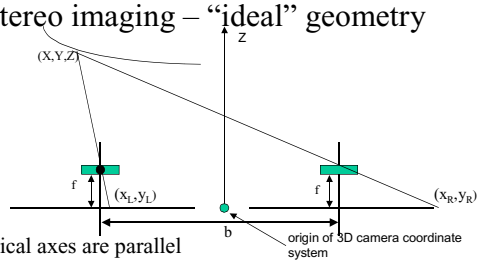
Stereo imaging – "ideal" geometry

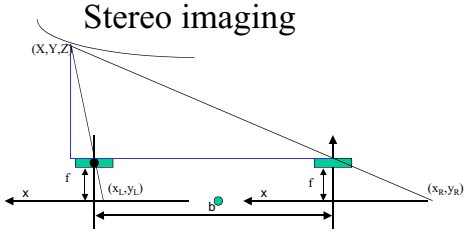


- Optical axes are parallel
- Optical axes separated by **baseline**, b.
- Line connecting lens centers is perpendicular to the optical axis, and the x axis is parallel to that line
- 3D coordinate system is a **cyclopean** system centered between the cameras

Stereo imaging

- (X,Y,Z) are the coordinates of P in the Cyclopean coordinate system.
- The coordinates of P in the left camera coordinate system are $(X_L,Y_L,Z_L) = (X-b/2, Y, Z)$
- The coordinates of P in the right camera coordinate system are $(X_R,Y_R,Z_R) = (X+b/2, Y, Z)$
- So, the x image coordinates of the projection of P are
  - $x_L = (X+b/2)f/Z$
  - $x_R = (X-b/2)f/Z$
- Subtracting the second equation from the first, and solving for Z we obtain:
  - $Z = bf/(x_L - x_R)$
- We can also solve for X and Y:
  - $X = b(x_L + x_R)/2(x_L-x_R)$
  - $Y = by/(x_L-x_R)$

Stereo imaging

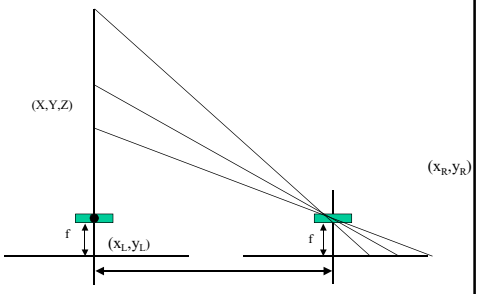


- $x_L - x_R$ is called the **disparity**, d, and is always negative
- $X= (b[x_R + x_L]/2)/ d \quad Y = by/ d \quad Z = bf/d$

Stereo imaging

- Distance is inversely proportional to |disparity|
  - disparity of 0 corresponds to points that are infinitely far away from the cameras
  - in digital systems, disparity can take on only integer values (ignoring the possibility of identifying point locations to better than a pixel resolution)
  - so, a disparity measurement in the image just constrains distance to lie in a given range
- Disparity is directly proportional to b
  - the larger b, the further we can accurately range
  - but as b increases, the images decrease in common field of view

Range versus disparity



Applications of stereo

- Photogrammetry
  - Creation of digital elevation models from high resolution aerial imagery
- Visual navigation
  - Obstacle detection
    - Positive versus negative obstacles
- Creating models for graphics applications
  - For objects difficult to design using CAD systems

Stereo imaging

- Definition: A scene point, P, visible in both cameras gives rise to a pair of image points called a **conjugate pair**.
  - the conjugate of a point in the left (right) image must lie on the same image row (line) in the right (left) image because the two have the same y coordinate
  - this line is called the **conjugate line**.
  - so, for our simple image geometry, all conjugate lines are parallel to the x axis

A more practical stereo image model

- Difficult, practically, to
  - have the optical axes parallel
  - have the baseline perpendicular to the optical axes
- Also, we might want to tilt the cameras towards one another to have more overlap in the images
- Calibration problem - finding the transformation between the two cameras
  - it is a rigid body motion and can be decomposed into a rotation, **R**, and a translation, **T**.

