ΜΥΕ037 Ψηφιακή Επεξεργασία Εικόνας: Εαρινό Εξάμηνο 2025

Εργασία: 30% του συνολικού βαθμού

Διδάσκων: Άγγελος Γιώτης

ΠΑΡΑΔΟΣΗ: Δευτέρα, 26 Μαΐου 2025, 23:59

May 5, 2025

1 Αποκατάσταση Παραμορφωμένων Εικόνων - Βασική θεωρία Ψηφιακής Επεξεργασίας Εικόνας στο πεδίο του χώρου και της συχνότητας.

1.1 Γενικές Οδηγίες:

Σε αυτήν την Εργασία, θα κατεβάσετε από τη σελίδα ecourse του μαθήματος το αρχείο `assignment.zip` και θα αποσυμπιέσετε το φάκελο `assignment/` τοπικά στον Η/Υ σας. Η εργασία περιλαμβάνει 2 μέρη:

1. Υποβολή του κώδικα Python που θα συμπληρώσετε στο σημειωματάριο `assignment.ipynb` που θα βρείτε στο φάκελο `assignment/` απαντώντας στα ερωτήματα της 1ης Άσκησης, και μετατροπή/αποθήκευση του notebook σε PDF αρχείο.
2. Απάντηση/αναφορά στα θεωρητικά ερωτήματα της 2ης Άσκησης, **σε μορφή PDF**.

Η 2η Άσκηση μπορεί να υλοποιηθεί:

- **είτε** απευθείας στο σημειωματάριο με *Markdown* κελιά κειμένου και LaTeX για εξισώσεις, και υποβάλετε το σημειωματάριο ως έχει, καθώς και σε μορφή PDF (συνολικά και για τις 2 ασκήσεις),
- **είτε** σε MS Word (με χρήση εξισώσεων Microsoft Equation), όπου θα μετατρέψετε το doc/docx σε PDF,
- **ή** στο χαρτί και σάρωση σε μορφή PDF, ώστε να υποβληθούν μαζί με την 1η Άσκηση σε **ένα ενιαίο PDF αρχείο** `assignment.pdf`.

Στην περίπτωση υλοποίησης με MS Word ή στο χαρτί, μπορείτε να χρησιμοποιήσετε εργαλεία όπως CombinePDF για την συνένωση των παραγόμενων PDF σε ένα αρχείο ("σημειωματάριο για την 1η άσκηση" + "σαρώσεις ή doc2pdf αρχείο για τη 2η άσκηση"). Με άλλα λόγια, συγκεντρωτικά, όλες οι απαντήσεις σας θα βρίσκονται σε **ένα** PDF αρχείο: `assignment.pdf`.

- Οι ασκήσεις είναι **ατομικές** - **δεν** επιτρέπεται η μεταξύ σας συνεργασία για την υλοποίηση/παράδοσή τους.

- Ο κώδικάς σας πρέπει να σχολιαστεί εκτενώς! Καλά σχολιασμένος κώδικας θα συνεκτιμηθεί στην αξιολόγησή σας.
- Αφού ολοκληρώσετε, εξαγάγετε το notebook σε **PDF** και υποβάλετε τόσο το `.ipynb` όσο και το `.pdf` αρχείο στο `turnin`, μαζί με ένα αρχείο `onoma.txt` (θα περιέχει το ον/μο σας και τον A.M. σας).
- Μια καλή πρακτική για την αποφυγή προβλημάτων απεικόνισης, π.χ., περικοπής εικόνων/κώδικα στα όρια της σελίδας, είναι η μετατροπή/αποθήκευση του `.ipynb` πρώτα σε **HTML** και μετά σε PDF μέσω browser.
- Οι απαντήσεις θα παραδοθούν με την εντολή:

```
turnin assignment@mye037 onoma.txt assignment.ipynb assignment.pdf
```

- Μπορείτε να χρησιμοποιήσετε βασικά πακέτα γραμμικής άλγεβρας (π.χ. `NumPy`, `SciPy`, `Matplotlib`), αλλά δεν επιτρέπεται να χρησιμοποιείτε τα πακέτα/βιβλιοθήκες που επιλύουν άμεσα τα προβλήματα, εκτός και αν αναφέρεται ρητά η χρήση συγκεκριμένου πακέτου σε κάποιο ζήτημα (π.χ. `OpenCV`). Αν δεν είστε βέβαιοι για κάποιο συγκεκριμένο πακέτο/βιβλιοθήκη ή συνάρτηση που θα χρησιμοποιήσετε, μη διστάσετε να ρωτήσετε τον διδάσκοντα.

Late Policy: Εργασίες που υποβάλλονται καθυστερημένα θα λαμβάνουν μείωση βαθμού 10% για κάθε 24 ώρες καθυστέρησης. Οι εργασίες δεν θα γίνονται δεκτές 96 ώρες (4 ημέρες) μετά την προθεσμία παράδοσης. Για παράδειγμα, παράδοση της εργασίας 2 ημέρες μετά την προθεσμία βαθμολογείται με άριστα το 24 (από 30).

2 Intro to Google Colab and Jupyter Notebook

Εισαγωγή

- Η Εργασία του μαθήματος ΜΥΕ037-Ψηφιακή Επεξεργασία Εικόνας περιλαμβάνει 2 Ασκήσεις στο αρχείο `assignment.ipynb`, το οποίο απαιτεί περιβάλλον Jupyter Notebook για προβολή και επεξεργασία, **είτε τοπικά** (local machine) στον υπολογιστή σας, **είτε μέσω της υπηρεσίας** νέφους Google Colab ή Colaboratory.

Working remotely on Google Colaboratory

Το Google Colaboratory είναι ένας συνδυασμός σημειωματαρίου Jupyter και Google Drive. Εκτελείται εξ' ολοκλήρου στο cloud και έρχεται προεγκατεστημένο με πολλά πακέτα (π.χ. PyTorch και Tensorflow), ώστε όλοι να έχουν πρόσβαση στις ίδιες εξαρτήσεις/βιβλιοθήκες. Ακόμη πιο ενδιαφέρον είναι το γεγονός ότι το Colab επωφελείται από την ελεύθερη πρόσβαση σε επιταχυντές υλικού (π.χ. κάρτες γραφικών) όπως οι GPU (K80, P100) και οι TPU.

