Projections and Transformations

DD2423 Image Analysis and Computer Vision

Mårten Björkman

Robotics, Perception and Learning Division
School of Electrical Engineering and Computer Science

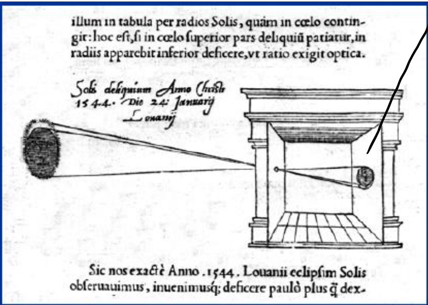
November 4, 2021

Topics for today

- Perspective projection
  - properties
  - approximations
- Homogeneous coordinates
- Image transformations
- Neighborhood concepts
- Connectivity, connected components
- Distance measures and transforms

Pinhole camera or “Camera Obscura”

(opposite position).


illum in tabula per radios Solis, quàm in cœlo contin-
git: hoc eſt, ſi in cœlo ſuperior pars deliquiū patiatur, in
radiis apparebit inferior deficere, vt ratio exigit optica.

Sic nos exactè Anno .1544. Louanii eclipſim Solis
obſeruauimus, inuenimusq; deficere paulò plus q̄ dex-

"When images of illuminated objects ... penetrate through a small hole into a very dark room ... you will see [on the opposite wall] these objects in their proper form and color, reduced in size ... in a reversed position, owing to the intersection of the rays".

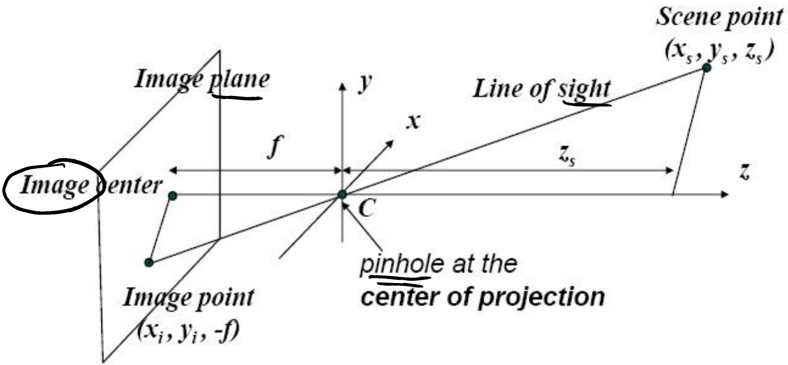
Leonardo Da Vinci

http://www.acmi.net.au/AIC/CAMERA_OBSCURA.html (Russell Naughton)

Pinhole camera and perspective projection

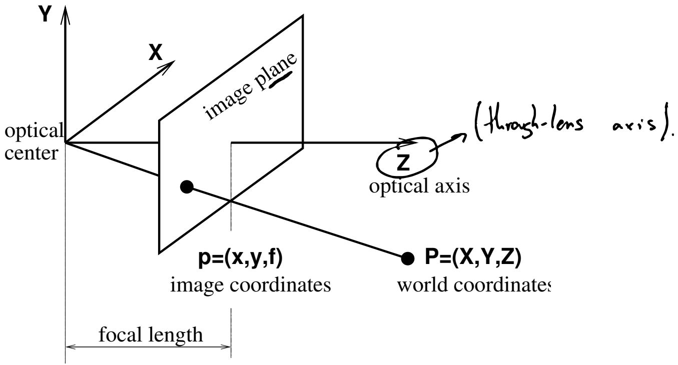
- A mapping from a three dimensionsal (3D) world onto a two dimensional (2D) plane in the previous example is called perspective projection.
- A pinhole camera is the simplest imaging device which captures the geometry of perspective projection.
- Rays of light enter the camera through an infinitesimally small aperture.
- The intersection of light rays with the image plane form the image of the object.

Perspective projection



- The point on the image plane that corresponds to a particular point in the scene is found by following the line that passes through the scene point and the center of projection

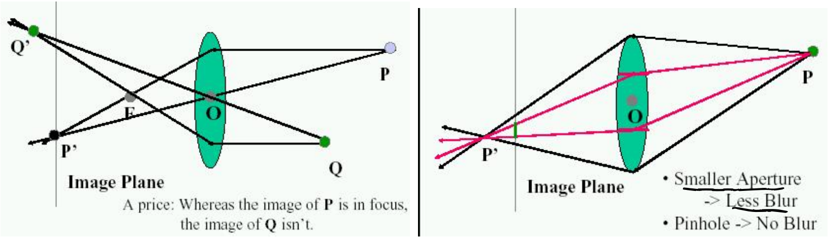
Pinhole camera - Perspective geometry



- The image plane is usually modeled in front of the optical center.
- The coordinate systems in the world and in the image domain are parallel. The optical axis is $\perp$ image plane.