General stereo matching

- Assume relative orientation of cameras is known
- An image point (x_L, y_L) in the left coordinate system is the image of some point on a ray through the origin of the left camera coordinate system. The points on this ray all have coordinates of the form

  $X_L = x_L s \quad Y'_L = y_L s \quad Z_L = fs$ since all points of this form project onto (x_L, y_L)
- In the right image system, the coordinates of points on this ray are:

  $X_R = (r_{11}x_L + r_{12}y_L + r_{13}f)s + t_1$

  $Y_R = (r_{21}x_L + r_{22}y_L + r_{23}f)s + t_2$

  $Z_R = (r_{31}x_L + r_{32}y_L + r_{33}f)s + t_3$
- These points project onto $x_R/f = X_R/Z_R$ and $y_R/f = Y_R/Z_R$

General stereo matching

- Let

  $X_R = as + u$

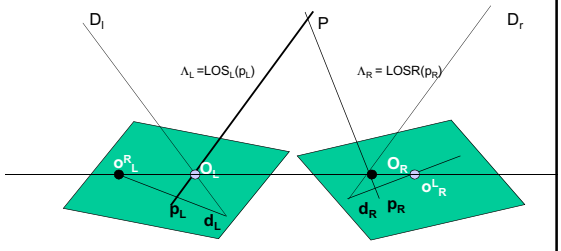
  $Y_R = bs + v$

  $Z_R = cs + w$
- Then

  $x_R/f = (as+u)/(cs+w)$

  $y_R/f = (bs+v)/(cs+w)$
- This is a straight line connecting the point
  - (u/w, v/w) which occurs for s=0 and is the image of the left camera center in the right camera coordinate system, to
  - (a/c, b/c) which occurs as s approaches infinity,the vanishing point for the ray

General stereo geometry



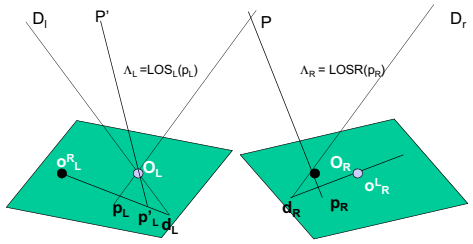
General stereo imaging

- Point P lies somewhere on the ray (line) Λ_L from p_L through O_L
  - but from the left image alone, we do not know where on this ray P lies
- Since the perspective projection of a line is a line, the perspective projection of Λ_L in the right image is a line

- The "first" point on Λ_L that might correspond to P is O_L
  - any point closer to the left image than O_L would be between the lens and the image plane, and could not be seen
  - the perspective projection of O_L in the right camera is the point o^L_R.
- The "last" point on Λ_L that might correspond to P is the point "infinitely" far away along the ray Λ_L
  - but its image is the vanishing point of the ray Λ_L in the right camera, d_R
  - any other possible location for P will project to a point in R on the line joining o^L_R to d_R.

- Given any point, p_L, in the left image of a stereo pair, its **conjugate point** must appear on a **line** in the right image
  - Furthermore, all of the conjugate lines for all of the points in the left image must pass through a common point in the right image
    - this is image of the left lens center in the right image
    - this point lies on the line of sight for **every** point in the left image
    - therefore, the conjugate lines must all contain (i.e., pass through) the image of this point
    - This point is called an **epipole.**
  - Finally, the conjugate line for p_L must also pass through the vanishing point in the right image for the line of sight through p_L

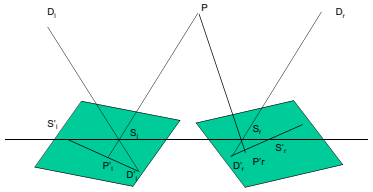
General stereo imaging



- Remember that
  - any three non-collinear points define a plane, and
  - the intersection of two planes is a straight line
- The points O_L, p_L, and o^R_L are three non-collinear points, so they form a plane, Π
  - the line Λ_L lies on this plane, since two points on the line lie on the plane
- The intersection of this plane with the right image plane is the conjugate line of p_L
  - and this would be the image of **any** line on this plane
- Let p'_L be some other point on the line joining p_L and o^R_L.
  - the line of sight through p'_L to P' lies on Π since two points on that line (p_L and o^R_L) lie on the plane

- ***Therefore***, the conjugate line for p'_L must be the same line as the conjugate line for p_L, or for **any other point** on the line containing p_L and o^R_L.
- The lines p_L - o^R_L and p_R - o^R_L are called **epipolar lines**

- Given any point, p_L, in the right image
  - it lies on a line containing the image of the right camera center in the left image, and
  - it has a conjugate line in the right camera
- Given **any point** on either of these two lines, its conjugate pair must lie on the other line
- These lines are called epipolar lines.