- Requirements:

Για να χρησιμοποιήσετε το Colab, πρέπει να έχετε λογαριασμό Google με συσχετισμένο Google Drive. Υποθέτοντας ότι έχετε και τα δύο (ο ακαδημαϊκός σας λογαριασμός είναι λογαριασμός google), μπορείτε να συνδέσετε το Colab στο Drive σας με τα ακόλουθα βήματα:

1. Κάντε κλικ στον τροχό στην επάνω δεξιά γωνία (στο Google Drive) και επιλέξτε `Ρυθμίσεις`.
2. Κάντε κλικ στην καρτέλα `Διαχείριση εφαρμογών`.
3. Στο επάνω μέρος, επιλέξτε `Σύνδεση περισσότερων εφαρμογών` που θα εμφανίσουν ένα παράθυρο του `GSuite Marketplace`.

4. Αναζητήστε το `Colab` και, στη συνέχεια, κάντε κλικ στην `Προσθήκη` (install).

- Workflow:

Η εργασία στη σελίδα ecourse του μαθήματος παρέχει έναν σύνδεσμο λήψης σε ένα αρχείο `assignment.zip` που περιέχει:

1. `images/`, φάκελος με τις εικόνες `distorted_img_i.png` ($i = 1, \ldots, 7$) που έχουν υποστεί παραμόρφωση σε σχέση με τη δοθείσα εικόνα-στόχο `lena.png`.
2. `assignment.ipynb`, το σημειωματάριο jupyter στο οποίο θα εργαστείτε και θα παραδώσετε.

- Βέλτιστες πρακτικές:

Υπάρχουν μερικά πράγματα που πρέπει να γνωρίζετε όταν εργάζεστε με την υπηρεσία Colab. Το πρώτο πράγμα που πρέπει να σημειωθεί είναι ότι οι πόροι δεν είναι εγγυημένοι (αυτό είναι το τίμημα της δωρεάν χρήσης). Εάν είστε σε αδράνεια για ένα συγκεκριμένο χρονικό διάστημα ή ο συνολικός χρόνος σύνδεσής σας υπερβαίνει τον μέγιστο επιτρεπόμενο χρόνο (~12 ώρες), το Colab VM θα αποσυνδεθεί. Αυτό σημαίνει ότι οποιαδήποτε μη αποθηκευμένη πρόοδος θα χαθεί. Έτσι, φροντίστε να αποθηκεύετε συχνά την υλοποίησή σας ενώ εργάζεστε.

Working locally on your machine

Linux

Εάν θέλετε να εργαστείτε τοπικά στον Η/Υ σας, θα πρέπει να χρησιμοποιήσετε ένα εικονικό περιβάλλον. Μπορείτε να εγκαταστήσετε ένα μέσω του Anaconda (συνιστάται) ή μέσω της native μονάδας `venv` της Python, ή τέλος, μέσα από το περιβάλλον του IDE (π.χ. Pycharm). Βεβαιωθείτε ότι χρησιμοποιείτε (τουλάχιστον) έκδοση τουλάχιστον Python 3.7.

- Εικονικό περιβάλλον Anaconda: Συνιστάται η χρήση της δωρεάν διανομής Anaconda, η οποία παρέχει έναν εύκολο τρόπο για να χειριστείτε τις εξαρτήσεις πακέτων. Μόλις εγκαταστήσετε το Anaconda, είναι εύχρηστο να δημιουργήσετε ένα εικονικό περιβάλλον για το μάθημα. Για να ρυθμίσετε ένα εικονικό περιβάλλον που ονομάζεται π.χ. `mye037`, εκτελέστε τα εξής στο τερματικό σας: `conda create -n mye037 python=3.7` (Αυτή η εντολή θα δημιουργήσει το περιβάλλον mye037 στη διαδρομή 'path/to/anaconda3/envs/') Για να ενεργοποιήσετε και να εισέλθετε στο περιβάλλον, εκτελέστε το `conda activate mye037`. Για να απενεργοποιήσετε το περιβάλλον, είτε εκτελέστε `conda deactivate mye037` είτε βγείτε από το τερματικό. Σημειώστε ότι κάθε φορά που θέλετε να εργαστείτε στην εργασία, θα πρέπει να εκτελείτε ξανά το `conda activate mye037`. Ενδεχομένως θα χρειαστεί να εγκαταστήστε τις βιβλιοθήκες Numpy, SciPy, Matplotlib, OpenCV μέσω του ενεργοποιημένου περιβάλλοντος `mye037`, με τον παρακάτω τρόπο: `conda install numpy scipy matplotlib opencv` ή `conda install -c conda-forge numpy scipy matplotlib opencv`.
- Εικονικό περιβάλλον Python venv: Για να ρυθμίσετε ένα εικονικό περιβάλλον που ονομάζεται mye037, εκτελέστε τα εξής στο τερματικό σας: `python3.7 -m venv ~/mye037` Για να ενεργοποιήσετε και να εισέλθετε στο περιβάλλον, εκτελέστε το `source ~/mye037/bin/activate`. Για να απενεργοποιήσετε το περιβάλλον, εκτελέστε: `deactivate` ή πραγματοποιήστε έξοδο από το τερματικό. Σημειώστε ότι κάθε φορά που θέλετε να εργαστείτε για την άσκηση, θα πρέπει να εκτελείτε ξανά το `source ~/mye037/bin/activate`.
- Εκτέλεση Jupyter Notebook: Εάν θέλετε να εκτελέσετε το notebook τοπικά με το Jupyter, βεβαιωθείτε ότι το εικονικό σας περιβάλλον έχει εγκατασταθεί σωστά (σύμφωνα με τις οδηγίες εγκατάστασης που περιγράφονται παραπάνω για περιβάλλον linux), ενεργοποιήστε το και, στη

