University of Idaho
Department of Computer Science
Coeur d'Alene

Lecture #17

Object Detection I

Garrett Wells
revised September 26, 2024

Table of Contents I

Review

Object Detection

Object Detection

Detecting the presence of an object of interest in an image. Difficult, but applicable in most industries.

Object Detection

Object Detection

Detecting the presence of an object of interest in an image. Difficult, but applicable in most industries.

- Vehicle counting

Object Detection

Object Detection

Detecting the presence of an object of interest in an image. Difficult, but applicable in most industries.

- Vehicle counting
- Security

Object Detection

Object Detection

Detecting the presence of an object of interest in an image. Difficult, but applicable in most industries.

- Vehicle counting
- Security
- Animal tracking

Object Detection

Object Detection

Detecting the presence of an object of interest in an image. Difficult, but applicable in most industries.

- Vehicle counting
- Security
- Animal tracking
- Sports

Object Detection

Object Detection

Detecting the presence of an object of interest in an image. Difficult, but applicable in most industries.

- Vehicle counting
- Security
- Animal tracking
- Sports
- Photography

Object Detection

- AI techniques are most common right now

Object Detection

- AI techniques are most common right now
- Can still accomplish this at a rudimentary level with image processing

Object Detection Methods

1. Histograms
2. Moments/Hu Moments
3. Hough Transforms
4. Optical Flow
5. Harris Corners
6. Keypoints and Descriptors
7. Cascade Classifiers
8. Image Stitching

Histograms

- Remember...

- Crude object detection

Histograms

- Remember. . .
  - Histograms store pixel counts by value

- Crude object detection

Histograms

- Remember...
  - Histograms store pixel counts by value
  - Object detection via histogram comparison
- Crude object detection

Histograms

- Remember...
  - Histograms store pixel counts by value
  - Object detection via histogram comparison
  - More effective when searching for small regions that match
- Crude object detection

Histograms

- Remember. . .
  - Histograms store pixel counts by value
  - Object detection via histogram comparison
  - More effective when searching for small regions that match
- Crude object detection
- Pitfalls. . .
  - feature recognition often requires *value* & *position* information
  - histograms only track number of x pixels

Histograms

- Remember...
  - Histograms store pixel counts by value
  - Object detection via histogram comparison
  - More effective when searching for small regions that match
- Crude object detection
- Pitfalls...
  - feature recognition often requires *value* & *position* information
  - histograms only track number of *x* pixels
  - fooled by changes in **size**, intensity, *etc.*

Histograms

- Remember…
  - Histograms store pixel counts by value
  - Object detection via histogram comparison
  - More effective when searching for small regions that match
- Crude object detection
- Pitfalls…
  - feature recognition often requires *value* & *position* information
  - histograms only track number of *x* pixels
  - fooled by changes in size, **intensity**, *etc.*

Moments & Hu Moments

- Uses weighted averages of pixel intensities
- Invariant to scale, rotation, *etc.*
- Calculating Hu moments give us features that we can detect/recognize

Hough Transforms

- Remember...
  - Use polar coordinates
  - Calculate all possible lines through point
  - Lines with sufficient votes (pixels) are recognized
  - Can find circles & lines

Optical Flow

- Tracking pixel motion
- Using taylor series expansion and image gradients
- see [1]

Optical Flow

Video Editors. . .

- Compression
- Stabilization
- Slow Motion

Others. . .

- Action Recognition
- Real-time Tracking

Optical Flow

Math...

$$I(x, y, t) = I(x + \Delta_x, y + \Delta_y, t + \Delta_t) \tag{1}$$

1. Assume $\Delta_t = 1$, *i.e.* frame count changes by one
2. Calculate Δ_x, Δ_y, the “motion vector”
   - Assume intensity remains the same

Optical Flow

Sparse Optical Flow...

Computes the motion vector for a specific set of objects [1]

1. Preprocess image.
2. Extract features (*ex.* corners).
3. Calculate motion vector for pixels in features.

Optical Flow

Dense Optical Flow…

Calculate a motion vector for *every* pixel in the image.

May be mapped to color space for visualization.

Sparse Optical Flow

Algorithms in OpenCV...

