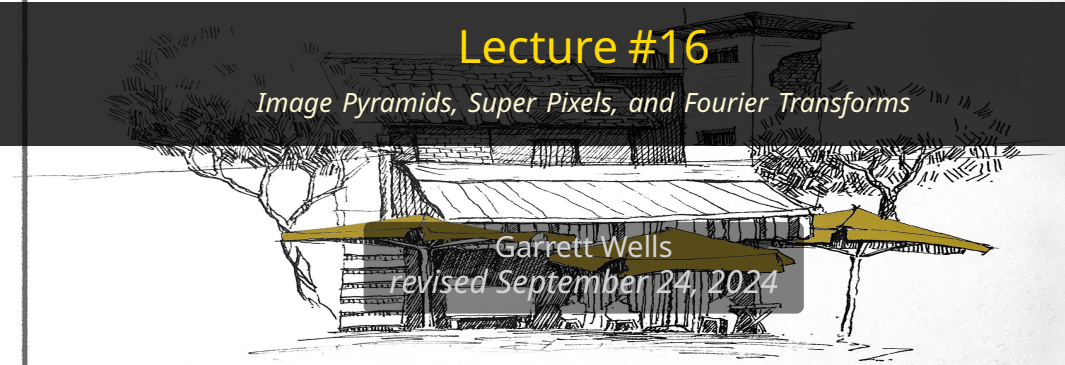
Lecture #16

Image Pyramids, Super Pixels, and Fourier Transforms



Garrett Wells

revised September 24, 2024

Table of Contents I

Table of Contents I

Table of Contents

Review

Super Pixels

Image Pyramids
Gaussian Pyramid
Laplacian Pyramid

Fourier Transforms

Review

Motion Detection

- Motion Size Estimation
- Motion Localization
- Motion Reporting/Visualization

Review
Motion Detection
- Motion Size Estimation
- Motion Localization
- Motion Reporting/Visualization

Super Pixels

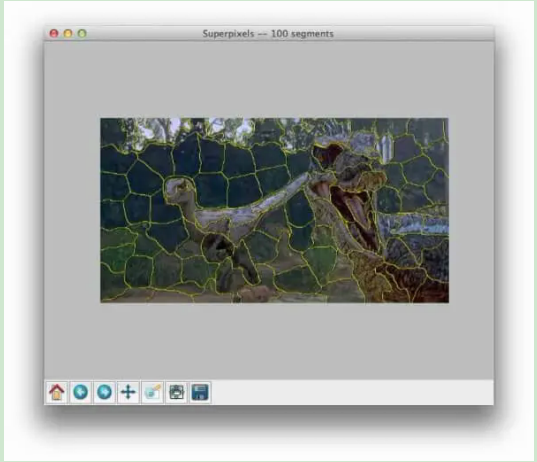
- Goal: *Segment an image into areas of similar color.*
- Segmented areas with similar color are “super pixels”
- Typically uses a clustering algorithm

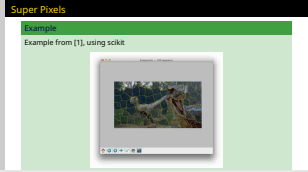
Super Pixels
- Goal: *Segment an image into areas of similar color.*
- Segmented areas with similar color are “super pixels”
- Typically uses a clustering algorithm

Super Pixels

Example

Example from [1], using scikit





Super Pixels

Benefits:

1. Computational Efficiency
2. Perceptual Significance
3. Oversegmentation
4. Graphing

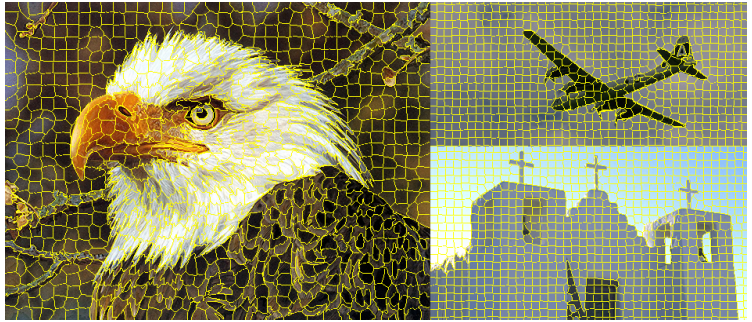


Figure: OpenCV Docs Example [2]

Super Pixels
Benefits:
1. Computational Efficiency
2. Perceptual Significance
3. Oversegmentation
4. Graphing

Figure: OpenCV Docs Example [2]

1. reduces the complexity of the image…
2. pixels grouped together by due to commonalities such as color or texture patterns
3. *Oversegmentation* means that important boundaries in the image are found, but some may be excessive. This is beneficial, as it means no pixels are lost.
4. Instead of constructing graphs which are thousands of pixels, we can graph between super pixels which already contain connected/related pixels. Much more efficient.