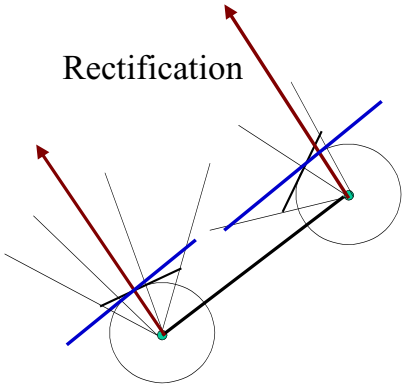


- Points S'_l, P'_l, and D'_l form a straight line
  - They all lie on the image of the line containing S_r, P and that line's vanishing point
  - There is a similar line in the right image
- These lines are called **conjugate** or **epipolar** lines and constrain matching
- All epipolar lines pass through one point - the image of the other lens center (which might not be in the finite field of view)

Rectification

- Process of transforming a stereo pair taken under general conditions into the "ideal" configuration
- Involves a rotation of one image so that the optical axes of the two image coordinate systems are parallel
  - Simplifies computational structure of stereo matching algorithm
  - But requires interpolation to create rotated image and can create a "large" rectified image if the rotation angles are large.

Rectification

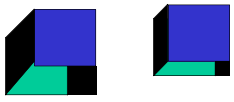


Stereo correspondence problem

- Given a point,p, in the left image, find its conjugate point in the right image
  - called the stereo correspondence problem
- What constraints simplify this problem?
  - Epipolar constraint - need only search for the conjugate point on the epipolar line
  - Disparity sign constraint - need only search the epipolar line to the "right" of the vanishing point in the right image of the ray through p in the left coordinate system
  - Continuity constraint - if we are looking at a continuous surface, images of points along a given epipolar line will be ordered the same way

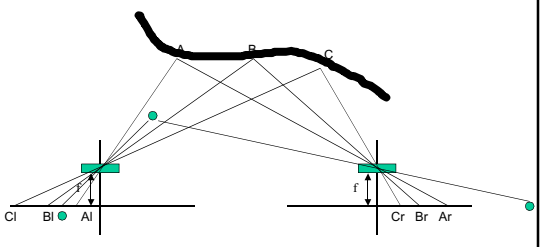
Stereo correspondence problem

- Similarity of correspondence functions along adjacent epipolar lines
- Disparity gradient constraint - disparity changes slowly over most of the image.
  - Exceptions occur at and near occluding boundaries where we have either discontinuities in disparity or large disparity gradients as the surface recedes away from sight.



As the blue surfaces become more slanted, they occupy a smaller area of the image

Continuity constraint

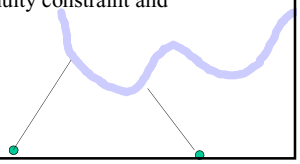


Why is the correspondence problem hard?

- Foreshortening effects
  - Match images at low resolutions
  - User resulting disparity map to warp images
  - Match at next higher resolution
- A square match window in one image will be distorted in the other if disparity is not constant - complicates correlation

Why is the correspondence problem hard

- Occlusion
  - Even for a smooth surface, there might be points visible in one image and not the other
  - Consider aerial photo pair of urban area - vertical walls of buildings might be visible in one image and not the other
  - Scene with depth discontinuities (lurking objects) violate continuity constraint and introduces occlusion



Why is the correspondence problem hard?

- Variations in intensity between images due to
  - noise
  - specularities
  - shape-from-shading differences
- Coincidence of edge and epipolar line orientation
  - consider problem of matching horizontal edges in an ideal left right stereo pair
  - will obtain good match all along the edge
  - so, edge based stereo algorithms only match edges that cross the epipolar lines

How do we solve the stereo correspondence problem?