συνέχεια, εκτελέστε `pip install notebook` για να εγκαταστήσετε το σημειωματάριο `Jupyter`. Στη συνέχεια, αφού κατεβάσετε και αποσυμπιέσετε το φάκελο της Άσκησης από τη σελίδα ecourse σε κάποιο κατάλογο της επιλογής σας, εκτελέστε `cd` σε αυτόν το φάκελο και στη συνέχεια εκτελέστε το σημειωματάριο `jupyter notebook`. Αυτό θα πρέπει να εκκινήσει αυτόματα έναν διακομιστή notebook στη διεύθυνση `http://localhost:8888`. Εάν όλα έγιναν σωστά, θα πρέπει να δείτε μια οθόνη που θα εμφανίζει όλα τα διαθέσιμα σημειωματάρια στον τρέχοντα κατάλογο, στην προκειμένη περίπτωση μόνο το `assignment.ipynb` (η εργασία σας). Κάντε κλικ στο `assignment.ipynb` και ακολουθήστε τις οδηγίες στο σημειωματάριο.

Windows

Τα πράγματα είναι πολύ πιο απλά στην περίπτωση που θέλετε να εργαστείτε τοπικά σε περιβάλλον `Windows`. Μπορείτε να εγκαταστήσετε την Anaconda για Windows και στη συνέχεια να εκτελέσετε το Anaconda Navigator αναζητώντας το απευθείας στο πεδίο αναζήτησης δίπλα από το κουμπί **έναρξης** των Windows. Το εργαλείο αυτό παρέχει επίσης άμεσα προεγκατεστημένα, τα πακέτα λογισμικού Jupyter Notebook και JupyterLab τα οποία επιτρέπουν την προβολή και υλοποίηση του σημειωματαρίου Jupyter άμεσα και εύκολα (εκτελώντας το απευθείας από τη διαδρομή αρχείου που βρίσκεται).

2.1 Άσκηση 1: Τεχνικές Ψηφιακής Επεξεργασίας Εικόνας για την αντιμετώπιση και αποκατάσταση παραμορφώσεων [18 μονάδες]

- Ο αποσυμπιεσμένος φάκελος περιλαμβάνει το φάκελο `images/` με την επιθυμητή εικόνα-στόχο `lena.png` καθώς και τις παραμορφωμένες εκδοχές της (distorted_img_1.png, . . . , distorted_img_7.png) τις οποίες πρέπει να επιδιορθώσετε αποκαθιστώντας την αρχική εικόνα-στόχο, υλοποιώντας την κατάλληλη τεχνική αποκατάστασης, για κάθε περίπτωση.
- Θα χρειαστεί να εργαστείτε σε γλώσσα Python (έκδοση Python τουλάχιστον 3.7). Για διευκόλυνσή σας, προτείνεται να εγκαταστήσετε το εργαλείο/διαχειριστή πακέτων εικονικού περιβάλλοντος Anaconda. Επίσης, μπορείτε να εργαστείτε σε κάποιο χρήσιμο IDE για περιβάλλον Windows/Linux (π.χ. Pycharm, η οποία είναι διαθέσιμη με «free educational license», με τον ακαδημαϊκό λογαριασμό σας).
- Μπορείτε να εκτελέσετε το σημειωματάριο **τοπικά** ή να εργαστείτε σε Google Collab, μέσω των οδηγιών που περιγράφονται στην προηγούμενη ενότητα.

Θα χρησιμοποιήσετε τα ακόλουθα πακέτα σε αυτή την άσκηση:

- Numpy
- SciPy
- Matplotlib
- OpenCV

Στις παρακάτω παραμορφωμένες εκδοχές της εικόνας-στόχου `lena.png` (σε **grayscale**) έχει εφαρμοστεί μια σειρά από λειτουργίες, όπως:

- Εικόνα με προσθήκη λευκού **Gaussian θορύβου**
- Θόλωση με γραμμικό φιλτράρισμα με φίλτρο μέσου $5x5$ (**Mean Blur**).
- Εικόνα με επεκταμένη αντίθεση (**Contrast Stretching / Overstretching**)
- Εικόνα με **ανεπαρκή εξισορρόπηση ιστογράμματος (Poor Histogram Equalization)**
- Εξομάλυνση με Γκαουσσιανό φίλτρο 9x9 (**Gaussian Blur**)

- Εικόνα χαμηλής αντίθεσης (**Low Contrast**)
- Εικόνα με θόρυβο τύπου **αλατοπίπερου (Salt & Pepper)**

Για την αντιμετώπιση των παραπάνω παραμορφώσεων, θα χρειαστεί να εφαρμόσετε μια σειρά από τεχνικές, **σε άγνωστη προς εσάς αντιστοίχιση** με το πρόβλημα, όπως:

- Όξυνση (**Sharpening**)
- Αφαίρεση εξομάλυνσης (**Unsharp Masking**)
- Ισοστάθμιση ή εξισορρόπηση ιστογράμματος (**Histogram Equalization**)
- Περιορισμός εύρους τιμών μέσω σημειακών τελεστών (**Log / Power Transformations**)
- Φιλτράρισμα **Gaussian ή Median** για απομάκρυνση θορύβου
- **Καθορισμός / Ταίριασμα ιστογράμματος**

ΠΡΟΣΟΧΗ
Σε ορισμένες περιπτώσεις, ζητείται από εσάς να **υλοποιήσετε οι ίδιοι** τις κατάλληλες διαδικασίες αποκατάστασης, χρησιμοποιώντας βασικές συναρτήσεις της Python/NumPy, **χωρίς να καταφύγετε σε έτοιμες υλοποιήσεις της βιβλιοθήκης OpenCV**, εκτός αν δηλώνεται ρητά διαφορετικά στο εκάστοτε πρόβλημα.

