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
Context and 3D Structure

WS 2009/10

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>Computer Vision: Tasks, Challenges, Learning, Performance measures</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>People Detection IV</td>
</tr>
<tr>
<td>18.12.2009</td>
<td><strong>Context and 3D Structure</strong></td>
</tr>
<tr>
<td>21.12.2009</td>
<td>Facial Feature Detection</td>
</tr>
</tbody>
</table>
This Lecture

- Context
  - Motivation
  - Scene Geometry
- Geometry from a single image
- Multi-View Geometry: Stereo
  - Epipolar Constraints
  - Disparity computation
The Role of the Scene
Context

Antonio Torralba, IJCV 2003
Limitations (continued)

- If considering windows in isolation, context is lost
Context
Context
Dynamics
Dynamics

[Images of dynamic movements]
3D Structure
Geometric vision

- Goal: Recovery of 3D structure
  - What cues in the image allow us to do this?
Visual Cues

- Shading

Merle Norman Cosmetics, Los Angeles
Visual Cues

- Shading
- Texture

*The Visual Cliff*, by William Vandivert, 1960
Visual Cues

- Shading
- Texture
- Focus

From *The Art of Photography*, Canon
Visual Cues

- Shading
- Texture
- Focus
- Perspective
Visual Cues

- Shading
- Texture
- Focus
- Perspective
- Motion

Figures from L. Zha

http://www.brainconnection.com/teasers/?main=illusion/motion-shape
Goal: Recovery of 3D Structure

- Generally: focus on perspective and motion
- recovery of structure from one image is inherently ambiguous, *multi-view geometry* allows disambiguation
To Illustrate This Point…

- Structure and depth are inherently ambiguous from single views.
Stereo Vision

http://www.well.com/~jimg/stereo/stereo_list.html

Slide credit: Kristen Grauman
Context and Geometry
From a Single Image
Motivation

- How to recognize the car-like blob?
- General object recognition cannot be solved locally.
- The interpretation of the entire image is required.
- Real relationships are 3D.
  - It’s sitting on the road.
  - It’s the “right” size, relative to other objects in the scene (cars, buildings, and pedestrians, etc.)
Today: Local and Independent
Local Object Detection

True Detection

False Detections

Missed

True Detections

Local Detector: [Dalal-Triggs 2005]
Real Relationships are 3D
What information is needed to compute real-world object size from image data?

- **Viewpoint**
  - Assume the objects all lie on the ground plane.
  - Then, we only need two variables:
    - Horizon position
    - Camera height
Camera Geometry

- **Camera Model with tilt angle**
  \[
  \begin{bmatrix}
  u \\
  v \\
  1
  \end{bmatrix} = \frac{1}{z} \begin{bmatrix}
  f & 0 & u_c \\
  0 & f & v_c \\
  0 & 0 & 1
  \end{bmatrix} \begin{bmatrix}
  1 & 0 & 0 & 0 \\
  0 & \cos \theta_x & -\sin \theta_x & y_c \\
  0 & \sin \theta_x & \cos \theta_x & 0
  \end{bmatrix} \begin{bmatrix}
  x \\
  y \\
  z \\
  1
  \end{bmatrix}
  \]

- **No tilt angle**
  \[
  \begin{bmatrix}
  u \\
  v \\
  1
  \end{bmatrix} = \frac{1}{z} \begin{bmatrix}
  f & 0 & u_c \\
  0 & f & v_c \\
  0 & 0 & 1
  \end{bmatrix} \begin{bmatrix}
  1 & 0 & 0 & 0 \\
  0 & 1 & 0 & y_c \\
  0 & 0 & 1 & 0
  \end{bmatrix} \begin{bmatrix}
  x \\
  y \\
  z \\
  1
  \end{bmatrix}
  \]
Solving for no tilt angle

- Simple matrix multiplication

\[
\begin{bmatrix}
  u \\
  v \\
  1
\end{bmatrix} = \begin{bmatrix}
  \frac{fx}{z} + uc \\
  \frac{f}{z}(y + yc) + vc \\
  1
\end{bmatrix}
\]

- Solve for \( y \)

\[
y = \frac{z}{f}(v - vc) - yc
\]