- Neighborhood intensity correlation
- Edge matching
  - Multiresolution techniques
  - Edge correlation and distance transforms
  - Smooth transformation of contours
- Global analysis
  - Dynamic programming

Grey level correlation

- For point (x,y) in left image
  - Consider its n x n neighborhood M
  - For all possible disparities, d, let N be the n x n neighborhood of (x+d, y) in the right image
- How can we measure the similarity of M and N
  - sum of squared differences:

$$SSD = \sum_{i=1}^{n}\sum_{j=1}^{n}[M(i,j) - N(i,j)]^2$$

  - correlation:

$$C = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} M(i,j)N(i,j)}{[\sum_{i=1}^{n}\sum_{j=1}^{n} M(i,j)^2 \sum_{i=1}^{n}\sum_{j=1}^{n} N(i,j)^2]^{1/2}}$$

Grey level correlation

- This correlation measure takes on values in the range [0,1]
  - it is 1 if and only if N = cM for some constant c
    - so N can be uniformly brighter or darker than M and the correlation will still be high.
    - the SSD is sensitive to these differences in overall brightness
  - The first term in the denominator, $\Sigma\Sigma M^2$ is the same for all disparities and can be ignored

Problems

- How do we choose n?
  - Algorithm assumes that disparity is constant within the n x n window since the size of the window is the same in the left and right image.
    - So, if we choose too large an n, this assumption will be violated.
  - If we choose n too small, then there won't be enough pixels to obtain peaked matches
  - Is there a better way to match regions that relaxes this assumption
- What do we do in areas of the image where the gray level is nearly constant?
  - Correlations will be very noisy in these regions.

Edge correlation

- Apply edge detector to image
- Represent each edge by (g_l, g_r. Θ)
  - g_l and g_r are gray levels to "left" and "right" of edge
  - Θ is orientation of edge
- Match each edge in left image to all possible matching edges in right image
  - Match score can be based on g_l and g_r or on their signed difference (contrast is positive or negative)
    - At occluding edges we would expect one of the gray levels to not match at the conjugate point.
- Problem: Need to use a small scale edge detector to get precise localization of edges
  - But then there might be many good matches between the images over the range of possible disparities

Marr-Poggio-Grimson stereo

- Multi-scale approach to stereo correspondence.
  - First apply a large scale edge detector to the left and right images
    - Relatively few edges are detected, and their localization is poor
    - But matching is easier
  - Then use progressively finer and finer edge detectors
    - Use the rougher disparity estimates from previous steps to center small search window at current scale
      - Need to interpolate disparities to obtain centering information at each edge at current scale
      - Search window size is on the order of the scale of the edge detector, guaranteeing that the window contains at most a few edges.

Problems with both approaches

- How do we evaluate the significance of a correlation score?
  - What is P[match:correlation] and P[~match:correlation]
    - Likelihood ration is P[match:correlation] / P[~match:correlation]
  - When do we decide that a pixel has no match (perhaps it is occluded)?
- How can we guarantee that the matches chosen for different pixels are consistent with the ordering constraint?
  - Simply taking the disparity with maximum likelihood will not guarantee this.
- How can we utilize continuity of edges across epipolar lines to find good matches

Dynamic programming – matrix multiplication

- Suppose we have four matrices A (50x10), B(10x40), C(40x30) and D(30x5) and we want to compute ABCD. There are five ways to do this:
  1) A((BC)D) - requiring 16000 multiplication (12000 to compute the 10x30 matrix BC, 1500 more to compute the 10x5 matrix BCD and then 2500 more to compute ABCD)
  2) A(B(CD)) - 10,500
  3) (AB)(CD) - 36,000
  4) (((AB)C)D) - 87,500
  5) (A(BC))D - 34,500

Matrix multiplication

- So, there can be a BIG difference in the amount of work it takes to do the multiplication
- But the number of possible orderings grows quickly with n, the number of matrices

$$T(n) = \sum_{i=1}^{n-1} T(i)T(n-i)$$

- Suppose last multiplication performed is
  - $(A_1A_2\ldots A_i)(A_{i+1}A_{i+2}\ldots A_n)$
  - There are T(i) ways to compute $(A_1A_2\ldots A_i)$
  - There are T(n-i) ways to compute $(A_{i+1}A_{i+2}\ldots A_n)$
  - There are n-1 places we could have cut the problem into two
- Solution is Catalan numbers, which grow exponentially

