Visuelle Perzeption für Mensch-Maschine Schnittstellen

Vorlesung, WS 2009

Prof. Dr. Rainer Stiefelhagen
Dr. Edgar Seemann

Institut für Anthropomatik
Universität Karlsruhe (TH)

http://cvhci.ira.uka.de
rainer.stiefelhagen@kit.edu
seemann@pedestrian-detection.com
Computer Vision:

Tasks, Challenges, Performance measures

WS 2008/09

Dr. Edgar Seemann

seemann@pedestrian-detection.com
<table>
<thead>
<tr>
<th>Termine</th>
<th>Thema</th>
</tr>
</thead>
<tbody>
<tr>
<td>19.10.2009</td>
<td>Introduction, Applications</td>
</tr>
<tr>
<td>23.10.2009</td>
<td>Basics: Cameras, Transformations, Color</td>
</tr>
<tr>
<td>26.10.2009</td>
<td>Basics: Image Processing</td>
</tr>
<tr>
<td>30.10.2009</td>
<td>Basics: Pattern recognition</td>
</tr>
<tr>
<td>02.11.2009</td>
<td><strong>Computer Vision: Tasks, Challenges, Learning, Performance measures</strong></td>
</tr>
<tr>
<td>06.11.2009</td>
<td>Face Detection I: Color, Edges (Birchfield)</td>
</tr>
<tr>
<td>09.11.2009</td>
<td><strong>Project 1: Intro + Programming tips</strong></td>
</tr>
<tr>
<td>13.11.2009</td>
<td>Face Detection II: ANNs, SVM, Viola &amp; Jones</td>
</tr>
<tr>
<td>16.11.2009</td>
<td>Project 1: Questions</td>
</tr>
<tr>
<td>20.11.2009</td>
<td>Face Recognition I: Traditional Approaches, Eigenfaces, Fisherfaces, EBGM</td>
</tr>
<tr>
<td>23.12.2009</td>
<td>Face Recognition II</td>
</tr>
<tr>
<td>27.11.2009</td>
<td>Head Pose Estimation: Model-based, NN, Texture Mapping, Focus of Attention</td>
</tr>
<tr>
<td>30.11.2009</td>
<td>People Detection I</td>
</tr>
<tr>
<td>03.12.2009</td>
<td>People Detection II</td>
</tr>
<tr>
<td>07.12.2009</td>
<td><strong>Project 1: Student Presentations, Project 2: Intro</strong></td>
</tr>
<tr>
<td>11.12.2009</td>
<td>People Detection III (Part-Based Models)</td>
</tr>
<tr>
<td>14.12.2009</td>
<td>Scene Context and Geometry</td>
</tr>
<tr>
<td>18.12.2009</td>
<td>Stereo and Optical Flow</td>
</tr>
<tr>
<td>21.12.2009</td>
<td>TBA</td>
</tr>
</tbody>
</table>
This Lecture

- Overview of computer vision
  - Recognition Problems/Challenges
  - Learning Process
  - Evaluation procedures

- Topics
  - Localization vs. Classification vs. Identification
  - Challenges (Illumination, Intra-Class Variation etc.)
  - Data Sets
  - Performance Measures
  - Non-Maximum Suppression
Recognition Problems
Recognition Problems

- Different Types of Recognition Problems:
  - Object Identification
    - recognize your pencil, your dog, your car
  - Object Classification
    - recognize any pencil, any dog, any car
    - also called: generic object recognition, object categorization, …

- Recognition and
  - Segmentation: separate pixels belonging to the foreground (object) and the background
  - Localization: position of the object in the scene, pose estimate (size/scale, orientation, 3D position)

- Terms are not used consistently in the literature
Classification

- Learn the properties of a whole object class
- Example:

- Challenge:
  - Generalize from training examples to unseen instances
  - What is a class? -> Visual Categories
Identification

- Distinguish an object instance from other similar instances
- Example

- Challenge:
  - Find features which make the object unique (e.g. shape, color)
  - Generally requires a more detailed model than classification
Localization

- Find an object’s position and size within the image

- Challenge:
  - Multiple object instances
  - Huge search space for possible locations and sizes
Computer Vision today

- A vast amount of research interest in these areas
  - Universities, Industry
- Standard databases and challenges exist
  - Visual Object Challenge, Urban Grand Challenge
- A lot of progress in recent years

![Result Comparison on Test Set 2](image)
Challenges
Realistic Environments

- Computer vision systems for human computer interaction need to work in realistic environments
- Challenges we in realistic environments
  - Illumination
  - Intra-Class Variation
  - Perspective Transformations and Scale
  - Articulations/Deformations
  - View-Points
  - Background clutter
  - Occlusion
  - Insufficient processing power
Challenges: Illumination