- Given \( v_b \) (bottom position of object in image) and \( v_t \) (top position) with \( y=0 \) for \( v_b \)

\[
0 = \frac{z}{f}(v_b - vc) - yc \\
z = \frac{fy_c}{v_b - v_c}
\]
**Height without tilt angle**

- y-position of top in real world
  \[ y_T = \frac{y_c}{v_b - v_c}(v_t - v_c) - y_c = \frac{y_c}{v_b - v_c}v_t - y_c \]

- Real world height
  \[ h = y_T - y_B = y_c \frac{v_t - v_c - (v_b - v_c)}{v_b - v_c} = y_c \frac{v_t - v_b}{v_b - v_c} \]
  \[ h = y_c \frac{v_t - v_b}{v_b - v_c} \]

- With a small tilt and a horizon position at \( v_0 \)
  \[ h \approx y_c \frac{v_t - v_b}{v_b - v_0} \]
Horizon and Camera position

• How do we determine these parameters from a single image?

• Start with simple priori models.
  – Horizon position: 0.50
  – Camera height: 1.67
Object Size ↔ Camera Viewpoint

Input Image

Loose Viewpoint Prior
Object Size ↔ Camera Viewpoint

Object Position/Sizes ⇒ Viewpoint

Edgar Seemann, 14.12.09
Object Size ↔ Camera Viewpoint

Object Position/Sizes

Viewpoint
Object Size ↔ Camera Viewpoint

Object Position/Sizes ↔ Viewpoint
Surface Estimation
Surface Types

- In many outdoor images, we can distinguish
  - Ground-Plane pixels
  - Vertical structures (90 degree angle to the ground plane)
  - Sky

- Goal
  - Assign geometric labels to all image pixels
  - i.e. segment image into distinct surface types
The process

Image  Super Pixels  geom. labels
Superpixels [Felzenszwalb et al]

- Larger image regions, which are assumed to correspond to a single label
- Typically
  - Uniform regions
  - Regions with the same texture
  - Respect object boundaries
Algorithm

- Label a set of training images
- Compute superpixels
- Compute various features on the superpixels
- Use AdaBoost to learn assignment
<table>
<thead>
<tr>
<th>Feature Descriptions</th>
<th>Num</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Color</strong></td>
<td></td>
</tr>
<tr>
<td>C1. RGB values: mean</td>
<td>16</td>
</tr>
<tr>
<td>C2. HSV values: C1 in HSV space</td>
<td>3</td>
</tr>
<tr>
<td>C3. Hue: histogram (5 bins) and entropy</td>
<td>3</td>
</tr>
<tr>
<td>C4. Saturation: histogram (3 bins) and entropy</td>
<td>6</td>
</tr>
<tr>
<td><strong>Texture</strong></td>
<td></td>
</tr>
<tr>
<td>T1. DOOG filters: mean abs response of 12 filters</td>
<td>12</td>
</tr>
<tr>
<td>T2. DOOG stats: mean of variables in T1</td>
<td>1</td>
</tr>
<tr>
<td>T3. DOOG stats: argmax of variables in T1</td>
<td>1</td>
</tr>
<tr>
<td>T4. DOOG stats: (max - median) of variables in T1</td>
<td>1</td>
</tr>
<tr>
<td><strong>Location and Shape</strong></td>
<td></td>
</tr>
<tr>
<td>L1. Location: normalized x and y, mean</td>
<td>2</td>
</tr>
<tr>
<td>L2. Location: norm. x and y, 10\text{th} and 90\text{th} pctl</td>
<td>4</td>
</tr>
<tr>
<td>L3. Location: norm. y wrt horizon, 10\text{th}, 90\text{th} pctl</td>
<td>2</td>
</tr>
<tr>
<td>L4. Shape: number of superpixels in region</td>
<td>1</td>
</tr>
<tr>
<td>L5. Shape: number of sides of convex hull</td>
<td>1</td>
</tr>
<tr>
<td>L6. Shape: num pixels/area(convex hull)</td>
<td>1</td>
</tr>
<tr>
<td>L7. Shape: whether the region is contiguous in ${0, 1}$</td>
<td>1</td>
</tr>
<tr>
<td><strong>3D Geometry</strong></td>
<td></td>
</tr>
<tr>
<td>G1. Long Lines: total number in region</td>
<td>35</td>
</tr>
<tr>
<td>G2. Long Lines: % of nearly parallel pairs of lines</td>
<td>1</td>
</tr>
<tr>
<td>G3. Line Intsect: hist. over 12 orientations, entropy</td>
<td>3</td>
</tr>
<tr>
<td>G4. Line Intsect: % right of center</td>
<td>1</td>
</tr>
<tr>
<td>G5. Line Intsect: % above center</td>
<td>1</td>
</tr>
<tr>
<td>G6. Line Intsect: % far from center at 8 orientations</td>
<td>8</td>
</tr>
<tr>
<td>G7. Line Intsect: % very far from center at 8 orient.</td>
<td>8</td>
</tr>
<tr>
<td>G8. Texture gradient: x and y “edginess” (T2) center</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 1: Features computed on superpixels (C1-C2,T1-T4,L1) and on regions (all). The “Num” column gives the number of features in each set. The boosted decision tree classifier selects a discriminative subset of these features.
Surface Estimation [Hoiem, Efros, Hebert ICCV 2005]