A dynamic programming solution

- Let c_i be the number of columns in matrix A_i
  - then A_i has c_{i-1} rows
  - A_0 has c_0 rows
- Let $m_{L,R}$ be the number of multiplications needed to multiple $A_L A_{L+1} \ldots A_{R-1} A_R$
  - $m_{L,L} = 0$
  - Suppose the LAST multiplication performed is

    $(A_L A_{L+1} \ldots A_i)(A_{i+1} \ldots A_{R-1} A_R)$

    Then the number of multiplications performed is

    $m_{Li} + m_{i+1,R} + c_{L-1} c_i c_R$

A dynamic programming solution

- Define $M_{L,R}$ to be the number of multiplications required in an *optimal ordering* of matrices.

$$M_{L,R} = \min_{L \le i \le R} \{ M_{L,i} + M_{i+1,R} + c_{L-1} c_i c_R \}$$

- This expression translates directly into a recursive program
  - that would run forever
- But there are only a total of about $n^2/2$ possible values for the $M_{L,R}$ that EVER need to be computed
  - if R-L = k, then the only values needed in the computation of $M_{L,R}$ are $M_{x,y}$ with y-x < k

The program

$A_1 = 3x5, A_2 = 5x8, A_3 = 8x4, A_4 = 4x3$

```
for L = 1 to n
   M_{L,L} = 0;
for k = 1 to n-1   {k is R-L}
   for L = 1 to n-k
   begin
      R = L + k
      M_{L,R} = maxint
      for i = L to R-1
         M' = M_{Li} + M_{i+1,R} + c_{L-1} c_i c_R
         if M' < M_{L,R} then M_{L,R} = M'
```

M

R \ L	1	2	3	4
4		256	96	0
3	220	160	0	
2	120	0		
1	0			

Application to stereo matching

- Want to find the "best" match, M, between two epipolar lines.
  - M maps pixels from the "left" epipolar line to pixels on the right epipolar line.
  - Ordering constraint requires that if i< j, then M(i) < M(j)
  - Uniqueness constraint requires that M(i) = M(j) implies i=j

Dynamic programming for stereo

- Let $P_s(l,r)$ be the probability that pixel l from the left matches with pixel r from the right, and let $P_f(l,r)$ be the probability that pixel l from the left image does NOT match pixel r in the right
  - Proportional to the correlation of the neighborhoods surrounding l and r.
  - The likelihood ratio, $m(l,r) = P_s(l)/P_f(r)$ is used to compute the the value of M(l,r)
- Occlusion – some points in the left image will not be visible in the right image
  - Have an occlusion probability, β, that allows us to assign a pixel in the left image as "null"
- Now, find the M that satisfies the uniqueness and ordering constraints and has maximal score.

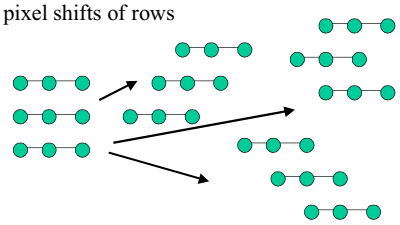
Dynamic programming for stereo

- Suppose there are n pixels on the epipolar lines.
- Let $C_{i,j}$ be the cost of the optimal matching of the first i pixels from the left epipolar line with the first j pixels from the right epipolar line.

$$C_{i+1,j+1} = \min(C_{i,j} + m(i,j), C_{i+1,j} + \beta, C_{i,j+1} + \beta)$$

Other uses for dynamic programming in stereo

- Neighborhood matching
  - Match nxn neighborhood of L(i,j) allowing one pixel shifts of rows

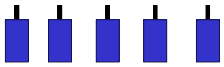


Other uses for dynamic programming in stereo

- Edge continuity across epipolar lines
  - Compute k best matches per epipolar line using DP algorithm
  - Then, search across the epipolar lines for best match along each line that also enforces edge continuity across lines

Multi-camera stereo

- Slider stereo
  - single baseline
  - equal separation of camera
  - can track features from left to right or vote on best match



Multi camera stereo

- multiple baseline systems
  - camera pairs have different epipolar geometry
  - must find matches in consistent positions in most or all images
  - allows matching of edges in all orientations