Computer Vision Systems Programming VO

Specific Object Recognition

Christopher Pramerdorfer

Computer Vision Lab, Vienna University of Technology

Topics

Introduction to object recognition

Specific object recognition

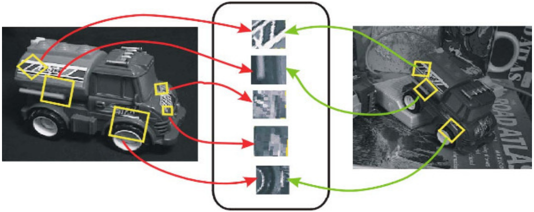


Image from Grauman and Leibe 2011

Object Recognition

Fundamental problem in Computer Vision

Many applications

- Panorama stitching, 3D reconstruction
- HCI and surveillance (face recognition)
- Image understanding (recall Fei-Fei Li's TED talk)

Object Recognition
Taxonomy – Instance vs. Category

Instance recognition (specific object recognition)

- Recognize a specific, uniquely looking object
- Face of a certain person, the Eiffel tower

Object category recognition

- Recognize objects of a certain category
- Human faces, buildings

Object Recognition
Taxonomy – Instance vs. Category

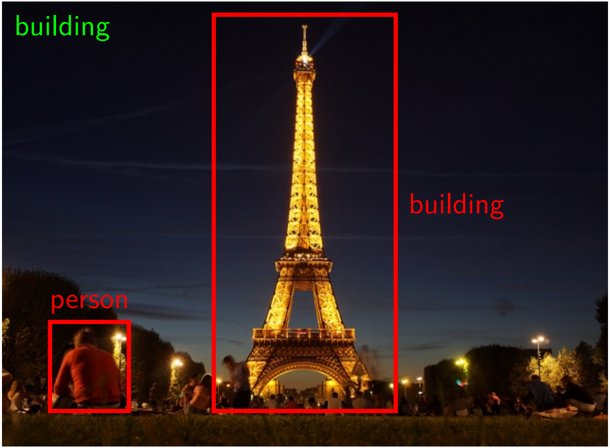
Object classification

- Recognize main object in image
- Location and other objects not relevant

Object detection

- Recognize multiple objects, possibly of different category

Object Recognition
Taxonomy – Classification vs. Detection



Object Recognition
Challenges

Instances of same category can look very differently

- Illumination, pose, viewpoint, occlusions, background

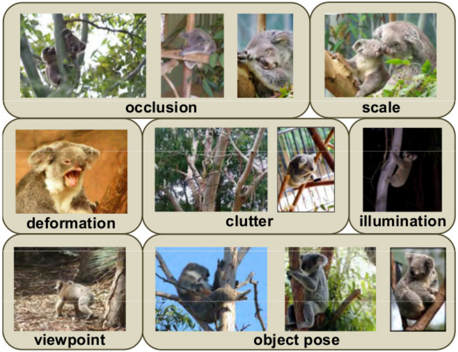


Image from Grauman and Leibe 2011

Planar Rigid Object Detection

We want to detect specific rigid planar objects

- Like markers, books
- Comparatively easy problem

Challenges

- Unknown object pose and scale
- Varying illumination
- Partial occlusions

Planar Rigid Object Detection

Application: Marker-Based AR

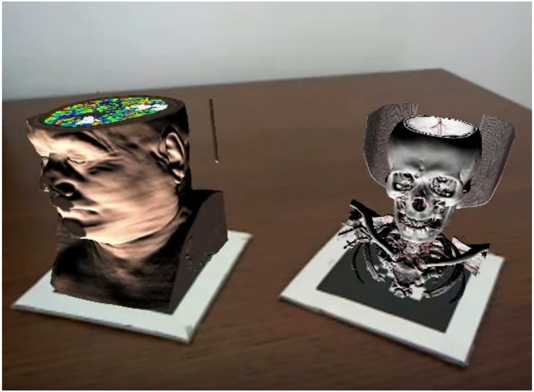


Image from youtube.com

Planar Rigid Object Detection

Application: Panorama Stitching

Assuming that object is far away (on *plane at infinity*)

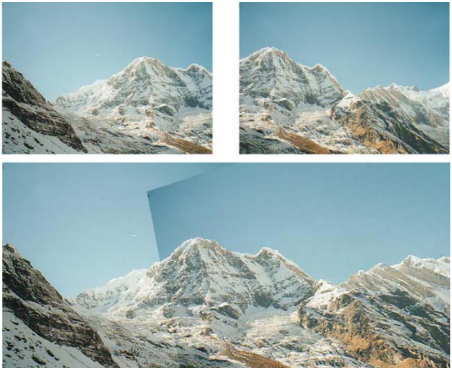


Image adapted from Brown and Lowe 2007

Our problem formulation is

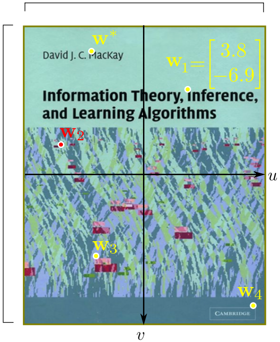
- Given a pixel location in a query image
- Predict location on object surface