Image  |  Ground  |  Vertical  |  Sky
--- | --- | --- | ---

Planar Left | Planar Center | Planar Right | Non-planar Porous | Non-planar Solid

Object Surface?  
Support?
Object Support
What does surface and viewpoint say about objects?

- **Image**
- **P(surfaces)**
- **P(viewpoint)**

- **P(object) = uniform**
- **P(object | surfaces)**
- **P(object | viewpoint)**
What does surface and viewpoint say about objects?

Image

P(surfaces)

P(viewpoint)

P(object) = uniform

P(object | surfaces, viewpoint)
Scene Parts Are All Interconnected

Objects

Viewpoint

3D Surfaces
Qualitative Results

Car: TP / FP  Ped: TP / FP

Initial: 2 TP / 3 FP
Final: 7 TP / 4 FP

Local Detector from [Murphy-Torralba-Freeman 2003]
Qualitative Results

Car: TP / FP  Ped: TP / FP

Initial: 1 TP / 14 FP  Final: 3 TP / 5 FP

Local Detector from [Murphy-Torralba-Freeman 2003]
Multi-View Geometry: Stereo Images
Depth with Stereo: Basic Idea

- Basic Principle: Triangulation
  - Gives reconstruction as intersection of two rays
  - Requires: Camera pose (calibration), Point correspondence
Camera Calibration

- **Extrinsic params**: rotation matrix and translation vector
- **Intrinsic params**: focal length, pixel sizes (mm), image center point, radial distortion parameters

We’ll assume for now that these parameters are given and fixed.
Geometry for a Simple Stereo System

- First, assuming parallel optical axes, known camera parameters (i.e., calibrated cameras):
Geometry for a Simple Stereo System

- Assume parallel optical axes, known camera parameters (i.e., calibrated cameras). We can triangulate via:

Similar triangles \((p_l, p, p_r)\) and \((O_l, p, O_r)\):

\[
\frac{T + x_l - x_r}{Z - f} = \frac{T}{Z}
\]

\[
Z = f \frac{T}{x_r - x_l}
\]
Depth From Disparity

Image $I(x, y)$    Disparity map $D(x, y)$    Image $I'(x', y')$
General Case With Calibrated Cameras

- The two cameras need not have parallel optical axes.

VS.

Slide credit: Kristen Grauman, Steve Seitz
Stereo Correspondence Constraints

- Given \( p \) in the left image, where can the corresponding point \( p' \) in the right image be?
Stereo Correspondence Constraints

- Given p in the left image, where can the corresponding point p’ in the right image be?
Stereo Correspondence Constraints
Stereo Correspondence Constraints

- Geometry of two views allows us to constrain where the corresponding pixel for some image point in the first view must occur in the second view.

