An introduction to 3D image reconstruction and understanding

concepts and ideas

Samuele Carli       Martin Hellmich

5 febbraio 2013
Introduction to the lecture series
Imagine...
Imagine...
Imagine... the Challenge
Stereo Vision
Techniques
Contents

Lecture 1: Human vision and image pre-processing

Lecture 2: Feature detection and 3D reconstruction

Lecture 3: Object recognition and scene understanding
Lecture 1
Introduction to the human visual system and image pre-processing

concepts and ideas

Samuele Carli       Martin Hellmich

5 febbraio 2013
Contents

A bit of anatomy

Few words on the basics

Image processing
Contents

A bit of anatomy
  Vision formation
  Cameras

Few words on the basics

Image processing
The eye
Eyes correlation
Horopter
Minimal context
Stereo matching in HVS

- Mostly guided by 'disparity detecting neurons'
- Efficient correlation of images (edges and high gradient spots)
- Less efficient correlation of textures
  - one of reasons why looking at random dot stereograms can be difficult
- We believe matching depends on correlation of retina image locations with second derivative of luminance (greatest change in signal instead of greatest signal)
- Indications of a 'fall-back' correlation mechanism when luminance is not enough
- Color has effect on matching (increased performance)
- Experience (and evolution) plays a big role (most interesting comes first)
Biofeedback

- Autonomous movement of eyes limited to high precision refinements
- Big movements are conscious and directed by brain as needed
- Brain can 'feel' position and focus of eyes: approximate distance/size of pointed object!
- Change of focus and parallax happens really often, helps to understand positions and occlusions
- Often the whole head gets moved to get an enhanced 3D impression
  - Both displacement and rotation helpful (baseline useful hint for distance)
The role of the brain

Lots of processing necessary in normal life:

- Differentiate objects of interest from background
- Locate objects in space
- Eventually predict movements/hazards!
- Recognize objects and associate them with meaning
- Find relationships, physical boundaries and connections (leaf/plants, tiger...)
- Act! (And act fast if the tiger is looking at you!)
Illusions
Notably, computers have eyes…

Hello, Dave!
...which are usually Cameras
Camera vs Eye

- CCD is not spherical
- but it has no blind spot
- retina is variable resolution (in color and light sensitivity)
- CCD is fixed constant resolution
- Eye focus is limited in range compared to camera
- Camera can even zoom and change perspective!
- Eye has integrated noise-reduction
- Retina is randomized → reduced aliasing!
- Eye can move 3D with really high precision (yes, even on the face plane! limited, but still...)
Contents

A bit of anatomy

Few words on the basics
  Representation primitives
  Projections
  Camera parameters

Image processing
Geometric primitives

- **3D points:** \( \mathbf{x} = (x, y, z) \in \mathcal{R}^3 \) or \( \tilde{\mathbf{x}} = (\tilde{x}, \tilde{y}, \tilde{z}, \tilde{w}) \in \mathcal{P}^3 \) using homogeneous coordinates in a projective space (note \( \mathbf{x} \equiv (ky, ky, kz, kw) \ \forall \ k \))

- **3D lines:**
  - Segment: \( r = (1 - \lambda)\mathbf{p} + \lambda\mathbf{q}, \quad \mathbf{p}, \mathbf{q} \in \mathcal{R}^3 \)
  - Projective: \( r = \mu\tilde{\mathbf{p}} + \lambda\tilde{\mathbf{q}} \)

  No elegant representation

- **3D planes:**
  \( \tilde{\mathbf{m}} = (a, b, c, d) \Rightarrow \tilde{\mathbf{x}} \cdot \tilde{\mathbf{m}} = ax + by + cz + d = 0 \) (where \( \tilde{\mathbf{x}} \) is a normalized vector \( (x, y, z, 1) \))
2D Transformations

- **Translation:** $x' = x + t$ or $\bar{x}' = \begin{bmatrix} 1 & t \\ 0^T & 1 \end{bmatrix} \bar{x}$

- **Euclidean (rotation + translation):** $x' = R x + t$ or $x' = [R \ t] \bar{x}$

  where $R = \begin{bmatrix} \cos \phi & -\sin \phi \\ \sin \phi & \cos \phi \end{bmatrix}$ with $RR^T = I$ and $|R| = 1$

- **Similarity (scaled rotation):** $x' = sR x + t$

- **Affine:** $x' = A \bar{x}$ where $A = \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ 0 & 0 & 1 \end{bmatrix}$

- **Projective:** $\tilde{x}' = \tilde{H} \tilde{x}$, $\tilde{H} \in M^{3 \times 3}$
## 2D Transformations summary

<table>
<thead>
<tr>
<th>Transformation</th>
<th>Matrix</th>
<th>DoF</th>
<th>preserves</th>
</tr>
</thead>
<tbody>
<tr>
<td>translation</td>
<td>$[I \ t]_{2\times3}$</td>
<td>2</td>
<td>orientation</td>
</tr>
<tr>
<td>euclidean</td>
<td>$[R \ t]_{2\times3}$</td>
<td>3</td>
<td>lengths</td>
</tr>
<tr>
<td>similarity</td>
<td>$[sR \ t]_{2\times3}$</td>
<td>4</td>
<td>angles</td>
</tr>
<tr>
<td>affine</td>
<td>$[A]_{2\times3}$</td>
<td>6</td>
<td>parallelism</td>
</tr>
<tr>
<td>projective</td>
<td>$[\tilde{H}]_{3\times3}$</td>
<td>8</td>
<td>straight lines</td>
</tr>
</tbody>
</table>

One can begin asking himself: how difficult can be to recognize two things are the same after transformation? Transformation is applied by optical systems and positions!
## 3D Transformations summary

