAxes.YLim = [0 height];
            app.TabGroup.Visible='on';
            app.RunButton.Enable='on';
            app.HistogramButton.Enable='on';

            app.imageData = im2double(app.imageData);
            app.I_gray = rgb2gray(app.imageData);
        end
    end

    % Button pushed function: RunButton
    function RunButtonPushed(app, event)
        close all;
        [app.I_gray_defog ,If,G_I_FFT] = homomorphic_filter(app.I_gray,app.H);

        if strcmp(app.I_FFTSwitch.Value,'On')
            figure,imshow(If),title('I_{FFT}');
        end
        if strcmp(app.GI_FFTSwitch.Value,'On')
            figure,imshow(G_I_FFT),title('G*I_{FFT}');
        end

        if strcmp(app.Dehazed_GraySwitch.Value,'On')
            figure,imshow(app.I_gray_defog),title('Dehazed Gray');
        end
        app.I_defog = zeros(size(app.imageData));
        for i = 1:3
            app.I_defog(:,:,i) = app.imageData(:,:,:,i).*app.I_gray_defog;
        end
        app.I_defog = rescale(app.I_defog);

        imshow(app.I_defog, 'Parent', app.UIAxes_2);
        [height, width, ~] = size(app.I_defog);
        app.UIAxes_2.XLim = [0 width];
        app.UIAxes_2.YLim = [0 height];
        if strcmp(app.Dehazed_RGBSwitch.Value,'On')
            figure,imshow(app.I_defog),title('Dehazed');
        end
        if strcmp(app.Adapt_EQSwitch.Value,'On')
            LAB = rgb2lab(uint8(app.I_defog*255));
            L = LAB(:,:,1)/100;
            L = adapthisteq(L,'NumTiles',[8 8],'ClipLimit',0.005);
            LAB(:,:,1) = L*100;
        end
    end
end

```

```

        app.imageJ_adapt = lab2rgb(LAB);
        figure,imshow(app.imageJ_adapt),title('J_{adaptEQ}');
    end
end

% Callback function: CEditField, D0EditField, GausHPTab, GausHPTab,
%
% ...and 2 other components
function GausHPTabButtonDown(app, event)
    gL = app.gLEditField.Value;
    gH = app.gHEditField.Value;
    C = app.CEditField.Value;
    D0 = app.D0EditField.Value;
    app.H = gaushp(app.I_gray, gL, gH, D0, C);
    app.Update_H()
end

% Callback function: ButterHPTab, ButterHPTab, D0EditField_2,
% ...and 1 other component
function ButterHPTabButtonDown(app, event)
    n = app.NEditField.Value;
    D0 = app.D0EditField_2.Value;
    app.H = butterhp(app.I_gray, D0,n);
    app.Update_H()
end

% Callback function: D0EditField_3, IdealHPTab, IdealHPTab
function IdealHPTabButtonDown(app, event)
    D0 = app.D0EditField_3.Value;
    app.H = idealhp(app.I_gray, D0);
    app.Update_H()
end

% Button pushed function: DispSelectedFilterButton
function DispSelectedFilterButtonPushed(app, event)
    close all;
    plotTFSurface(app.H);
end

% Button pushed function: HistogramButton
function HistogramButtonPushed(app, event)
    figure;imhist(app.I_gray);title('Histogram');
end

% Changes arrangement of the app based on UIFigure width
function updateAppLayout(app, event)
    currentFigureWidth = app.UIFigure.Position(3);
    if(currentFigureWidth <= app.onePanelWidth)
        % Change to a 2x1 grid
        app.GridLayout.RowHeight = {516, 516};
        app.GridLayout.ColumnWidth = {'1x'};
        app.RightPanel.Layout.Row = 2;
        app.RightPanel.Layout.Column = 1;
    else
        % Change to a 1x2 grid
        app.GridLayout.RowHeight = {'1x'};
        app.GridLayout.ColumnWidth = {248, '1x'};
        app.RightPanel.Layout.Row = 1;
        app.RightPanel.Layout.Column = 2;
    end
end

% Component initialization
methods (Access = private)

    % Create UIFigure and components
    function createComponents(app)