- Epipolar constraint: Why is this useful?
  - Reduces correspondence problem to 1D search along conjugate epipolar lines.
Epipolar Geometry

- Epipolar Plane
- Epipoles
- Baseline
- Epipolar Lines

Slide adapted from Marc Pollefeys
Epipolar Geometry: Terms

- **Baseline**: line joining the camera centers
- **Epipole**: point of intersection of baseline with the image plane
- **Epipolar plane**: plane containing baseline and world point
- **Epipolar line**: intersection of epipolar plane with the image plane
- All epipolar lines intersect at the epipole.
- An epipolar plane intersects the left and right image planes in epipolar lines.
Epipolar Constraint

- Potential matches for $p$ have to lie on the corresponding epipolar line $l'$.
- Potential matches for $p'$ have to lie on the corresponding epipolar line $l$.

http://www.ai.sri.com/~luong/research/Meta3DViewer/EpipolarGeo.html
Example
Example: Converging Cameras

As position of 3d point varies, epipolar lines “rotate” about the baseline

Slide credit: Kristen Grauman

Figure from Hartley & Zisserman
For a given stereo rig, how do we express the epipolar constraints algebraically?
Stereo Geometry With Calibrated Cameras

- If the rig is calibrated, we know:
  - How to rotate and translate camera reference frame 1 to get to camera reference frame 2.
    - Rotation: 3 x 3 matrix; translation: 3 vector.
Rotation Matrix

Express 3d rotation as series of rotations around coordinate axes by angles $\alpha, \beta, \gamma$

Overall rotation is product of these elementary rotations:

$$ R = R_x R_y R_z $$
3D Rigid Transformation

\[
\begin{bmatrix}
X' \\
Y' \\
Z'
\end{bmatrix}
= 
\begin{bmatrix}
r_{11} & r_{12} & r_{13} \\
r_{21} & r_{22} & r_{23} \\
r_{31} & r_{32} & r_{33}
\end{bmatrix}
\begin{bmatrix}
X \\
Y \\
Z
\end{bmatrix}
+ 
\begin{bmatrix}
T_x \\
T_y \\
T_z
\end{bmatrix}
\]

\[X' = RX + T\]
Stereo Geometry With Calibrated Cameras

- Camera-centered coordinate systems are related by known rotation $R$ and translation $T$:

$$X' = RX + T$$
Cross Product

\[ \vec{a} \times \vec{b} = \vec{c} \]
\[ \vec{a} \cdot \vec{c} = 0 \]
\[ \vec{b} \cdot \vec{c} = 0 \]

- Vector cross product takes two vectors and returns a third vector that’s perpendicular to both inputs.
- So here, c is perpendicular to both a and b, which means the dot product = 0.
From Geometry to Algebra

\[ X' = RX + T \]

\[ T \times X' = T \times RX + T \times T \]

Normal to the plane

\[ = T \times RX \]

\[ X' \cdot (T \times X') = X' \cdot (T \times RX) = 0 \]

Slide credit: Kristen Grauman
Matrix Form of Cross Product

\[ \vec{a} \times \vec{b} = \vec{c} \]

\[ \vec{a} \cdot \vec{c} = 0 \]
\[ \vec{b} \cdot \vec{c} = 0 \]

\[
\begin{bmatrix}
0 & -a_z & a_y \\
a_z & 0 & -a_x \\
-a_y & a_x & 0
\end{bmatrix}
\]

\[ \tilde{a} \times \tilde{b} = [a_x] \tilde{b} \]
From Geometry to Algebra

\[
\begin{align*}
X' &= RX + T \\
T \times X' &= T \times RX + T \times T \\
\text{Normal to the plane} &= T \times RX
\end{align*}
\]

\[
X' \cdot (T \times X') = X' \cdot (T \times RX) = 0
\]

Slide credit: Kristen Grauman
Essential Matrix

\[
X' \cdot (T \times RX) = 0 \\
X' \cdot (T_x RX) = 0
\]

Let \( E = T_x R \)

\[
X'^T E X = 0
\]

- This holds for the rays \( p \) and \( p' \) that are parallel to the camera-centered position vectors \( X \) and \( X' \), so we have:
- \( E \) is called the essential matrix, which relates corresponding image points [Longuet-Higgins 1981]

\[
p'^T E p = 0
\]
Essential Matrix and Epipolar Lines

Epipolar constraint: if we observe point \( p \) in one image, then its position \( p' \) in second image must satisfy this equation.

\[
p'^T E p = 0
\]

\( E^T p \) is the coordinate vector representing the epipolar line for point \( p \)

\( E p' \) is the coordinate vector representing the epipolar line for point \( p' \)
Essential Matrix: Properties

- Relates image of corresponding points in both cameras, given rotation and translation.
- Assuming intrinsic parameters are known

\[ E = T_x R \]
Essential Matrix Example: Parallel Cameras

For the parallel cameras image of any point must lie on same horizontal line in each image plane.