So we know how to select $\mathbf{x}$ and $\mathbf{w}$

- $\mathbf{x} = (x, y)$: pixel location in query image
- $\mathbf{w} = (u, v)$: corresponding location on object surface

As $\mathbf{w} \in \mathbb{R}^2$ this is a regression problem

Planar Rigid Object Detection
Selecting $\mathbf{x}$ and $\mathbf{w}$



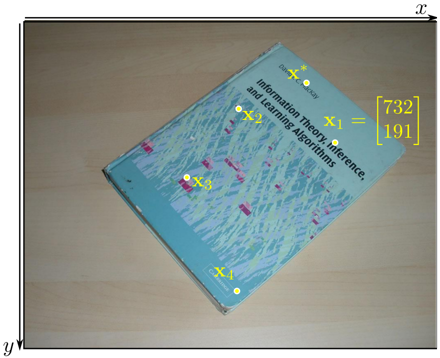


Image adapted from Prince 2012

Planar Rigid Object Detection
Model Selection

Images of planar objects are always related by a homography $\mathbf{\Phi}$

- 3×3 matrix mapping between corresponding points

In homogeneous coordinates this means that

$$\lambda \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \mathbf{\Phi} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

The model of choice is thus (disregarding noise)

$$\mathbf{w} = \Gamma(\mathbf{x}) = \begin{pmatrix} u \\ v \end{pmatrix} \quad , \quad \lambda \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \mathbf{\Phi} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

We again learn parameters $\boldsymbol{\theta}$ from samples $\{(\mathbf{x}_i, \mathbf{w}_i)\}_{i=1}^n$

- $\boldsymbol{\theta}$ contains 9 parameters comprising $\boldsymbol{\Phi}$

Usually no exact solution because of noisy $\mathbf{x}_i$

- Formulate as a least squares problem instead

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \left[\sum_{i=1}^{n} (\mathbf{w}_i - \Gamma(\mathbf{x}_i))^\top (\mathbf{w}_i - \Gamma(\mathbf{x}_i)) \right]$$

This least squares approach is optimal

- If noise is distributed normally with spherical covariance

This is a nonlinear optimization problem

- Solvable using any general nonlinear least squares solver
- OpenCV has an own function `findHomography`

$\mathbf{\Phi}$ is a 2D transformation

We may want to know the position and orientation of the object

- This is called pose estimation
- Required e.g. for marker-based AR like above

This information can be extracted from $\mathbf{\Phi}$

- If we know the intrinsic camera parameters
- See Prince 2012 for details

Planar Rigid Object Detection
Obtaining Point Correspondences

How can we compute $\{(\mathbf{x}_i, \mathbf{w}_i)\}_{i=1}^{n}$ automatically?

- We first select $\mathbf{w}_i$, then search corresponding $\mathbf{x}_i$

$\mathbf{w}_i$ can be selected

- Manually (e.g. specific corners on markers)
- Automatically (e.g. SIFT)

We opt for the second approach and use SIFT (or similar)

- Features invariant to rotation, scale, illumination
- Robust to affine transformations

Approach

- Compute keypoints and descriptors in both images
- Match descriptors (e.g. nearest neighbor association)
- Use keypoint locations of matches as $\{(\mathbf{x}_i, \mathbf{w}_i)\}_{i=1}^{n}$

There will likely be incorrect matches

- Would greatly impact the least squares solution
- Hence we use a robust alternative like RANSAC

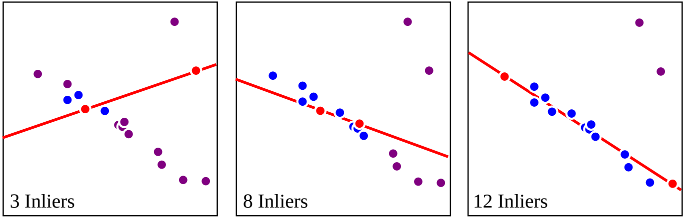


Image adapted from Prince 2012

```cpp
// read images (SIFT expects grayscale images)
cv::Mat object = cv::imread("object.jpg", cv::IMREAD_GRAYSCALE);
cv::Mat search = cv::imread("search.jpg", cv::IMREAD_GRAYSCALE);

cv::SIFT sift; // using default arguments here

// compute keypoints
std::vector<cv::KeyPoint> kobject, ksearch;
sift.detect(object, kobject); sift.detect(search, ksearch);

// compute descriptors
cv::Mat dobject, dsearch;
sift.compute(object, kobject, dobject); sift.compute(search, ksearch, dsearch);
```