Συγκεκριμένα, επιτρέπονται τα παρακάτω:

- **Φιλτράρισμα με φίλτρο όξυνσης (Sharpening)**: Θα χρειαστεί να ορίσετε τον κατάλληλο πυρήνα ενίσχυσης ακμών και να τον εφαρμόσετε με χρήση της ρουτίνας `scipy.signal.convolve2d()`, ή με χρήση `NumPy`. **Δεν** επιτρέπεται η χρήση `cv2.filter2D()`.
- **Καθορισμός / Ταίριασμα ιστογράμματος (Histogram Specification)**: Παρέχεται helper block που πρέπει να προσαρμοστεί στο αντίστοιχο κελί της εικόνας με κακή ισοστάθμιση ιστογράμματος.
- **Περιορισμός εύρους τιμών (Log / Power Transformations)**: Θα εφαρμόσετε σημειακούς μετασχηματισμούς με χρήση NumPy.
- **Median filtering**: Θα πρέπει να υλοποιήσετε την αντίστοιχη ρουτίνα με χρήση `NumPy`.

Επίσης, **επιτρέπεται** η χρήση των σχετικών έτοιμων συναρτήσεων από τη βιβλιοθήκη **OpenCV**, όπως:

- `cv2.GaussianBlur()` για αποθορυβοποίηση ή φιλτράρισμα με Gaussian φίλτρο.
- `cv2.equalizeHist()` για **Ισοστάθμιση ιστογράμματος**

2.1.1 Αξιολόγηση Αποτελεσμάτων

Μετά την αποκατάσταση κάθε εικόνας, μπορείτε να συγκρίνετε την αποκατεστημένη εικόνα με την αρχική `lena.png`, τόσο οπτικά/ποιοτικά, όσο και ποσοτικά, χρησιμοποιώντας δείκτες αξιολόγησης όπως MSE, PSNR και SSIM. Ένα ενδεικτικό τμήμα κώδικα (σε σχόλια) παρέχεται σε κάθε ερώτημα για να υπολογίσετε αυτούς τους δείκτες.

- **MSE (Mean Squared Error)**: Μετρά το μέσο τετράγωνο της διαφοράς pixel ανάμεσα στις εικόνες. Όσο μικρότερη η τιμή, τόσο καλύτερη η αποκατάσταση.
- **PSNR (Peak Signal-to-Noise Ratio)**: Εκφράζει την ποιότητα της εικόνας σε dB. Όσο μεγαλύτερη η τιμή της σηματοθορυβικής σχέσης, τόσο καλύτερη η αποκατάσταση.

- **SSIM (Structural Similarity Index Measure)**: Μετρά τη δομική ομοιότητα των εικόνων, λαμβάνοντας υπόψη την αντίθεση, φωτεινότητα και υφή. Τιμές κοντά στο 1 δείχνουν υψηλή ομοιότητα.

[ ]:
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
from skimage.metrics import structural_similarity as ssim
from skimage.metrics import peak_signal_noise_ratio as psnr

def show_image(title, img):
    plt.figure()
    # αυτόματο scaling στο [0,1] ή [0,255] με βάση το dtype κάθε εικόνας
    if img.dtype == np.uint8:
        vmin, vmax = 0, 255
    else:
        vmin, vmax = float(np.min(img)), float(np.max(img))

    plt.imshow(img, cmap='gray', vmin=vmin, vmax=vmax)
    plt.title(title)
    plt.axis('off')
    plt.show()
```

[ ]:
```
# Φόρτωση εικόνας στόχου
target = cv2.imread('images/lena.png', cv2.IMREAD_GRAYSCALE)

# Φόρτωση παραμορφωμένων εικόνων
distorted_images = {
    "Distorted Image #1": cv2.imread("images/distorted_img_1.png", cv2.
 ↪IMREAD_GRAYSCALE),
    "Distorted Image #2": cv2.imread("images/distorted_img_2.png", cv2.
 ↪IMREAD_GRAYSCALE),
    "Distorted Image #3": cv2.imread("images/distorted_img_3.png", cv2.
 ↪IMREAD_GRAYSCALE),
    "Distorted Image #4": cv2.imread("images/distorted_img_4.png", cv2.
 ↪IMREAD_GRAYSCALE),
    "Distorted Image #5": cv2.imread("images/distorted_img_5.png", cv2.
 ↪IMREAD_GRAYSCALE),
    "Distorted Image #6": cv2.imread("images/distorted_img_6.png", cv2.
 ↪IMREAD_GRAYSCALE),
    "Distorted Image #7": cv2.imread("images/distorted_img_7.png", cv2.
 ↪IMREAD_GRAYSCALE),
}

# Επιβεβαίωση μεγέθους εικόνων
print("Target:", target.shape)
for k, img in distorted_images.items():
```

```
    print(k, "->", img.shape)

# Εμφάνιση εικόνας στόχου
plt.figure()
plt.imshow(target, cmap='gray')
plt.title("Original-target image")
plt.axis('off')
plt.show()

# Εμφάνιση όλων των παραμορφώσεων
for title, img in distorted_images.items():
    show_image(title, img)
```

2.1.2 Helper για Histogram Specification (Καθορισμός Ιστογράμματος)

Στο πλαίσιο της άσκησης, θα χρειαστεί να εντοπίσετε ποια από τις επτά παραμορφωμένες εικόνες έχει υποστεί **ανεπαρκή ισοστάθμιση ιστογράμματος (poor histogram equalization)**. Για την περίπτωση αυτή, σας παρέχεται το παρακάτω βοηθητικό μπλοκ κώδικα που υπολογίζει την τεχνική του **καθορισμού ιστογράμματος (histogram specification)**, χρησιμοποιώντας την εικόνα-στόχο `lena.png` ως πηγή για το επιθυμητό ιστόγραμμα.

Θα χρειαστεί να **προσαρμόσετε κατάλληλα** τον κώδικα εντός του σωστού κελιού (ανάλογα με το distortion), ώστε να επιτύχετε την αποκατάσταση.

Περιγραφή Μεθόδου Η διαδικασία περιλαμβάνει:

1. **Υπολογισμό αθροιστικής κατανομής (CDF)** της παραμορφωμένης εικόνας και του reference ιστογράμματος.
2. **Εντοπισμό mapping** τιμών μέσω αντιστοίχισης των CDF: για κάθε τιμή φωτεινότητας στην παραμορφωμένη εικόνα, αντιστοιχίζεται η πλησιέστερη τιμή του reference CDF (βασισμένο στην εξίσωση CDF matching).
3. **Εφαρμογή του mapping** στην εικόνα για να δημιουργηθεί η νέα εικόνα με την επιθυμητή κατανομή φωτεινότητας.

Σημείωση: Η υλοποίηση βασίζεται σε `np.interp()` για κατασκευή του mapping πίνακα, ακολουθώντας την αρχή του αντιστρόφου μετασχηματισμού CDF. Παρά το γεγονός ότι δεν παράγεται τέλεια προσαρμογή, το αποτέλεσμα είναι κατά κανόνα σημαντικά βελτιωμένο.

[ ]:
```
# Το παρακάτω block σας βοηθά να αντιμετωπίσετε την περίπτωση παραμόρφωσης με␣
↪ανεπαρκή ισοστάθμιση ιστογράμματος.
# Μην το εκτελέσετε όπως είναι - χρησιμοποιήστε το ως οδηγό και προσαρμόστε το␣
↪σωστά στο κελί που αντιστοιχεί
# στο κατάλληλο distortion.