Edgar Seemann, 14.12.09

Slide credit: Kristen Grauman
Essential Matrix Example: Parallel Cameras

For the parallel cameras, image of any point must lie on same horizontal line in each image plane.

Edgar Seemann, 14.12.09

Slide credit: Kristen Grauman
More General Case

Image I(x,y)  Disparity map D(x,y)  Image I´(x´,y´)

What about when cameras’ optical axes are not parallel?
Stereo Image Rectification

- In practice, it is convenient if image scanlines are the epipolar lines.

- Algorithm
  - Reproject image planes onto a common plane parallel to the line between optical centers
  - Pixel motion is horizontal after this transformation
  - Two homographies (3x3 transforms), one for each input image reprojection

C. Loop & Z. Zhang, Computing Rectifying Homographies for Stereo Vision. CVPR’99
Stereo Image Rectification: Example

Source: Alyosha Efros
Stereo Reconstruction

- Main Steps
  - Calibrate cameras
  - Rectify images
  - Compute disparity
  - Estimate depth
Correspondence Problem

Multiple match hypotheses satisfy epipolar constraint, but which is correct?

- Hypothesis 1
- Hypothesis 2
- Hypothesis 3

Figure from Gee & Cipolla 1999

Slide credit: Kristen Grauman
Correspondence Problem

- To find matches in the image pair, we will assume
  - Most scene points visible from both views
  - Image regions for the matches are similar in appearance
Additional Correspondence Constraints

- Similarity
- Uniqueness
- Ordering
- Disparity gradient
Dense Correspondence Search

- For each pixel in the first image
  - Find corresponding epipolar line in the right image
  - Examine all pixels on the epipolar line and pick the best match (e.g. SSD, correlation)
  - Triangulate the matches to get depth information

- This is easiest when epipolar lines are scanlines
  ⇒ Rectify images first

adapted from Svetlana Lazebnik, Li Zhang
Example: Window Search

- Data from University of Tsukuba

Scene

Ground truth
Example: Window Search

- Data from University of Tsukuba

Window-based matching (best window size) | Ground truth
Effect of Window Size

Want window large enough to have sufficient intensity variation, yet small enough to contain only pixels with about the same disparity.
Sparse Correspondence Search

- Restrict search to sparse set of detected features
- Rather than pixel values (or lists of pixel values) use feature descriptor and an associated feature distance
- Still narrow search further by epipolar geometry

What would make good features?
Dense vs. Sparse

- Sparse
  - Efficiency
  - Can have more reliable feature matches, less sensitive to illumination than raw pixels
  - But…
    - Have to know enough to pick good features
    - Sparse information

- Dense
  - Simple process
  - More depth estimates, can be useful for surface reconstruction
  - But…
    - Breaks down in textureless regions anyway
    - Raw pixel distances can be brittle
    - Not good with very different viewpoints
Difficulties in Similarity Constraint

Untextured surfaces

Occlusions
Additional Correspondence Constraints

- Similarity
- **Uniqueness**
- Ordering
- Disparity gradient
Uniqueness

- For opaque objects, up to one match in right image for every point in left image
Ordering

- Points on *same surface* (opaque object) will be in same order in both views

\[ Q_c \] Left image
\[ Q'_c \] Right image

- Satisfies ordering constraint

\[ Q_c \] Left image
\[ Q'_c \] Right image

- Violates ordering constraint

Figure from Gee & Cipolla 1999
Disparity Gradient

- Assume piecewise continuous surface, so want disparity estimates to be locally smooth

Given matches ● and ○, point ○ in the left image must match point 1 in the right image. Point 2 would exceed the disparity gradient limit.
Additional Correspondence Constraints

- Similarity
- Uniqueness
- Ordering
- Disparity gradient

- Epipolar lines constrain the search to a line, and these appearance and ordering constraints further reduce the possible matches.
Possible Sources of Error?

- Low-contrast / textureless image regions
- Occlusions
- Camera calibration errors
- Violations of *brightness constancy* (e.g., specular reflections)
- Large motions