Planar Rigid Object Detection
Obtaining Point Correspondences Using OpenCV

```
// find two nearest neighbors x,x' for each w
cv::FlannBasedMatcher matcher; // fast nearest neighbor search
std::vector<std::vector<cv::DMatch> > kMatches;
matcher.knnMatch(dobject, dsearch, kMatches, 2);

// keep match (x,w) if x is clearly more similar than x'
// this is a popular matching strategy
std::vector<cv::DMatch> matches;
for(const std::vector<cv::DMatch>& match : kMatches)
    if(match[0].distance < match[1].distance * 0.8) // x, x'
        matches.push_back(match[0]); // (x,w)
```

```cpp
// collect feature locations of correspondences from before
std::vector<cv::Point2f> pobject, psearch;
for(const cv::DMatch& match : matches) {
    pobject.push_back(kobject.at(match.queryIdx).pt);
    psearch.push_back(ksearch.at(match.trainIdx).pt);
}

// estimate homography using RANSAC for robustness
cv::Mat inliers; // contains indices of valid correspondences
cv::Mat homography = cv::findHomography(pobject, psearch, CV_RANSAC, 2, inliers);
```

Nonplanar Rigid Object Detection

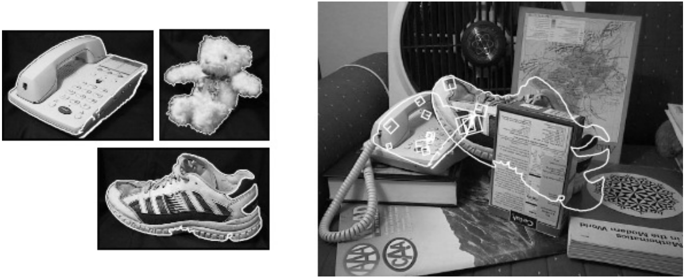


Image adapted from Lowe 2004

Nonplanar Rigid Object Detection

We use the same problem formulation as before

- Given a pixel location in an image $\mathbf{x}$
- Predict location on (nonplanar) object surface $\mathbf{w}$

As object is no longer planar, we have $\mathbf{w} = (u, v, w)$

How are $\mathbf{x}$ and $\mathbf{w}$ related in this case?

- Let's recap the pinhole camera model

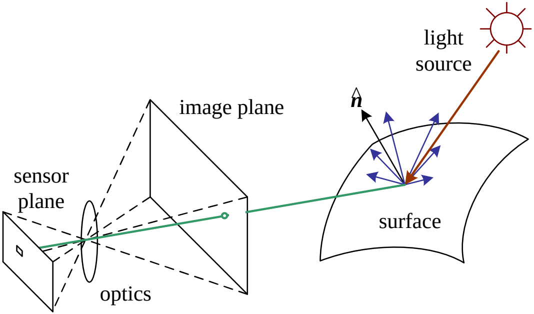


Image from Szeliski 2010

Nonplanar Rigid Object Detection
Pinhole Camera Model

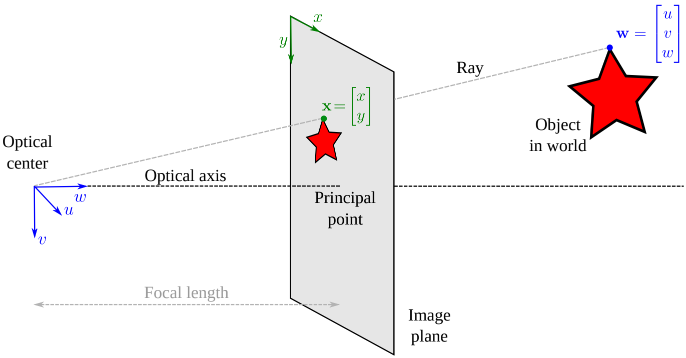


Image adapted from Prince 2012

We see that in homogeneous coordinates

$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} u \\ v \\ w \\ 1 \end{pmatrix}$$

f is the focal length in pixels

p_x, p_y are the principal point coordinates

World and camera coordinate systems generally differ

- Transform $\mathbf{w}$ to camera coordinates before projection

$$\mathbf{w}' = \begin{pmatrix} u' \\ v' \\ w' \end{pmatrix} = \begin{pmatrix} \omega_{11} & \omega_{12} & \omega_{13} \\ \omega_{21} & \omega_{22} & \omega_{23} \\ \omega_{31} & \omega_{32} & \omega_{33} \end{pmatrix} \begin{pmatrix} u \\ v \\ w \end{pmatrix} + \begin{pmatrix} \tau_u \\ \tau_v \\ \tau_w \end{pmatrix}$$

τ encode translation, ω encode rotation

We combine this for the full pinhole camera model

- Standard camera model in Computer Vision

$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \omega_{11} & \omega_{12} & \omega_{13} & \tau_u \\ \omega_{21} & \omega_{22} & \omega_{23} & \tau_v \\ \omega_{31} & \omega_{32} & \omega_{33} & \tau_w \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u \\ v \\ w \\ 1 \end{pmatrix}$$

Bibliography

Brown, Matthew and David G Lowe (2007). *Automatic panoramic image stitching using invariant features*.

Grauman, Kristen and Bastian Leibe (2011). *Visual object recognition*. Morgan & Claypool.

Lowe, David G (2004). *Distinctive image features from scale-invariant keypoints*.

Prince, S.J.D. (2012). *Computer Vision: Models Learning and Inference*. Cambridge University Press.

Szeliski, Richard (2010). *Computer vision: algorithms and applications*. Springer.