Super Pixels in OpenCV

Documentation [2]

```
# install opencv-contrib-python first
#   pip install opencv-contrib-python
# Linear Spectral Sampling
cv.ximgproc.createSuperpixelLSC(image,           # input image, 3
                                region_size=10, # choose average
                                ratio=0.075f    # enforces compac
)

# Superpixels Extracted via Energy Driven Sampling
cv.ximgproc.createSuperpixelSEEDS(...)
# Simple Linear Iterative Clustering
cv.ximgproc.createSuperpixelSLIC(...)
```

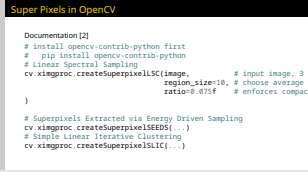



Image Pyramids

- Uses:
  - object detection
  - upscaling/downscaling an image

Image Pyramids
- Uses:
  - object detection
  - upscaling/downscaling an image

Image Pyramids

- Creates a “pyramid” of images at different resolutions [3]
- Highest resolution at bottom, lowest on top
- Different Construction Methods:
  - Gaussian
  - Laplacian

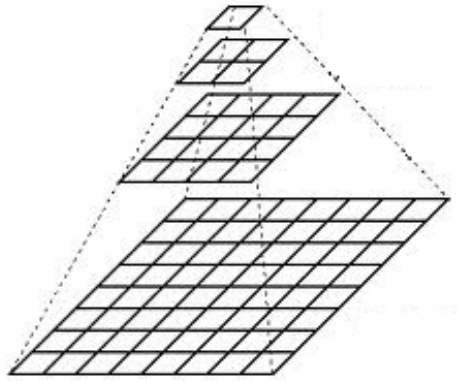
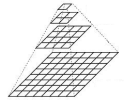


Image Pyramids
- Creates a “pyramid” of images at different resolutions [3]
- Highest resolution at bottom, lowest on top
- Different Construction Methods:
  - Gaussian
  - Laplacian



Gaussian Pyramid

Pyramid Down Algorithm:

1. Start with current resolution, $M \times N$
2. Construct next "Octave" (image with $1/4$ current resolution)
   - 2.1 Allocate image with dimensions, $M/2 \times N/2$.
   - 2.2 Fill each pixel in Octave by sampling 5 pixels from level below with Gaussian weights.
   - 2.3 Repeat from 2 until target resolution reached.

Gaussian Pyramid

Pyramid Down Algorithm:

1. Start with current resolution, $M \times N$
2. Construct next "Octave" (image with $1/4$ current resolution)
   - 2.1 Allocate image with dimensions, $M/2 \times N/2$.
   - 2.2 Fill each pixel in Octave by sampling 5 pixels from level below with Gaussian weights.
   - 2.3 Repeat from 2 until target resolution reached.

Gaussian Pyramid

Pyramid Down Algorithm:

1. Start with current resolution, $M \times N$
2. Construct next "Octave" (image with $1/4$ current resolution)
   - 2.1 Allocate image with dimensions, $M/2 \times N/2$.
   - 2.2 Fill each pixel in Octave by sampling 5 pixels from level below with Gaussian weights.
   - 2.3 Repeat from 2 until target resolution reached.

- Why bother?

Gaussian Pyramid

Pyramid Down Algorithm:

1. Start with current resolution, $M \times N$
2. Construct next "Octave" (image with $1/4$ current resolution)
   - 2.1 Allocate image with dimensions, $M/2 \times N/2$.
   - 2.2 Fill each pixel in Octave by sampling 5 pixels from level below with Gaussian weights.
   - 2.3 Repeat from 2 until target resolution reached.

- Why bother?

Gaussian Pyramid

Pyramid Down Algorithm:

1. Start with current resolution, $M \times N$
2. Construct next "Octave" (image with $1/4$ current resolution)
   - 2.1 Allocate image with dimensions, $M/2 \times N/2$.
   - 2.2 Fill each pixel in Octave by sampling 5 pixels from level below with Gaussian weights.
   - 2.3 Repeat from 2 until target resolution reached.