        % Get the file path for locating images
        pathToMLAPP = fileparts(mfilename('fullpath'));

```

```
% Create UIFigure and hide until all components are created
app.UIFigure = uifigure('Visible', 'off');
app.UIFigure.AutoScaleChildren = 'off';
app.UIFigure.Position = [100 100 650 516];
app.UIFigure.Name = 'MATLAB App';
app.UIFigure.SizeChangedFcn = createCallbackFcn(app, @updateAppLayout, true);

% Create GridLayout
app.GridLayout = uigridlayout(app.UIFigure);
app.GridLayout.ColumnWidth = {248, '1x'};
app.GridLayout.RowHeight = {'1x'};
app.GridLayout.ColumnSpacing = 0;
app.GridLayout.RowSpacing = 0;
app.GridLayout.Padding = [0 0 0 0];
app.GridLayout.Scrollable = 'on';

% Create LeftPanel
app.LeftPanel = uipanel(app.GridLayout);
app.LeftPanel.Layout.Row = 1;
app.LeftPanel.Layout.Column = 1;

% Create Dehazed_RGBSwitch
app.Dehazed_RGBSwitch = uiswitch(app.LeftPanel, 'slider');
app.Dehazed_RGBSwitch.Enable = 'off';
app.Dehazed_RGBSwitch.Position = [150 211 45 20];

% Create SelectImageButton
app.SelectImageButton = uibutton(app.LeftPanel, 'push');
app.SelectImageButton.ButtonPushedFcn = createCallbackFcn(app,
@SelectImageButtonPushed, true);
app.SelectImageButton.Position = [74 483 100 23];
app.SelectImageButton.Text = 'Select Image';

% Create DispSelectedFilterButton
app.DispSelectedFilterButton = uibutton(app.LeftPanel, 'push');
app.DispSelectedFilterButton.ButtonPushedFcn = createCallbackFcn(app,
@DispSelectedFilterButtonPushed, true);
app.DispSelectedFilterButton.Enable = 'off';
app.DispSelectedFilterButton.Position = [68 323 120 22];
app.DispSelectedFilterButton.Text = 'Disp Selected Filter';

% Create I_FFTSwitchLabel
app.I_FFTSwitchLabel = uilabel(app.LeftPanel);
app.I_FFTSwitchLabel.HorizontalAlignment = 'center';
app.I_FFTSwitchLabel.Enable = 'off';
app.I_FFTSwitchLabel.Position = [43 293 37 22];
app.I_FFTSwitchLabel.Text = 'I_FFT';

% Create I_FFTSwitch
app.I_FFTSwitch = uiswitch(app.LeftPanel, 'slider');
app.I_FFTSwitch.Enable = 'off';
app.I_FFTSwitch.Position = [42 267 45 20];

% Create RunButton
app.RunButton = uibutton(app.LeftPanel, 'push');
app.RunButton.ButtonPushedFcn = createCallbackFcn(app, @RunButtonPushed, true);
app.RunButton.Enable = 'off';
app.RunButton.Position = [114 120 110 65];
app.RunButton.Text = 'Run';

% Create Adapt_EQSwitchLabel
app.Adapt_EQSwitchLabel = uilabel(app.LeftPanel);
app.Adapt_EQSwitchLabel.HorizontalAlignment = 'center';
app.Adapt_EQSwitchLabel.Enable = 'off';
app.Adapt_EQSwitchLabel.Position = [38 144 61 22];
app.Adapt_EQSwitchLabel.Text = 'Adapt_EQ';

% Create Adapt_EQSwitch
app.Adapt_EQSwitch = uiswitch(app.LeftPanel, 'slider');
app.Adapt_EQSwitch.Orientation = 'vertical';
```

```

app.Adapt_EQSwitch.Enable = 'off';
app.Adapt_EQSwitch.Position = [12 133 20 45];

% Create TabGroup
app.TabGroup = uitabgroup(app.LeftPanel);
app.TabGroup.Visible = 'off';
app.TabGroup.Position = [12 352 218 94];

% Create GausHPTab
app.GausHPTab = uitab(app.TabGroup);
app.GausHPTab.SizeChangedFcn = createCallbackFcn(app, @GausHPTabButtonDown, true);
app.GausHPTab.Title = 'GausHP';
app.GausHPTab.ButtonDownFcn = createCallbackFcn(app, @GausHPTabButtonDown, true);