Lenses

- Purpose: gather light from from larger opening (aperture)
- Problem: only light rays from points on the focal plane intersect the same point on the image plane ⟶ (not in focus).
- Result: blurring in-front or behind the focal plane
- Focal depth: the range of distances with acceptable blurring



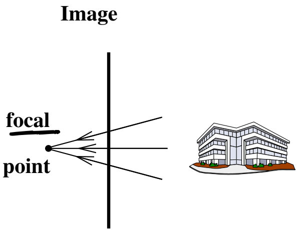
Imaging geometry - Basic camera models

- **Perspective projection** (general camera model)

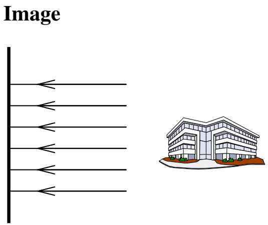
  All visual rays converge to a common point - the focal point

- **Orthographic projection** (approximation: distant objects, center of view)

  All visual rays are perpendicular to the image plane

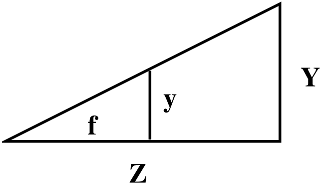


Perspective projection



Orthographic projection

Projection equations



- Perspective mapping

$$\frac{x}{f} = \frac{X}{Z}, \quad \frac{y}{f} = \frac{Y}{Z}$$

f : focal

- Orthographic projection

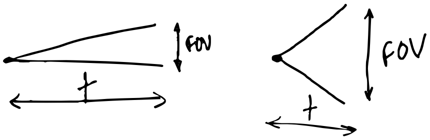
$$x = X, \quad y = Y$$

- Scaled orthography - Z_0 constant (representative depth)

$$\frac{x}{f} = \frac{X}{Z_0}, \quad \frac{y}{f} = \frac{Y}{Z_0}$$

Akira Kurusawa used telescopic lenses to ’flatten’ his movies to create more action, when you lose the sense of depth and distances.

$\downarrow$ focal length $\propto$ large FOV :



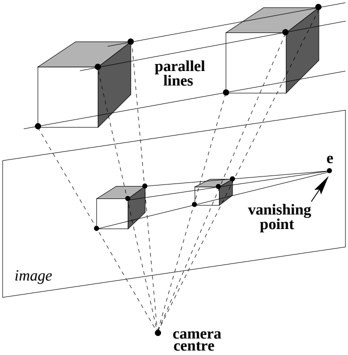
Fish-eye lens video example

The field of view is wide, but it results in radial distortion, when straight lines in the world are no longer straight in the image.

Perspective transformation

- A perspective transformation has three components:
  - Rotation - from world to camera coordinate system
  - Translation - from world to camera coordinate system
  - Perspective projection - from camera to image coordinates
- Basic properties which are preserved:
  - lines project to lines,
  - collinear features remain collinear,
  - tangencies,
  - intersections.

Perspective transformation (cont)



Each set of parallel lines meet at a different vanishing point - vanishing point associated to this direction. Sets of parallel lines on the same plane lead to collinear vanishing points - the line is called the horizon for that plane.

Homogeneous coordinates

- Model points (X, Y, Z) in $\mathcal{R}^3$ world by (kX, kY, kZ, k) where k is arbitrary $\neq 0$, and points (x, y) in $\mathcal{R}^2$ image domain by (cx, cy, c) where c is arbitrary $\neq 0$.

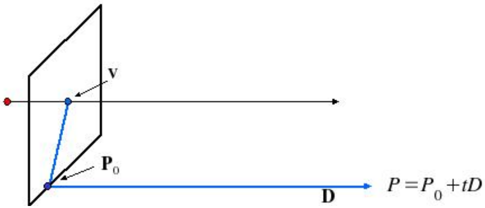
$$(X, Y, Z) \rightarrow (kX, kY, kZ, k)$$

- Equivalence relation: $(k_1 X, k_1 Y, k_1 Z, k_1)$ is regarded as the same as $(k_2 X, k_2 Y, k_2 Z, k_2)$. In 2D this means that all points along a ray (cx, cy, c) in homogeneous coordinates are equivalent.
- To go back to Euclidean coordinates: → only "k" changes.

$$(X, Y, Z, W) \rightarrow (X/W, Y/W, Z/W)$$

- Possible to represent "points in infinity" with homogeneous coordinates $(X, Y, Z, 0)$ - intersections of parallel lines.