- Depending on the lighting conditions average pixel values can be rather bright or dark
- Overexposure or underexposure can hide certain details
- Effective local contrast normalization is essential for good performance (see e.g. [Dalal and Triggs, CVPR’05])
- For humans it still appears similar, but it’s quite different in terms of pixel values
Approaches: Illumination

- Histogram equalization
  - make sure color values spread the full range of possible values

- Local normalization
  - Normalize the feature response in a local area (e.g. gradient strength)
  - Does not assume that all image areas are illuminated in the same manner
Background Clutter

- Background structures can distract detection
  - Background structures can have similar characteristics as the object of interest
  - Background structures may hide actual object boundaries
Challenges: Intra-Class Variation

- Objects often do not share a common color or texture
  - Difficult to derive recognition cues
- They can even be quite similar to other objects
Intra-Class Variation

- Model needs to be able to capture the variance
- But still retain information of how the object class is different from other classes

- Shape vs. Appearance
  - In some cases shape information is more promising to characterize an object (e.g. cups, people)
  - “internal regions are unreliable cues” [Dalal&Triggs]
  - In some cases appearance information (i.e. color pixel values or texture) is more appropriate
Challenges: View-Points

- We distinguish
  - In-Plane rotation
  - Out-Of-Plane rotation

- Objects may look completely different from various viewing angles
Approaches: View-Points

- **In-Plane rotation:**
  - Rotation-invariant object description (e.g. histograms)
  - Computation of a dominant orientation as a reference (e.g. local descriptors, SIFT)

- **Out-Of-Plane rotation:**
  - Separate detector for different view-points
Perspective Transformations

- Slight changes in camera position can alter pixel values considerably

- Common approaches:
  - Just ignore it and hope that your model is flexible enough to capture these differences
  - Affine invariant local interest points
    -> compute homography from matching points
Scale

- Generally, we have no a-priori information on the pixel size of the object
- We also typically do not know
  - the real-world dimensions of the object
  - the scene geometry
- Object details may be present/absent depending on the image resolution (if resolution is too small, objects are often hard to distinguish, even for humans)
Approaches: Scale

- Exhaustive search of all possible image locations and scales
  - Sliding window

- Scale estimation from local image regions
  - Scale-space theory
  - Scale-invariant interest points
Sliding Window Technique

- We obtain for each position/scale a recognition score
- Parameters: scale range, scale steps, x/y-steps
- Positions with low scores can be discarded

Red: score > 0.8
Sliding Window Technique

- Computationally expensive
- Often a very simple “pre-filter” is applied to already discard non-promising regions with minimal computation effort
  - E.g. discard homogeneous regions
  - Examples:
    - Cascade of classifiers [Viola&Jones2000]
Challenges: Articulations / Deformations

- We distinguish:
  - Rigid objects (e.g. cars, cups)
  - Non-Rigid objects (e.g. people, towels)
Articulations vs. Deformations

- **Articulated objects**
  - Models for different articulations (e.g. body parts and configurations)
  - Ideally, an object model should only allow configurations, which are common or even possible based
  - Models can be
    - hand-crafted (often used for tracking)
    - automatic (more common in recognition)

- **Deformable objects**
  - Widely unsolved problem
  - Often the object’s texture is used as recognition cue
Challenges: Occlusion

- In realistic environments the object may not be always completely visible
  - Not only missing data, but also distracting data
  - Need to find a way to ignore occluded portions of the object
Approaches: Occlusion

1. Part-based models
   - Model object parts instead of the whole object
   - Detect the individual object parts

2. Iteratively find image pixels which comply with the object model
   - E.g. Robust PCA

- Problems:
  - Overlapping objects: which image part belongs to which object hypothesis?
  - Background: How to distinguish hypothesis from partly visible objects from background structures
Challenges: Processing Power

- Many computer vision algorithms require a huge amount of computation
  - Hard to do in real-time
  - Computer Vision community started to use graphics hardware, multi-processors, computer clusters, FPGA

- This is one reason why computer vision has advanced considerably in recent years (and still will advance)
  - Many things have already been done
  - But there’s still much room for improvement

- Some people claim, that enough memory (and fast access to it) is the key to recognition
Challenges: scale, efficiency

- About half of the cerebral cortex in primates is devoted to processing visual information [Felleman and van Essen 1991]
- 3,000-30,000 human recognizable object categories
- 30+ degrees of freedom in the pose of articulated objects (humans)
- Billions of images indexed by Google Image Search (not to mention video)
- 18 billion+ prints produced from digital camera images in 2004
- 295.5 million camera phones sold in 2005
Learning Process
Computer Vision Algorithms