% Create D0EditFieldLabel
app.D0EditFieldLabel = uilabel(app.GausHPTab);
app.D0EditFieldLabel.HorizontalAlignment = 'right';
app.D0EditFieldLabel.Position = [12 38 25 22];
app.D0EditFieldLabel.Text = 'D0';

% Create D0EditField
app.D0EditField = uieditfield(app.GausHPTab, 'numeric');
app.D0EditField.Limits = [1 100];
app.D0EditField.ValueChangedFcn = createCallbackFcn(app, @GausHPTabButtonDown, true);
app.D0EditField.Position = [52 38 30 22];
app.D0EditField.Value = 1;

% Create gLEditFieldLabel
app.gLEditFieldLabel = uilabel(app.GausHPTab);
app.gLEditFieldLabel.HorizontalAlignment = 'right';
app.gLEditFieldLabel.Position = [130 39 25 22];
app.gLEditFieldLabel.Text = 'gL';

% Create gLEditField
app.gLEditField = uieditfield(app.GausHPTab, 'numeric');
app.gLEditField.Limits = [0.09 1];
app.gLEditField.ValueChangedFcn = createCallbackFcn(app, @GausHPTabButtonDown, true);
app.gLEditField.Position = [170 39 30 22];
app.gLEditField.Value = 0.1;

% Create gHEditFieldLabel
app.gHEditFieldLabel = uilabel(app.GausHPTab);
app.gHEditFieldLabel.HorizontalAlignment = 'right';
app.gHEditFieldLabel.Position = [130 6 25 22];
app.gHEditFieldLabel.Text = 'gH';

% Create gHEditField
app.gHEditField = uieditfield(app.GausHPTab, 'numeric');
app.gHEditField.Limits = [1 4];
app.gHEditField.ValueChangedFcn = createCallbackFcn(app, @GausHPTabButtonDown, true);
app.gHEditField.Position = [170 6 30 22];
app.gHEditField.Value = 1.1;

% Create CEditFieldLabel
app.CEditFieldLabel = uilabel(app.GausHPTab);
app.CEditFieldLabel.HorizontalAlignment = 'right';
app.CEditFieldLabel.Position = [12 6 25 22];
app.CEditFieldLabel.Text = 'C';

% Create CEditField
app.CEditField = uieditfield(app.GausHPTab, 'numeric');
app.CEditField.Limits = [1 100];
app.CEditField.ValueChangedFcn = createCallbackFcn(app, @GausHPTabButtonDown, true);
app.CEditField.Position = [52 6 30 22];
app.CEditField.Value = 2;

% Create ButterHPTab
app.ButterHPTab = uitab(app.TabGroup);
app.ButterHPTab.SizeChangedFcn = createCallbackFcn(app, @ButterHPTabButtonDown,
true);
app.ButterHPTab.Title = 'ButterHP';

```

```

app.ButterHPTab.ButtonDownFcn = createCallbackFcn(app, @ButterHPTabButtonDown, true);

% Create D0EditField_2Label
app.D0EditField_2Label = uilabel(app.ButterHPTab);
app.D0EditField_2Label.HorizontalAlignment = 'right';
app.D0EditField_2Label.Position = [74 41 25 22];
app.D0EditField_2Label.Text = 'D0';

% Create D0EditField_2
app.D0EditField_2 = uieditfield(app.ButterHPTab, 'numeric');
app.D0EditField_2.Limits = [1 100];
app.D0EditField_2.ValueChangedFcn = createCallbackFcn(app, @ButterHPTabButtonDown,
true);
app.D0EditField_2.Position = [114 41 30 22];
app.D0EditField_2.Value = 1;

% Create NEditFieldLabel
app.NEditFieldLabel = uilabel(app.ButterHPTab);
app.NEditFieldLabel.HorizontalAlignment = 'right';
app.NEditFieldLabel.Position = [74 11 25 22];
app.NEditFieldLabel.Text = 'N';