<table>
<thead>
<tr>
<th>Transformation</th>
<th>Matrix</th>
<th>DoF</th>
<th>preserves</th>
</tr>
</thead>
<tbody>
<tr>
<td>translation</td>
<td>$[I \ t]_{3\times4}$</td>
<td>3</td>
<td>orientation</td>
</tr>
<tr>
<td>rigid (euclidean)</td>
<td>$[R \ t]_{3\times4}$</td>
<td>6</td>
<td>lengths</td>
</tr>
<tr>
<td>similarity</td>
<td>$[sR \ t]_{3\times4}$</td>
<td>7</td>
<td>angles</td>
</tr>
<tr>
<td>affine</td>
<td>$[A]_{3\times4}$</td>
<td>12</td>
<td>parallelism</td>
</tr>
<tr>
<td>projective</td>
<td>$[\tilde{H}]_{4\times4}$</td>
<td>15</td>
<td>straight lines</td>
</tr>
</tbody>
</table>
Projections (1)

3D view

orthography
Projections (2)

- scaled orthography
- para-perspective
Projections (3)

- perspective
- object-centered
Projections (4)

Most used is 3D perspective:

$$\tilde{x} = \mathcal{P}_z(p) = \begin{bmatrix} x/z \\ y/z \\ 1 \end{bmatrix}$$

or

$$\tilde{x} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tilde{p}$$
Camera intrinsics

Map 3D rays to 2D pixels on sensor:

\[ \tilde{x}_s = K p_c, \quad K \in M^{3 \times 3} \]

*K* is the **calibration matrix**: position of sensor relative to lens

- Rotation
- Translation
- Scale \((S_x, S_y)\)
Camera intrisics and estrinsics
Camera matrix: intrinsics + extrinsics

Adds rotation and translation of whole camera:

\[ \mathbf{P} = \mathbf{K} \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix} \in \mathbb{M}^{3 \times 4} \]

The full rank version:

\[ \tilde{\mathbf{P}} = \begin{bmatrix} \mathbf{K} & 0 \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0^T & 1 \end{bmatrix} \]

is invertible and maps 3D world points \( \bar{\mathbf{p}}_w = (x_w, y_w, z_w, 1) \) to screen coordinates \( x_s = (x_s, y_s, 1, d) \)
Other camera parameters

- Lens distortion (barrel, pincushion, fisheye)

- Chromatic aberration (glass index of refraction not constant in wavelength)

- Vignetting (brightness diminishes near borders) can be at least partially overcome with proper camera models
Lens distortions correction

A known pattern with as many known camera parameters as possible is necessary for measuring lens characteristics.
Camera CCD structure
Camera image sensing pipeline
Contents

A bit of anatomy

Few words on the basics

Image processing
  Transforms
  Filters
  Image resolution
  Image transformation
Pixel transforms

- Operation pixel by pixel on one or more images (assumed of the same size): \( g(x) = h(f_0(x), ..., f_n(x)) \)
- Different operators: contrast, brightness, linear image blend, gamma correction
  - Often requires conversions between different color spaces!
  - Example changing luminosity: add value to RGB of each pixel affects contrast and hue as well; RGB \( \rightarrow \) XYZ \( \rightarrow \) increase Y luminance \( \rightarrow \) RGB
Few words on color spaces
Histogram equalization

Problem: Determine best values for brightness, contrast, tone, etc.
Common solution: individual color channels and luminance histograms equalization
Filters: linear operators

Most commonly used are linear filters:

\[ g(i, j) = \sum_{k,l} f(i - k, j - l) h(k, l) = \sum_{k,l} f(k, l) h(i - k, j - l) \]

to obtain blurring, sharpening, smoothing, binaryzation...

Note: boundary effects usually solved with different kinds of image padding (zero, constant, clamp, wrap...
Filters: nonlinear operators

- Non-linear operation: composition of filters becomes not commutative
- May or may not maintain locality
- Can be applied iteratively

More effective than linear filters for sharpening, blur, noise removal
Image resolution

Often it is needed to scale up or down images:

- Match size of different images (mix/compare/match)
- Visualization (Screen, print...)
- Appropriate resolution unknown: ex. face recognition, what’s the scale for the face?
Interpolation and Decimation

Simplest forms:

- Linear interpolation (upsample):

\[ g(i, j) = \sum_{k,l} f(k, l) h(i - rk, j - rl) \]

- Linear interpolation (downsample):

\[ g(i, j) = \frac{1}{r} \sum_{k,l} f(k, l) h\left(\frac{i - k}{r}, \frac{j - l}{r}\right) \]

The kernel \( h \) can be the same for interpolation and decimation! Better results can be obtained using higher order interpolation.
Multi-resolution representations: Pyramids

Pyramid of images at different resolution

Constructed scaling down with low-pass filter to avoid aliasing. Used in coarse to fine search operations, pattern recognition etc.
Geometric transformation

It may be needed to rotate/warp an image

- using any geometric transformation: affine, projection
- or mesh-based warping
- can be complicated (introduction of holes, aliasing, image degradation)
- can be computationally expensive (to avoid degradation)

Many techniques available to overcome and optimize the problem (vast literature)
Let’s assume we can do this efficiently on image pyramids
Recap

- Vision is a difficult task...
- ...which requires understanding more than precision (for real-life application as robotics)
- HVS comes from millions years of evolution aimed at maximizing real life performance
- We see what we want (need?) to see, not what’s there!
- We have advanced mathematics able to describe 3D world and many of sensing characteristics efficiently
- We have efficient methods to perform the basic image handling needed for more advanced tasks
- But much more than this is needed! (useful) vision is primarily a high level task.