Computing vanishing points



$$P_t = \begin{bmatrix} P_x + tD_x \\ P_y + tD_y \\ P_z + tD_z \\ 1 \end{bmatrix} \simeq \begin{bmatrix} P_x/t + D_x \\ P_y/t + D_y \\ P_z/t + D_z \\ 1/t \end{bmatrix} \to P_\infty = \begin{bmatrix} D_x \\ D_y \\ D_z \\ 0 \end{bmatrix}, \quad \text{when } t \to \infty$$

Properties:

- P_∞ is a point at infinity, v is its image projection
- They depend only on line direction D
- Parallel lines $P_0 + tD$ and $P_1 + tD$ intersect at P_∞

Homogeneous coordinates (cont)

In homogeneous coordinates the projection equations can be written

$$\begin{pmatrix} cx \\ cy \\ c \end{pmatrix} = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} kX \\ kY \\ kZ \\ k \end{pmatrix} = \begin{pmatrix} fkX \\ fkY \\ kZ \end{pmatrix}$$

Image coordinates obtained by normalizing the third component to one (divide by $c = kZ$).

$$x = \frac{xc}{c} = \frac{fkX}{kZ} = f\frac{X}{Z}, \quad y = \frac{yc}{c} = \frac{fkY}{kZ} = f\frac{Y}{Z}$$

Transformations in homogeneous coordinates

- Translation

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \to \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + \begin{pmatrix} \Delta X \\ \Delta Y \\ \Delta Z \end{pmatrix}$$

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \to \begin{pmatrix} 1 & 0 & 0 & \Delta X \\ 0 & 1 & 0 & \Delta Y \\ 0 & 0 & 1 & \Delta Z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

- Scaling

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \to \begin{pmatrix} S_X & 0 & 0 & 0 \\ 0 & S_Y & 0 & 0 \\ 0 & 0 & S_Z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

- Rotation around the Z axis

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

- Mirroring in the XY plane

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Common case: Rigid body transformations (Euclidean)

$$\begin{pmatrix} X' \\ Y' \\ Z' \end{pmatrix} \rightarrow R \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + \begin{pmatrix} \Delta X \\ \Delta Y \\ \Delta Z \end{pmatrix}$$

where R is a rotation matrix ($R^{-1} = R^T$) is written

$$\begin{pmatrix} X' \\ Y' \\ Z' \\ 1 \end{pmatrix} = \begin{pmatrix} & R & & \Delta X \\ & & & \Delta Y \\ & & & \Delta Z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

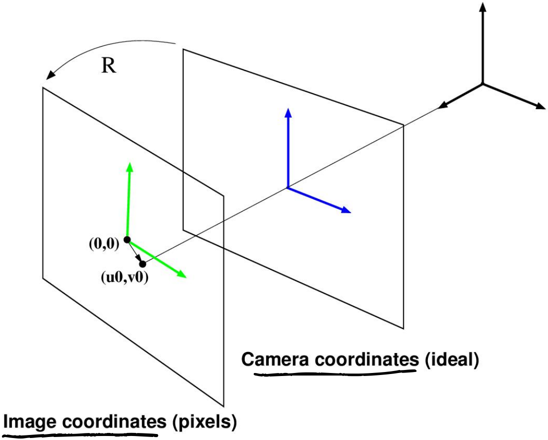
Consider world coordinates $(X', Y', Z', 1)$ expressed in a coordinate system not aligned with the camera coordinate system

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{pmatrix} & R & & \Delta X \\ & & & \Delta Y \\ & & & \Delta Z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X' \\ Y' \\ Z' \\ 1 \end{pmatrix} = A \begin{pmatrix} X' \\ Y' \\ Z' \\ 1 \end{pmatrix}$$

Perspective projection (more general later)

$$c \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = PA \begin{pmatrix} X' \\ Y' \\ Z' \\ 1 \end{pmatrix} = M \begin{pmatrix} X' \\ Y' \\ Z' \\ 1 \end{pmatrix}$$

Intrinsic camera parameters



Due to imperfect placement of the camera chip relative to lens system, there is always a small relative rotation and shift of center position.

Intrinsic camera parameters

A more general projection matrix allows image coordinates with an offset origin, non-square pixels and skewed coordinate axes.