- Computer vision algorithms are typically data-driven
  - i.e. example data is used instead of completely hand-crafted models
  - Often machine learning techniques are used to process data
- In this lecture we consider mainly ‘supervised’ methods
- Typically, we distinguish 3 phases when we develop a computer vision algorithm
  - Training
  - Validation
  - Testing
Training Data

- Object recognition performance may strongly depend on the quality of the training data

- Generally, training data should:
  - Cover the full range of object variations
  - Contain realistic background structures
  - Be carefully pre-processed
  - The more, the better
Preprocessing

- Preprocessing reduces the degrees of freedom for a learning algorithm
  - We remove the variations, the algorithm may safely ignore

- Alignment
  - Resize the training objects to a common size
  - Align object centers and crop an image region around the object
  - Mostly accomplished by manual labeling

- Normalization
  - E.g. Histogram equalization
Increasing the number of training examples

- Methods to increase the number of training examples
  - Mirroring
  - Perturbations in x-/y-positions

- You can never have enough training data!
  - Joint effort: LabelMe Database
From Training Data to an Object Model

1. Feature Representation
   - E.g. Color-Histograms, Edge information

2. Typically learning approach
   - E.g. SVM, Neural Networks, Bayesian approaches

3. Possibly additional hand-crafted information
   - Domain knowledge (e.g. cars are generally weakly textured in the interior)
   - Object restrictions (e.g. a body model)
Object Models

- We distinguish
  - Generative Models
  - Discriminative Models

1. Generative Models
   - Learn a representative model (i.e. learn how the object looks)
   - Generally, only needs positive training examples
   - Example: Principal Component Analysis

2. Discriminative Models
   - Learn what distinguishes the object from the background or other classes (i.e. only learn the differences)
   - Needs positive and negative training examples
   - Example: Support Vector Machines
Model and Training

- Problems:
  - Generalization
    - We are able to recognize data, which was not in the training set
    - Typically better for generative approaches
  
  - Discriminative power
    - We do not get too many false detections
    - Typically better for discriminative approaches
Which model is best?

- Training sample
- Test sample
- Simple model
- Complex model
Occam’s razor

- „All other things being equal, the simplest solution is the best“

- It is often unknown, what an algorithm has learned exactly
  - i.e. make models as simple as possible, but as complex as necessary
  - The same applies to the training data (e.g. reduce unnecessary degrees of freedom, remove outliers)
Overfitting and Learning

- Overfitting
  - Occurs when the model has too many parameters and may perfectly fit the training data

![Graph showing Overfitting and Learning](image)

- Error on training data
- Error on test data
- Best generalization
Validation Data

- Overfitting can be avoided/detected by using a validation data set
  - If performance on validation set drops, we stop training the model

- Problem:
  - Amount of available data is limited

- Solution:
  - Cross-Validation (e.g. leave-one-out)
Leave-One-Out

- Divide training data into k-partitions
- Repeat for all possible partitions i
  - Choose partition i as a validation set
  - Use the remaining k-1 partitions to train the model
- Combine the parameters of the resulting k models for the final model

- Positive:
  - Does not waste data
- Negative:
  - High-Computational effort
Testing

- Run algorithm
- Measure performance (this is not completely trivial)
Performance Measures
Performance Measures

- Measuring the performance of object recognition algorithms is not trivial
- There are different measures depending on the application

1. For classification (i.e. yes/no decision, if object is present or not)
   - ROC (Receiver-Operating-Characteristic)

2. For localization (i.e. detecting the object’s position)
   - RPC (Recall-Precision-Curve)
   - DET (Detection Error Trade-Off)
Classifying a hypothesis

- When comparing recognition hypotheses with ground-truth annotations have to consider four cases:

<table>
<thead>
<tr>
<th>Positive examples (Pos)</th>
<th>Predicted positive</th>
<th>Predicted negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>True positive (TP)</td>
<td>False negative (FN)</td>
<td></td>
</tr>
<tr>
<td>Negative example (Neg)</td>
<td>False positive (FP)</td>
<td>True negatives (TN)</td>
</tr>
</tbody>
</table>

- Example:

<table>
<thead>
<tr>
<th>Prediction</th>
<th>Case</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>TP</td>
<td>TP</td>
</tr>
<tr>
<td>No</td>
<td>FN</td>
<td>FN</td>
</tr>
<tr>
<td>Yes</td>
<td>FP</td>
<td>FP</td>
</tr>
<tr>
<td>No</td>
<td>TN</td>
<td>TN</td>
</tr>
</tbody>
</table>
ROC

- Used for the task of classification
- Measures the trade-off between true positive rate and false positive rate:

\[
\text{true positive rate} = \frac{TP}{Pos} = \frac{TP}{TP+FN}
\]

\[
\text{false positive rate} = \frac{FP}{Neg} = \frac{FP}{FP+TN}
\]

- Example:
  - Algorithm X detects 80% of all cups (true positive rate), while making 25% error on images not containing cups
- Each prediction hypothesis has generally an associated probability value or score.
- The performance values can therefore plotted into a graph for each possible score as a threshold.