% Create NEditField
app.NEditField = uieditfield(app.ButterHPTab, 'numeric');
app.NEditField.Limits = [1 100];
app.NEditField.RoundFractionalValues = 'on';
app.NEditField.ValueChangedFcn = createCallbackFcn(app, @ButterHPTabButtonDown,
true);
app.NEditField.Position = [114 11 30 22];
app.NEditField.Value = 2;

% Create IdealHPTab
app.IdealHPTab = uitab(app.TabGroup);
app.IdealHPTab.SizeChangedFcn = createCallbackFcn(app, @IdealHPTabButtonDown, true);
app.IdealHPTab.Title = 'IdealHP';
app.IdealHPTab.ButtonDownFcn = createCallbackFcn(app, @IdealHPTabButtonDown, true);

% Create D0EditField_3Label
app.D0EditField_3Label = uilabel(app.IdealHPTab);
app.D0EditField_3Label.HorizontalAlignment = 'right';
app.D0EditField_3Label.Position = [70 25 25 22];
app.D0EditField_3Label.Text = 'D0';

% Create D0EditField_3
app.D0EditField_3 = uieditfield(app.IdealHPTab, 'numeric');
app.D0EditField_3.Limits = [1 100];
app.D0EditField_3.RoundFractionalValues = 'on';
app.D0EditField_3.ValueChangedFcn = createCallbackFcn(app, @IdealHPTabButtonDown,
true);
app.D0EditField_3.Position = [110 25 30 22];
app.D0EditField_3.Value = 1;

% Create GI_FFTSwitchLabel
app.GI_FFTSwitchLabel = uilabel(app.LeftPanel);
app.GI_FFTSwitchLabel.HorizontalAlignment = 'center';
app.GI_FFTSwitchLabel.Enable = 'off';
app.GI_FFTSwitchLabel.Position = [144 292 52 22];
app.GI_FFTSwitchLabel.Text = 'G*I_FFT';

% Create GI_FFTSwitch
app.GI_FFTSwitch = uiswitch(app.LeftPanel, 'slider');
app.GI_FFTSwitch.Enable = 'off';
app.GI_FFTSwitch.Position = [150 266 45 20];

% Create Dehazed_GraySwitchLabel
app.Dehazed_GraySwitchLabel = uilabel(app.LeftPanel);
app.Dehazed_GraySwitchLabel.HorizontalAlignment = 'center';
app.Dehazed_GraySwitchLabel.Enable = 'off';
app.Dehazed_GraySwitchLabel.Position = [19 235 86 22];
app.Dehazed_GraySwitchLabel.Text = 'Dehazed_Gray';

```

```
% Create Dehazed_GraySwitch
app.Dehazed_GraySwitch = uiswitch(app.LeftPanel, 'slider');
app.Dehazed_GraySwitch.Enable = 'off';
app.Dehazed_GraySwitch.Position = [42 211 45 20];

% Create Dehazed_RGBSwitchLabel
app.Dehased_RGBSwitchLabel = uilabel(app.LeftPanel);
app.Dehased_RGBSwitchLabel.HorizontalAlignment = 'center';
app.Dehased_RGBSwitchLabel.Enable = 'off';
app.Dehased_RGBSwitchLabel.Position = [127 234 86 22];
app.Dehased_RGBSwitchLabel.Text = 'Dehased_RGB';

% Create HistogramButton
app.HistogramButton = uibutton(app.LeftPanel, 'push');
app.HistogramButton.ButtonPushedFcn = createCallbackFcn(app, @HistogramButtonPushed,
true);
app.HistogramButton.Enable = 'off';
app.HistogramButton.Position = [74 453 100 22];
app.HistogramButton.Text = 'Histogram';

% Create Image
app.Image = uiimage(app.LeftPanel);
app.Image.Position = [24 7 199 106];
app.Image.ImageSource = fullfile(pathToMLAPP, '2023-05-26_18h16_08.png');

% Create RightPanel
app.RightPanel = uipanel(app.GridLayout);
app.RightPanel.Layout.Row = 1;
app.RightPanel.Layout.Column = 2;

