Rectification and Depth Computation

CMPE 264: Image Analysis and Computer Vision
Spring’02, Hai Tao
4/26/02
Image correspondences

Two different problems

- Compute sparse correspondences with unknown camera motions – used for camera motion estimation and sparse 3D reconstruction
- Given camera intrinsic and extrinsic parameters, compute dense pixel correspondences (one correspondence per pixel) – used for recovering dense scene structure: one depth per pixel

We will focus on the second problem in this lecture.

Figure 7.1  An illustration of the correspondence problem. A matching between corresponding points of an image pair is established (only some correspondences are shown).
Examples of dense depth recovery

Figure 7.2  (a) One image from a stereo pair of Emanuele Trucco’s face. (b) 3-D rendering of stereo reconstruction. Courtesy of the Turing Institute, Glasgow (UK).
Examples of dense depth recovery

(a) 

(b) 

(c)
If we know the camera matrix and the camera motion, for any pixel $p_l$ in the left image, its correspondence must lie on the epipolar line in the right image.

This suggests a method to compute the depth (3D position) of each pixel in the left image:

- For each pixel $p_l$ in the left image, search for the best match $p_r$ along its epipolar line in the right image.
- The corresponding 3D scene points is the intersection of $O_l p_l$ and $O_r p_r$. This process is called triangulation.
Rectification

- The search of the best match along the epipolar line can be very efficient for a special configuration where the epipolar lines become parallel to the horizontal image axis and collinear (the same scan lines in both images).
- For such a configuration, to find the correspondence of pixel \((x,y)\) in the right image, only pixels \((*,y)\) are considered.
- This special configuration is called a simple or standard stereo system.
- In such a system, the 3D transformation between the two cameras is
  \[ P_r = P_l - [T_x,0,0]^T \]
Rectification

Images taken from two cameras with arbitrary relative motion $R, T$ can be rectified. The resultant images transformed so that they are as if taken from a standard stereo system with the two camera centers unchanged.
Image transformation for a rotating camera

- Question: From the original image, can we compute the image taken from the camera center at the same position but with different orientation? If the answer is yes, we can rotate the two cameras so that the resultant images are rectified.

- Suppose the camera rotation matrix is $R$. For a image point, the corresponding 3D scene point is $K^{-1}p^TZ_c$. After rotation, the coordinates of this point is $RK^{-1}p^TZ_c$. The new homogeneous image coordinates are $p' \approx KRK^{-1}p^T$. This can be rewritten as:

$$p' \approx Hp^T$$

where $H = KRK^{-1}$ is a $3 \times 3$ transformation matrix (Homography).

- The image transformation caused by a camera rotation is a 2D homography.
Rectification algorithm

- Build rectification matrix $R_{rect}$ as

$$R_{rect} = \begin{pmatrix}
    e_1^T \\
    e_2^T \\
    e_3^T
\end{pmatrix}$$

where $e_1 = \frac{T}{\|T\|}$, $e_2 = -e_1 \times [0,0,1]^T = \frac{1}{\sqrt{T_x^2 + T_y^2}}[-T_y, T_x, 0]^T$, and $e_3 = e_1 \times e_2$

- Rotate left camera by $R_{rect}$ and right camera by $R_{rect}R$ using the corresponding homographies derived in the previous slide
Disparity and depth in a simple stereo system

- In a simple/standard binocular stereo system, the correspondences are along the same scan line in the two images. The following figure shows the relationship between the depth $Z$ and the disparity $d = x_r - x_l$.

- The following relationship can be easily proved:

$$ Z = f_x \frac{T}{d} $$

- The depth is inversely proportional to the disparity. The closer the object, the larger the disparity.

- For a scene point at infinity, the disparity is 0.

Figure 7.4 A simple stereo system. 3-D reconstruction depends on the solution of the correspondence problem (a); depth is estimated from the disparity of corresponding points (b).
Finding correspondences – correlation-based method

- **Assumptions**
  - Most scene points are visible from both viewpoints
  - Corresponding image regions are similar

- **Correlation_matching_algorithm**
  Let $p_l$ and $p_r$ be pixels in the left and right image, $2W+1$ is the width of the correlation window, $[-L,L]$ is the disparity search range in the right image for $p_l$
  - For each disparity $d$ in the range of $[-L,L]$ compute the similarity measure $c(d)$
  - Output the disparity with the maximum similarity measure
Finding correspondences – correlation-based method

Figure 7.5 An illustration of correlation-based correspondence. We look for the right-image point corresponding to the central pixel of the left-image window. This window is correlated to several windows of the same size in the right image (only a few are shown here). The center of the right-image window producing the highest correlation is the corresponding point sought.
Finding correspondences – correlation-based method

- **Different similarity measures**
  
  - **Sum of squared differences (SSD)**
    
    \[
    c(d) = - \sum_{i=-W}^{W} \sum_{j=-W}^{W} (I_l(x_l + i, y_l + j) - I_r(x_l + d + i, y_l + j))^2
    \]
  
  - **Sum of absolution differences (SAD)**
    
    \[
    c(d) = - \sum_{i=-W}^{W} \sum_{j=-W}^{W} |I_l(x_l + i, y_l + j) - I_r(x_l + d + i, y_l + j)|
    \]
  
  - **Normalized cross-correlation**
    
    \[
    c(d) = \frac{C_{lr}}{\sqrt{C_{ll}C_{rr}}}
    \]

  where

  \[
  C_{lr} = \sum_{i=-W}^{W} \sum_{j=-W}^{W} [I_l(x_l + i, y_l + j) - \bar{I}_l(x_l, y_l)][I_r(x_l + d + i, y_l + j) - \bar{I}_r(x_l + d, y_l)]
  \]

  \[
  C_{ll} = \sum_{i=-W}^{W} \sum_{j=-W}^{W} [I_l(x_l + i, y_l + j) - \bar{I}_l(x_l, y_l)]^2
  \]

  \[
  C_{rr} = \sum_{i=-W}^{W} \sum_{j=-W}^{W} [I_r(x_l + d + i, y_l + j) - \bar{I}_r(x_l + d, y_l)]^2
  \]
Finding correspondences – feature-based method

- Search is restricted to a set of features in two images, such as edges, corners, etc.
- A similarity measure is used for matching features
- Constraints such as the uniqueness constraint (each feature can only have one match) can be used
- Algorithm_feature_matching
  - Compute the similarity measure between \( f_l \) and each feature in the right image
  - Select the right image feature with the largest similarity measure and output the disparity
- Sample similarity measure for line segments
  \[
  S_l = \frac{w_0(l_l - l_r)^2 + w_1(\theta_l - \theta_r)^2 + w_2(m_l - m_r)^2 + w_3(c_l - c_r)^2}{w_0(l_l - l_r)^2 + w_1(\theta_l - \theta_r)^2 + w_2(m_l - m_r)^2 + w_3(c_l - c_r)^2}
  \]
  where \( l \) is the length of the line segment, \( \theta \) the orientation, \( m \) the midpoint, and \( c \) the average contrast along the edge line