![ROC Graph]

- True positive rate
- False positive rate
- ROC heaven
- ROC hell

100% TPR, 100% FPR
85% TPR, 68% FPR
72% TPR, 30% FPR
60% TPR, 11% FPR
35% TPR, 3% FPR
15% TPR, 1% FPR
Recall-Precision

- For localization ROC is not appropriate, since the number of hypothesis varies
- We therefore define:
  - Recall: percentage of objects found
  - Precision: percentage of correct hypotheses

\[
\begin{align*}
\text{true positive rate} & = \frac{TP}{Pos} = \frac{TP}{TP+FN} \\
\text{false positive rate} & = \frac{FP}{Neg} = \frac{FP}{FP+TN} \\
\text{recall} & = \frac{TP}{Pos} \\
\text{precision} & = \frac{TP}{TP+FP} = \frac{TP}{\#\text{hypotheses}}
\end{align*}
\]
Plotting Recall-Precision

- RPC are typically plotted on a recall vs. 1-precision scale:

![Plot of Recall-Precision](image)

- 72% TPR, 70% precision
- 70% recall, 68% precision
- 60% recall, 89% precision
- 35% TPR, 97% precision
- 15% recall, 99% precision
Real-World Example

Result Comparison on Test Set 2

- 4D-ISM
- ISM (Shape Context + HesLap)
- Dalal&Triggs (INRIA)
- ISM (Patches + DoG)
Detection Error Trade-Off

- DET measures the number of false detections per tested image window with respect to the miss-rate (1 – recall)
- Used for a sliding window based detector

Disadvantages:
- Chart more difficult to read (e.g. log-scale)
- Depends on the number of windows tested (i.e. image size, sliding window parameters)
- Does not measure the performance of the complete detection system including non-maximum suppression
Further measures

- In order to express the performance in a single figure, the following measures are common:
  - **Equal Error Rate (EER)**
    - The point where the errors for true positives and false positives are equal (i.e. points on the diagonal from (0,1) to (1,0))
    - Not well-defined for RPC
  - **Area Under Curve (for ROC)**
    - The area under the curve ;-)
  - **Mean Average Precision (for RPC)**
    - Average precision values
    - Sometimes measured only at pre-defined recall values
  - **False Positives per Image (FPPI)**
Localization and Ground-Truth

- For localization, the test data is mostly annotated with ground-truth bounding boxes.
- It is often not obvious when to count a hypotheses as true or false detection.
  - Misaligned hypotheses
  - Double detections
Comparing hypotheses to Ground-Truth

- Comparison measures
  1. Relative Distance
  2. Cover and Overlap

1. Relative distance

$$d_r = \sqrt{\left(\frac{2 \cdot \Delta x}{w}\right)^2 + \left(\frac{2 \cdot \Delta y}{h}\right)^2}$$
Comparing hypotheses to Ground-Truth

2. Cover and Overlap

- There is no standard for which values to choose
- Sometimes used as threshold:
  - Cover > 50%, Overlap > 50%, Relative Distance < 0.5
- Double detections are counted as false positive
Example: Localization

- Algorithm output:
  - Image 1: (13, 34, 200, 200), Prob: 0.8
  - Image 1: (341, 211, 174, 174), Prob: 0.4
  - ...
  - Image n: (78, 12, 55, 55), Prob: 0.7
- Sort hypotheses according to the score/probability
- Classify hypotheses into true and false positives
- Increase threshold gradually
- Compute for each new hypothesis the recall and precision values
Non-Maximum Suppression

- A good detector will generally not only fire on the exact position

- Need to reduce the number of detections, since every additional detection (even on the object) will count as false positive detection
Naïve Approaches

- Based on cover/overlap
  - Accept the strongest hypothesis
  - Remove all other hypotheses, which strongly overlap

- Work reasonably well in practice
- Cannot deal well with partially overlapping objects
Advanced approaches

- Finding modes in a non-parametric distribution
  - Kernel Density Estimation
  - Mean-Shift Mode Estimation (MSME) (e.g. [Dalal’05])

- Pixel-based reasoning (e.g. [Leibe et al. 2004])
  - Infer an object segmentation
  - Use segmentation to determine, which pixels belong to which object

- Clustering techniques