# === 1. Υπολογισμός reference ιστογράμματος από την αρχική εικόνα ===
# Χρησιμοποιείται ως "στόχος" για την ανακατανομή των εντάσεων
hist_ref, _ = np.histogram(target.flatten(), bins=256, range=(0, 256))
```

```
hist_ref = hist_ref.astype(np.float64) / hist_ref.sum()  # Κανονικοποίηση ώστε␣
 ↪να αναπαριστά πιθανότητες

# === 2. Συνάρτηση: Καθορισμός Ιστογράμματος μέσω CDF Matching ===
def histogram_specification(source_img, reference_hist):
    # Βήμα 1: Υπολογισμός ιστογράμματος της εισόδου
    src_hist, _ = np.histogram(source_img.flatten(), bins=256, range=(0, 256))
    src_cdf = np.cumsum(src_hist).astype(np.float64)
    src_cdf /= src_cdf[-1]  # Κανονικοποίηση στο [0, 1]

    # Βήμα 2: Υπολογισμός CDF της reference κατανομής
    ref_cdf = np.cumsum(reference_hist).astype(np.float64)
    ref_cdf /= ref_cdf[-1]

    # Βήμα 3: Δημιουργία mapping s_k -> z_q μέσω πλησιέστερης τιμής CDF
    mapping = np.interp(src_cdf, ref_cdf, np.arange(256))

    # Βήμα 4: Εφαρμογή mapping στα pixels της εικόνας
    matched = np.interp(source_img.flatten(), np.arange(256), mapping)
    return matched.reshape(source_img.shape).astype(np.uint8)

# === 3. Παράδειγμα χρήσης (συμπληρώστε με τον σωστό αριθμό εικόνας) ===
# input_img = distorted_images["Distorted Image #i"].copy()
# restored = histogram_specification(input_img, hist_ref)

# === 4. Οπτική αξιολόγηση (προαιρετικά) ===
# show_image("Distorted Image #i", input_img)
# show_image("Restored Image", restored)

# === 5. Συγκριτική απεικόνιση ιστογραμμάτων (προαιρετικά) ===
# hist_input, _ = np.histogram(input_img.flatten(), bins=256, range=(0, 256))
# hist_input = hist_input.astype(np.float64) / hist_input.sum()

# hist_restored, _ = np.histogram(restored.flatten(), bins=256, range=(0, 256))
# hist_restored = hist_restored.astype(np.float64) / hist_restored.sum()

# hist_target, _ = np.histogram(target.flatten(), bins=256, range=(0, 256))
# hist_target = hist_target.astype(np.float64) / hist_target.sum()

# plt.figure(figsize=(10, 4))
# plt.plot(hist_input, label='Distorted Image', color='gray')
# plt.plot(hist_restored, label='Restored Image', color='green', linestyle='--')
# plt.plot(hist_target, label='Original Lena', color='red', linestyle=':')
# plt.title("Overlay of Histograms")
# plt.xlabel("Intensity")
# plt.ylabel("Normalized Frequency")
# plt.legend()
```