- Pyramid Lucas-Kanade
- Sparse RLOF

Sparse Optical Flow

Lucas-Kanade

- From [2]
- Assume nearby pixels have the same displacement direction
- Fixed size window to create a system of equations (9 equations)
- Least squares fit to get answer
- Image Pyramids to deal with large motion

Dense Optical Flow

- Tracking all pixels
- Algorithms options abound...
  1. Dense Pyramid Lucas-Kanade
  2. Farneback
  3. PCAFlow
  4. RLOF
  5. DeepFlow
  6. DualTVL1
  7. ...

Dense Optical Flow in OpenCV

```
# returns flow image, type CV_32FC2
flow = cv.calcOpticalFlowFarneback(
    prev,      # first 8-bit single channel input img
    next,      # second input image
    flow,      # output flow image, set None
    pyr_scale,# specify the image scale, 0.5 is classic pyramid
    levels,    # number of pyramid layers, 1 is none
    winsize,   # averaging window size
    iterations, # iterations @ each pyramid level
    poly_n,    # pixel neighborhood size
    poly_sigma, # standard dev. of Gaussian used for smoothing
    flags      # operation flags
    )
```

Some Comments on Features

Let's pause...

Features

Any unique pattern, shape, line, or other information which allows us to identify an object.

- How do you put together a jigsaw puzzle?

Some Comments on Features

Let's pause...

Figure: Features [3]

Some Comments on Features

Let's pause...

- Ex: Corners...
  - "Maximum variation when moved (by a small amount) in all regions around it" [3]

Feature Description

When we observe a feature in an image, we describe it using words. For example, "the top right corner of the building" or "the leaves of the tree next to the sidewalk".

Computers also describe the region around the feature to assist matching with other images.

Harris Corners

Harris Corners

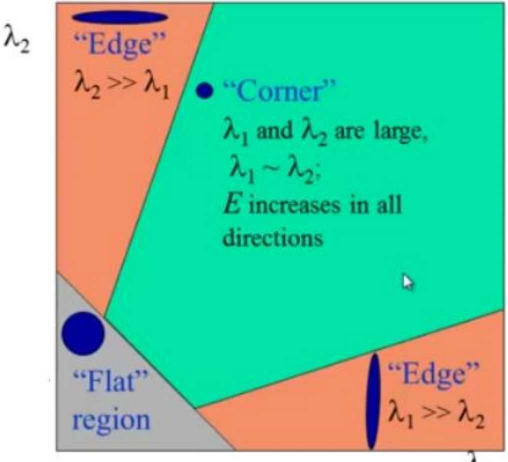
"Finds the difference in intensity for a displacement of (u, v) in all directions." [4]

$$E(u,v) = \sum_{x,y} w(x,y)[I(x+u, y+v) - I(x,y)]^2 \tag{1}$$

$w(x,y)$	window function
$I(x+u, y+v)$	shifted intensity
$I(x,y)$	intensity

Harris Corners

1. Calculate $E(u, v)$ in windows across image.
2. Maximize $E(u, v)$
3. Score to determine if window contains a corner.



Harris Corners in OpenCV

```
# returns a image with harris detector responses
output = cv.cornerHarris(
    src,          # input image, single channel 8-bit or fp
    blockSize,    # neighborhood size
    ksize,        # aperture parameter for sobel operator
    k,            # Harris detector free param
    borderType=BORDER_DEFAULT
)

# Note use cv.subPix(...) to refine Harris corners accuracy
```

Bibliography I

[1] "Optical Flow in OpenCV (C++/Python) | LearnOpenCV #," (Jan. 4, 2021), [Online]. Available: https://learnopencv.com/optical-flow-in-opencv/ (visited on 09/26/2024).

[2] "OpenCV: Optical Flow," (), [Online]. Available: https://docs.opencv.org/4.10.0/d4/dee/tutorial_optical_flow.html (visited on 09/26/2024).

[3] "Understanding Features — OpenCV-Python Tutorials beta documentation," (), [Online]. Available: https://opencv24-python-tutorials.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_features_meaning/py_features_meaning.html (visited on 09/26/2024).

Bibliography II

[4] “Harris Corner Detection — OpenCV-Python Tutorials beta documentation,” (), [Online]. Available: `https://opencv24-python-tutorials.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_features_harris/py_features_harris.html` (visited on 09/26/2024).