% Create UIAxes_2
app.UIAxes_2 = uiaxes(app.RightPanel);
title(app.UIAxes_2, 'Dehazed')
app.UIAxes_2.XColor = 'none';
app.UIAxes_2.XTick = [];
app.UIAxes_2.YColor = 'none';
app.UIAxes_2.YTick = [];
app.UIAxes_2.ZColor = 'none';
app.UIAxes_2.GridColor = 'none';
app.UIAxes_2.MinorGridColor = 'none';
app.UIAxes_2.Position = [25 12 350 250];

% Create UIAxes
app.UIAxes = uiaxes(app.RightPanel);
title(app.UIAxes, 'Original')
app.UIAxes.XColor = 'none';
app.UIAxes.XTick = [];
app.UIAxes.YColor = 'none';
app.UIAxes.YTick = [];
app.UIAxes.ZColor = 'none';
app.UIAxes.GridColor = 'none';
app.UIAxes.MinorGridColor = 'none';
app.UIAxes.Position = [25 265 350 250];

% Show the figure after all components are created
app.UIFigure.Visible = 'on';
end
end

% App creation and deletion
methods (Access = public)

    % Construct app
    function app = HomomorphicDehazeImage

        % Create UIFigure and components
        createComponents(app)

        % Register the app with App Designer
        registerApp(app, app.UIFigure)
```

```

if nargout == 0
    clear app
end
end

% Code that executes before app deletion
function delete(app)

    % Delete UIFigure when app is deleted
    delete(app.UIFigure)
end
end

```

```

function [Iout ,If,G] = homomorphic_filter(I,H)
tic
I_log = log(1 + I);
If = fft2(I_log);
If = fftshift(If);
G = H.*If;
Iout = real(ifft2(ifftshift(G)));
Iout = exp(Iout) - 1;
toc
end

```

```

function H = gaushp(I, gL, gH, D0, C)
%gL      - low values
%gH      - high values
%D0      - size of the gaussian, affects more/less frequencies
%C      - as with D0, it affects the exponential in H. It is like a tug of war
%      between D0 and C, working in different ranges. D0 is a sharpness factor
%      for the big changes in the image whereas c is sharpening in a small range.

[M,N] = size(I);
% Generate a meshgrid for the frequency domain
[X, Y] = meshgrid(1:N,1:M);
gaussianNumerator = (X - floor(N/2)).^2 + (Y - floor(M/2)).^2;
H = (gH-gL)*(1-exp(-C*gaussianNumerator./(D0.^2)))+gL;
end

```

```

function [out]=butterhp(I, D0,n)
%D0      - BHPF passes all the frequencies greater than D_{0} value without attenuation and cuts
% off all the frequencies less than it. The transition point between H(u, v) = 1 and H(u, v) = 0,
% so this is termed as cutoff frequency. But instead of making a sharp cut-off(like, Ideal High
% pass Filter (IHPF)), it introduces a smooth transition from 0 to 1 to reduce ringing artifacts.
%n      - filter order value

[M,N] = size(I);
% Generate a meshgrid for the frequency domain
[X, Y] = meshgrid(1:N,1:M);
D = sqrt((X - floor(N/2)).^2 + (Y - floor(M/2)).^2);
out=1./(1.+(D0./D).^(2*n));
end

```

```

function H = idealhp(I, cutoff)
% size: size of the filter in pixels
% cutoff: frequency cutoff for the high-pass filter

[M,N] = size(I);
% Generate a meshgrid for the frequency domain
[X, Y] = meshgrid(1:N,1:M);
D = ((X - floor(N/2)).^2 + (Y - floor(M/2)).^2);

% Create the ideal high-pass filter
H = double(D >= cutoff);
end

```

```

function plotTFSurface(Z)
[M,N] = size(Z);
figure,imshow(Z,[],title('H - TF'));
% Zoom in on the center of the plot
ylim([floor(M/2)-100, floor(M/2)+100]);
xlim([floor(N/2)-100, floor(N/2)+100]);

% Create the surface plot
figure;
surf(Z,'FaceAlpha',0.5);
xlabel('X');
ylabel('Y');
zlabel('Transfer Function');
title('Surface Plot of Transfer Function');

% Zoom in on the center of the plot
ylim([floor(M/2)-100, floor(M/2)+100]);
xlim([floor(N/2)-100, floor(N/2)+100]);

end

```