```
# plt.grid(True)
# plt.tight_layout()
# plt.show()
```

2.1.3 Αποκατάσταση: Distorted Image #i

Παρατήρησε την παρακάτω παραμορφωμένη εικόνα κάθε περίπτωσης (για $i = 1, \ldots, 7$) και προσπάθησε να εντοπίσεις το είδος της παραμόρφωσης που έχει υποστεί. Στη συνέχεια, σχεδίασε και εφάρμοσε μια κατάλληλη τεχνική αποκατάστασης, με στόχο η παραγόμενη εικόνα να προσεγγίζει όσο το δυνατόν περισσότερο την αρχική `lena.png`.

Αξιολόγησε την ποιότητα της αποκατεστημένης εικόνας τόσο οπτικά όσο και με χρήση κατάλληλων ποσοτικών μετρικών (π.χ., PSNR, SSIM, MSE).

Σημείωση:

- Στην περίπτωση που εφαρμόζετε την τεχνική **Unsharp Masking**, επιτρέπεται η χρήση της `cv2.GaussianBlur()` αποκλειστικά για το στάδιο του θολώματος. Τα επόμενα βήματα της διαδικασίας (υπολογισμός της μάσκας και προσθήκη της στην αρχική εικόνα) πρέπει να υλοποιηθούν **χειροκίνητα**, χρησιμοποιώντας `NumPy` — **όχι** με τις έτοιμες συναρτήσεις `cv2.subtract()` ή `cv2.addWeighted()`.
- Στην περίπτωση χρήσης **ισοστάθμισης ιστογράμματος** για την αντιμετώπιση της αντίστοιχης παραμόρφωσης, παρατηρείστε την τιμή του δείκτη MSE μετά την ισοστάθμιση ιστογράμματος και **εξηγήστε** (σε σχόλια στο αντίστοιχο κελί) σε τι μπορεί να οφείλεται αυτή η διαφορά παρόλο που η εικόνα φαίνεται "καλύτερη" οπτικά;
- Για την απομάκρυνση Gaussian θορύβου, μπορείτε να χρησιμοποιήσετε Gaussian φίλτρο εξομάλυνσης είτε με NumPy/SciPy (`scipy.ndimage.gaussian_filter`) είτε με OpenCV (`cv2.GaussianBlur`).
- Στην περίπτωση **Sharpening**, απαιτείται πλήρης χειροκίνητη υλοποίηση: θα πρέπει να ορίσετε ένα φίλτρο ενίσχυσης ακμών (sharpening kernel) και να το εφαρμόσετε με χρήση 2D συνέλιξης (`NumPy` ή `scipy.signal.convolve2d()`). Η χρήση της `cv2.filter2D()` **δεν επιτρέπεται**. Στην περίπτωση, ενδεχομένως μόνο ο δείκτης MSE να παρουσιάζει βελτίωση σε σχέση με την παραμορφωμένη εικόνα για κατάλληλα επιλεγμένο φίλτρο όξυνσης. **Εξηγήστε** γιατί συμβαίνει αυτό σε σχόλια στο κατάλληλο κελί που αντιστοιχεί σε αυτή την παραμόρφωση.
- Στην περίπτωση **median filtering**, ζητείται αυτόνομη υλοποίηση, χωρίς χρήση έτοιμων συναρτήσεων από `OpenCV` ή `SciPy`. Η λογική περιλαμβάνει: Padding της εικόνας (π.χ. με αναπαραγωγή άκρων `np.pad(img, pad, mode='edge')`), ανάκτηση των παραθύρων (π.χ. 3x3 ή 5x5, με κατάλληλη δεικτοδότηση του padded window) και υπολογισμός διαμέσου (`np.median`) για το φιλτράρισμα.
- Σε περιπτώσεις όπου απαιτούνται **σημειακοί μετασχηματισμοί** (όπως `np.power` ή `np.log1p`), **θα πρέπει να δοκιμάσετε περισσότερους από έναν** μετασχηματισμούς και να αξιολογήσετε την ποιότητα της αποκατάστασης, τόσο οπτικά όσο και ποσοτικά (PSNR, SSIM), προκειμένου να επιλέξετε την πιο κατάλληλη λύση. Υπόδειξη: Εξετάστε το ιστόγραμμα της παραμορφωμένης εικόνας ώστε να εντοπίσετε σε ποια περιοχή του φάσματος (σκοτεινά ή φωτεινά) συγκεντρώνονται οι περισσότερες εντάσεις. Αυτό θα σας βοηθήσει να επιλέξετε τον κατάλληλο μετασχηματισμό και τις αντίστοιχες τιμές παραμέτρων (c, γ).

[ ]:
```
## Αποκατάσταση: Distorted Image #1

input_img = distorted_images["Distorted Image #1"].copy()

# === Πριν την αποκατάσταση: Αξιολόγηση "distorted" ===
mse_distorted = np.mean((target - input_img) ** 2)
psnr_distorted = psnr(target, input_img)
ssim_distorted = ssim(target, input_img)
print("Before Restoration:")
print("MSE:", mse_distorted)
print("PSNR:", psnr_distorted)
print("SSIM:", ssim_distorted)

# === Εφαρμογή τεχνικών αποκατάστασης ===
# restored = ...

# === Οπτική αξιολόγηση ===
show_image("Distorted Image #1", input_img)
# show_image("Restored Image", restored)

# === Ποσοτική αξιολόγηση μετά την αποκατάσταση (με χρήση της lena.jpg ως ground␣
↪truth) ===
# mse_score = np.mean((target - restored) ** 2)
# psnr_score = psnr(target, restored)
# ssim_score = ssim(target, restored)

# print("MSE:", mse_score)
# print("PSNR:", psnr_score)
# print("SSIM:", ssim_score)
```

[ ]:
```
## Αποκατάσταση: Distorted Image #2

input_img = distorted_images["Distorted Image #2"].copy()

# === Πριν την αποκατάσταση: Αξιολόγηση "distorted" ===
mse_distorted = np.mean((target - input_img) ** 2)
psnr_distorted = psnr(target, input_img)
ssim_distorted = ssim(target, input_img)
print("Before Restoration:")
print("MSE:", mse_distorted)
print("PSNR:", psnr_distorted)
print("SSIM:", ssim_distorted)

# === Εφαρμογή τεχνικών αποκατάστασης ===
# restored = ...

# === Οπτική αξιολόγηση ===
```

```
show_image("Distorted Image #2", input_img)
# show_image("Restored Image", restored)

# === Ποσοτική αξιολόγηση μετά την αποκατάσταση (με χρήση της lena.jpg ως ground truth) ===
# mse_score = np.mean((target - restored) ** 2)
# psnr_score = psnr(target, restored)
# ssim_score = ssim(target, restored)

# print("MSE:", mse_score)
# print("PSNR:", psnr_score)
# print("SSIM:", ssim_score)
```

[ ]:
```
## Αποκατάσταση: Distorted Image #3

input_img = distorted_images["Distorted Image #3"].copy()

# === Πριν την αποκατάσταση: Αξιολόγηση "distorted" ===
mse_distorted = np.mean((target - input_img) ** 2)
psnr_distorted = psnr(target, input_img)
ssim_distorted = ssim(target, input_img)
print("Before Restoration:")
print("MSE:", mse_distorted)
print("PSNR:", psnr_distorted)
print("SSIM:", ssim_distorted)

# === Εφαρμογή τεχνικών αποκατάστασης ===
# restored = ...

# === Οπτική αξιολόγηση ===
show_image("Distorted Image #3", input_img)
# show_image("Restored Image", restored)

# === Ποσοτική αξιολόγηση μετά την αποκατάσταση (με χρήση της lena.jpg ως ground truth) ===
# mse_score = np.mean((target - restored) ** 2)
# psnr_score = psnr(target, restored)
# ssim_score = ssim(target, restored)