- Why bother?
  - ☞ Allows us to check image at each size for present features.

Gaussian Pyramid

Pyramid Down Algorithm:

1. Start with current resolution, $M \times N$
2. Construct next "Octave" (image with $1/4$ current resolution)
   - 2.1 Allocate image with dimensions, $M/2 \times N/2$.
   - 2.2 Fill each pixel in Octave by sampling 5 pixels from level below with Gaussian weights.
   - 2.3 Repeat from 2 until target resolution reached.

- Why bother?
  - ☞ Allows us to check image at each size for present features.

Gaussian Pyramid in OpenCV

```
output_img = cv.pyrDown(src,             # input image
           dstsize=Size(),               # output img size
           borderType=cv.BORDER_DEFAULT # how to fill in edges
)
# same parameters here
cv.pyrUp(...)
```


Gaussian Pyramid in OpenCV

output_img = cv.pyrDown(src, # input image
dstsize=Size(), # output img size
borderType=cv.BORDER_DEFAULT # how to fill in edges
)
# same parameters here
cv.pyrUp(...)

Laplacian Pyramid

- Formed from Gaussian pyramids
- Produces edge image
- Formed by taking difference between current level in Gaussian, and the expanded upper level of Gaussian
- No OpenCV function

Laplacian Pyramid
- Formed from Gaussian pyramids
- Produces edge image
- Formed by taking difference between current level in Gaussian, and the expanded upper level of Gaussian
- No OpenCV function

Fourier Transforms

- Image is composed of signals
- Pixel values map to frequencies [4]
- Applications
  1. Filtering by frequency
  2. Edge Detection
  3. Blurring/Sharpening
  4. Rotation
  5. *etc.*

Fourier Transforms: 3B1B

Fourier Transforms
- Image is composed of signals
- Pixel values map to frequencies [4]
- Applications
  1. Filtering by frequency
  2. Edge Detection
  3. Blurring/Sharpening
  4. Rotation
  5. *etc.*
Fourier Transforms: 3B1B

Fourier Transforms in OpenCV

```
# discrete fourier transform
cv.dft(src,            # input image/array 1D or 2D
       flags=0,        # transformation flags
       nonzeroRows=0   # assumes # rows of nonzero data
)
```


Fourier Transforms in OpenCV

```
# discrete fourier transform
cv.dft(src,            # input image/array 1D or 2D
       flags=0,        # transformation flags
       nonzeroRows=0   # assumes # rows of nonzero data
)
```

Bibliography I

[1] A. Rosebrock, “Segmentation: A SLIC Superpixel Tutorial using Python,” PyImageSearch. (Jul. 28, 2014), [Online]. Available: https://pyimagesearch.com/2014/07/28/a-slic-superpixel-tutorial-using-python/ (visited on 09/24/2024).

[2] “OpenCV: Superpixels,” (), [Online]. Available: https://docs.opencv.org/3.4/df/d6c/group__ximgproc__superpixel.html (visited on 09/24/2024).

[3] “OpenCV: Image Pyramids,” (), [Online]. Available: https://docs.opencv.org/4.x/d5/d0f/tutorial_js_pyramids.html (visited on 09/24/2024).

Bibliography I

[1] A. Rosebrock, “Segmentation: A SLIC Superpixel Tutorial using Python,” PyImageSearch. (Jul. 28, 2014), [Online]. Available: https://pyimagesearch.com/2014/07/28/a-slic-superpixel-tutorial-using-python/ (visited on 09/24/2024).

[2] “OpenCV: Superpixels,” (), [Online]. Available: https://docs.opencv.org/3.4/df/d6c/group__ximgproc__superpixel.html (visited on 09/24/2024).

[3] “OpenCV: Image Pyramids,” (), [Online]. Available: https://docs.opencv.org/4.x/d5/d0f/tutorial_js_pyramids.html (visited on 09/24/2024).

Bibliography II

[4] “OpenCV 3 Image Fourier Transform : OpenCV FFT & DFT - 2020,” (), [Online]. Available: https://www.bogotobogo.com/python/OpenCV_Python/python_opencv3_Image_Fourier_Transform_FFT_DFT.php (visited on 09/24/2024).

Bibliography II

[4] “OpenCV 3 Image Fourier Transform : OpenCV FFT & DFT - 2020,” (), [Online]. Available: https://www.bogotobogo.com/python/OpenCV_Python/python_opencv3_Image_Fourier_Transform_FFT_DFT.php (visited on 09/24/2024).