# print("MSE:", mse_score)
# print("PSNR:", psnr_score)
# print("SSIM:", ssim_score)
```

[ ]:
```
## Αποκατάσταση: Distorted Image #4

input_img = distorted_images["Distorted Image #4"].copy()
```

```
# === Πριν την αποκατάσταση: Αξιολόγηση "distorted" ===
mse_distorted = np.mean((target - input_img) ** 2)
psnr_distorted = psnr(target, input_img)
ssim_distorted = ssim(target, input_img)
print("Before Restoration:")
print("MSE:", mse_distorted)
print("PSNR:", psnr_distorted)
print("SSIM:", ssim_distorted)

# === Εφαρμογή τεχνικών αποκατάστασης ===
# restored = ...

# === Οπτική αξιολόγηση ===
show_image("Distorted Image #4", input_img)
# show_image("Restored Image", restored)

# === Ποσοτική αξιολόγηση μετά την αποκατάσταση (με χρήση της lena.jpg ως ground truth) ===
# mse_score = np.mean((target - restored) ** 2)
# psnr_score = psnr(target, restored)
# ssim_score = ssim(target, restored)

# print("MSE:", mse_score)
# print("PSNR:", psnr_score)
# print("SSIM:", ssim_score)
```

[ ]:
```
## Αποκατάσταση: Distorted Image #5

input_img = distorted_images["Distorted Image #5"].copy()

# === Πριν την αποκατάσταση: Αξιολόγηση "distorted" ===
mse_distorted = np.mean((target - input_img) ** 2)
psnr_distorted = psnr(target, input_img)
ssim_distorted = ssim(target, input_img)
print("Before Restoration:")
print("MSE:", mse_distorted)
print("PSNR:", psnr_distorted)
print("SSIM:", ssim_distorted)

# === Εφαρμογή τεχνικών αποκατάστασης ===
# restored = ...

# === Οπτική αξιολόγηση ===
show_image("Distorted Image #5", input_img)
# show_image("Restored Image", restored)
```

```
# === Ποσοτική αξιολόγηση μετά την αποκατάσταση (με χρήση της lena.jpg ως ground truth) ===
# mse_score = np.mean((target - restored) ** 2)
# psnr_score = psnr(target, restored)
# ssim_score = ssim(target, restored)

# print("MSE:", mse_score)
# print("PSNR:", psnr_score)
# print("SSIM:", ssim_score)
```

[ ]:
```
## Αποκατάσταση: Distorted Image #6

input_img = distorted_images["Distorted Image #6"].copy()

# === Πριν την αποκατάσταση: Αξιολόγηση "distorted" ===
mse_distorted = np.mean((target - input_img) ** 2)
psnr_distorted = psnr(target, input_img)
ssim_distorted = ssim(target, input_img)
print("Before Restoration:")
print("MSE:", mse_distorted)
print("PSNR:", psnr_distorted)
print("SSIM:", ssim_distorted)

# === Εφαρμογή τεχνικών αποκατάστασης ===
# restored = ...

# === Οπτική αξιολόγηση ===
show_image("Distorted Image #6", input_img)
# show_image("Restored Image", restored)

# === Ποσοτική αξιολόγηση μετά την αποκατάσταση (με χρήση της lena.jpg ως ground truth) ===
# mse_score = np.mean((target - restored) ** 2)
# psnr_score = psnr(target, restored)
# ssim_score = ssim(target, restored)

# print("MSE:", mse_score)
# print("PSNR:", psnr_score)
# print("SSIM:", ssim_score)
```

[ ]:
```
## Αποκατάσταση: Distorted Image #7

input_img = distorted_images["Distorted Image #7"].copy()

# === Πριν την αποκατάσταση: Αξιολόγηση "distorted" ===
mse_distorted = np.mean((target - input_img) ** 2)
psnr_distorted = psnr(target, input_img)
```

```
ssim_distorted = ssim(target, input_img)
print("Before Restoration:")
print("MSE:", mse_distorted)
print("PSNR:", psnr_distorted)
print("SSIM:", ssim_distorted)

# === Εφαρμογή τεχνικών αποκατάστασης ===
# restored = ...

# === Οπτική αξιολόγηση ===
show_image("Distorted Image #7", input_img)
# show_image("Restored Image", restored)

# === Ποσοτική αξιολόγηση μετά την αποκατάσταση (με χρήση της lena.jpg ως ground␣
↪truth) ===
# mse_score = np.mean((target - restored) ** 2)
# psnr_score = psnr(target, restored)
# ssim_score = ssim(target, restored)

# print("MSE:", mse_score)
# print("PSNR:", psnr_score)
# print("SSIM:", ssim_score)
```

2.1.4 BONUS [3 μονάδες]:

Προσπαθήστε να προσδιορίσετε τι είδους παραμόρφωση έχει υποστεί κάθε εικόνα (`Distorted Image #1` έως `#7`), όπως:

- blur (θόλωση)
- noise (θόρυβος)
- contrast (αντίθεση)
- ισοστάθμιση ιστογράμματος (ιστογραμματική αλλοίωση)

και **τεκμηριώστε** την εκτίμησή σας με βάση οπτικά ή/και ποσοτικά χαρακτηριστικά.

(Συμπληρώστε τον παρακάτω πίνακα)

Distorted Image	Εκτίμηση Παραμόρφωσης	Σύντομη Τεκμηρίωση
#1		
#2		
#3		
#4		
#5		
#6		
#7		

2.2 Άσκηση 2: Θεωρητικά ζητήματα ψηφιακής επεξεργασίας εικόνας [12 μονάδες]

2.2.1 Οδηγίες Παράδοσης:

- Μπορείτε να γράψετε τη λύση **με το χέρι στο χαρτί** και να την υποβάλετε **σαρωμένη σε PDF**. Θα πρέπει να συνενώσετε το PDF με αυτό που θα παραχθεί από την εξαγωγή του notebook σε PDF (για την 1η Άσκηση), σε ενιαίο αρχείο (βλ. Ενότητα: Γενικές Οδηγίες).
- Εναλλακτικά, μπορείτε να γράψετε τη λύση σε **Microsoft Word** (με χρήση εξισώσεων), και να την υποβάλετε ως **PDF**. Πάλι θα πρέπει να συνενώσετε το παραγόμενο PDF με αυτό της 1ης Άσκησης σε ενιαίο αρχείο (βλ. Ενότητα: Γενικές Οδηγίες).
- Διαφορετικά, μπορείτε να συμπληρώσετε **Markdown ή LaTeX** στα παρακάτω κελιά και να υποβάλετε το εξαγόμενο **PDF** συνολικά και για τις 2 Ασκήσεις.

2.2.2 Ζήτημα 1 [2 μονάδες]:

Εξηγήστε γιατί η μέθοδος διακριτής ισοστάθμισης ιστογράμματος δεν οδηγεί, γενικά, σε επίπεδο (flat) ιστόγραμμα.

Απάντηση:

2.2.3 Ζήτημα 2 [2 μονάδες]:

Σε μια εφαρμογή, εφαρμόζεται ένας πυρήνας εξομάλυνσης (smoothing kernel) σε εικόνες εισόδου για μείωση θορύβου, και στη συνέχεια εφαρμόζεται ένας πυρήνας Laplacian για ενίσχυση των λεπτομερειών.
Θα παίρναμε το ίδιο αποτέλεσμα αν αντιστρέφαμε τη σειρά των δύο αυτών λειτουργιών;

Απάντηση:

2.2.4 Ζήτημα 3 [4 μονάδες]:

Να δείξετε ότι ο Λαπλασιανός τελεστής (Laplacian), όπως ορίζεται στην Εξίσωση (3-50) του βιβλίου, είναι ισοτροπικός (δηλαδή αμετάβλητος ως προς την περιστροφή των συντεταγμένων). Υποθέστε συνεχή μεγέθη. Σύμφωνα με τον Πίνακα 2.3 (σελίδα 71), η περιστροφή των συντεταγμένων κατά γωνία θ δίνεται από:

$x' = x\cos\theta - y\sin\theta$ και $y' = x\sin\theta + y\cos\theta$

όπου ((x, y)) και ((x', y')) είναι οι μη περιστραμμένες και οι περιστραμμένες συντεταγμένες, αντίστοιχα.

Υπόδειξη:

- Υπολογίστε τον Λαπλασιανό τελεστή $\nabla^2 f$ στις αρχικές συντεταγμένες (x, y)
- Χρησιμοποιήστε τον κανόνα αλυσίδας για να εκφράσετε τις παραγώγους ως προς (x', y')
- Δείξτε ότι το αποτέλεσμα για τον $\nabla^2 f$ είναι ίδιο και στις δύο περιπτώσεις (δηλ. είναι αναλλοίωτο υπό περιστροφή).

Απάντηση:

2.2.5 Ζήτημα 4 [4 μονάδες]:

Αποδείξτε το θεώρημα συνέλιξης (convolution theorem) για μία συνεχώς ορισμένη μεταβλητή, όπως δίνεται στις Εξισώσεις (4-25) και (4-26) του βιβλίου.

Υπόδειξη:

Ξεκινήστε από τον ορισμό της συνεχούς συνέλιξης σε μία διάσταση:

$(f * g)(x) = \int_{-\infty}^{+\infty} f(\alpha)g(x-\alpha)\,d\alpha$

Υπολογίστε το μετασχηματισμό Fourier του ολοκληρώματος και χρησιμοποιήστε τις βασικές ιδιότητες της μετατόπισης και της συνέλιξης στον χώρο Fourier.

Απάντηση:

2.3 Οδηγίες υποβολής

Μην ξεχάσετε να κάνετε turnin το αρχείο Jupyter notebook **και** το PDF αρχείο αυτού του notebook μαζί με το συνοδευτικό αρχείο `onoma.txt`:

turnin assignment@mye037 onoma.txt assignment.ipynb assignment.pdf

Στην περίπτωση που δεν απαντήσετε επάνω στο σημειωματάριο για τη **2η Άσκηση**, θα πρέπει να μετατρέψετε την απάντησή σας σε PDF και να συνενώσετε (merge - CombinePDF) τα παραγόμενα PDF του σημειωματαρίου και αυτό της 2ης Άσκησης σε **ένα τελικό αρχείο** `assignment.pdf`.

Βεβαιωθείτε ότι το περιεχόμενο σε **κάθε κελί εμφανίζεται** καθαρά στο τελικό σας αρχείο PDF. Για να μετατρέψετε το σημειωματάριο σε PDF, μπορείτε να επιλέξετε **έναν** από τους παρακάτω τρόπους:

1. Google Colab (Συνιστάται): You can `print` the web page and save as PDF (e.g. Chrome: Right click the web page → Print... → Choose "Destination: Save as PDF" and click "Save"). Προσοχή στην περίπτωση όπου κώδικας/σχόλια εμφανίζονται εκτός των ορίων της σελίδας. Μια λύση είναι η αλλαγή γραμμής π.χ. σε σχόλια που υπερβαίνουν το πλάτος της σελίδας.

- Στην περίπτωση που οι εικόνες εξόδου δεν εμφανίζονται σωστά, μια λύση μέσω colab είναι (εργαλείο nbconvert):
  - Ανέβασμα του αρχείου `assignment.ipynb` στο home directory του Colaboratory (ο κατάλογος home είναι: /content/).
  - Εκτελέστε σε ένα κελί colab ενός νέου notebook: `!jupyter nbconvert --to html /content/assignment.ipynb`
  - Κάνετε λήψη του assignment.html τοπικά στον υπολογιστή σας και ανοίξτε το αρχείο μέσω browser ώστε να το εξάγετε ως PDF.

2. Local Jupyter/JupyterLab: You can `print` the web page and save as PDF (File → Print... → Choose "Destination: Save as PDF" and click "Save"). Προσοχή στην περίπτωση όπου κώδικας/σχόλια εμφανίζονται εκτός των ορίων της σελίδας. Μια λύση είναι η αλλαγή γραμμής π.χ. σε σχόλια που υπερβαίνουν το πλάτος της σελίδας.

3. Local Jupyter/JupyterLab(**Συνιστάται!**): You can `export` and save as **HTML** (File → Save & Export Notebook as... → HTML). Στη συνέχεια μπορείτε να μετατρέψετε το HTML αρχείο αποθηκεύοντάς το ως PDF μέσω ενός browser.