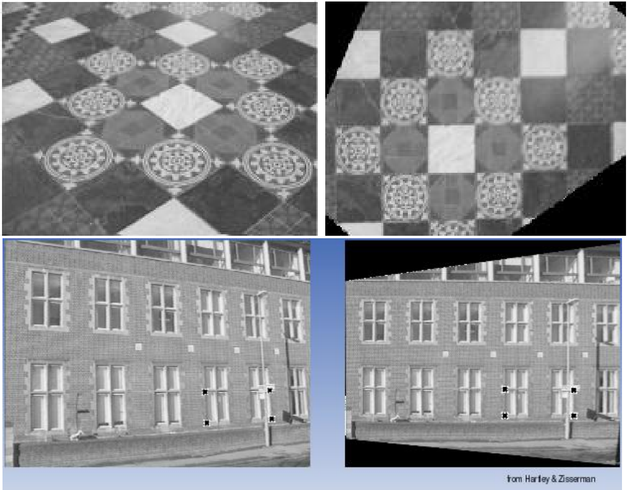
$$K = \begin{pmatrix} f_u & \gamma & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{pmatrix}, \quad P = \begin{pmatrix} K & 0 \end{pmatrix} = \begin{pmatrix} f_u & \gamma & u_0 & 0 \\ 0 & f_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Five variables known as the camera's intrinsic parameters

- Focal length (f_u, f_v), often assumed to be equal
- Skew γ, often close to zero
- Principal point (u_0, v_0), often close to image center

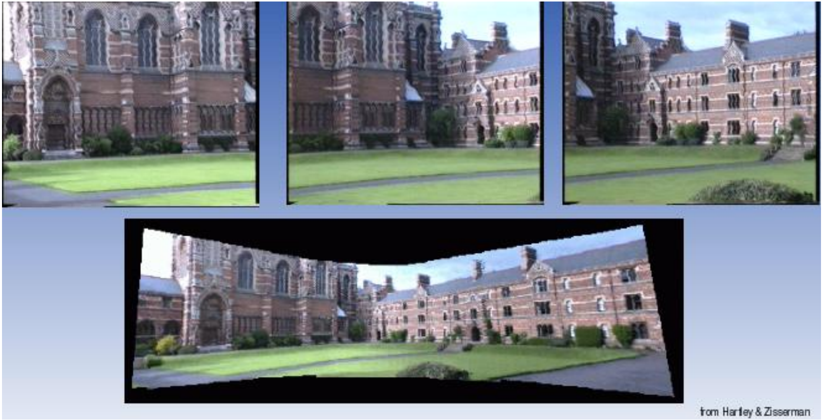
The process of finding the camera matrix K for an unknown camera is called camera calibration.

Example: Perspective mapping



Once the camera is calibrated you can e.g. map a plane from one image to another, as if the camera were placed elsewhere.

Example: Mosaicing - creating a panorama



Since the world is not a plane, you will get small artifacts.

Image stitching video example

Image stitching can be used to combine images into a larger mosaic of higher resolution.

Exercise

Assume you have a point at $(3, -2, 8)$ with respect to the cameras coordinate system. What are the image coordinates, if the image has a size $(w, h) = (640, 480)$ and origin in the upper-left corner, and the focal length is $f = 480$?

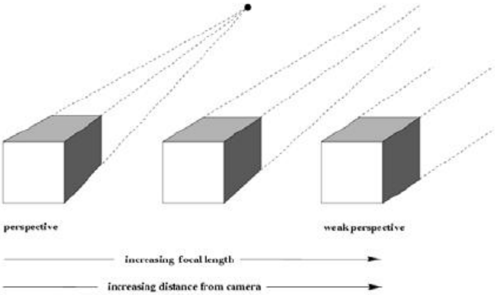
Assume you have a point at $(3, -2, 8)$ with respect to the cameras coordinate system. What are the image coordinates, if the image has a size $(w, h) = (640, 480)$ and origin in the upper-left corner, and the focal length is $f = 480$?

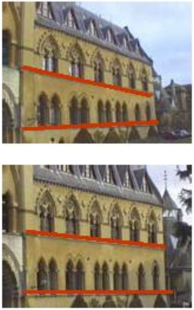
Answer:

$$x = f\frac{X}{Z} + \overbrace{\frac{w}{2}} = (480 * 3/8 + 640/2) = 500$$

$$y = \underbrace{f\frac{Y}{Z}}_{\text{img}} + \frac{h}{2} = (-480 * 2/8 + 480/2) = 120$$

Weak-perspective cameras





Assume you move further away, put zoom in (increase focal length), then parallel lines in 3D become more and more parallel in 2D too. We lose the sense of perspective. The world appears flat.

Approximation: affine camera

Then we could use an alternative model - an affine camera model.

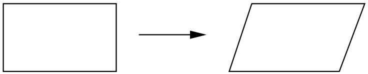
- A linear approximation of perspective projection

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

→ preferable to use as few params as possible, while still close to reality.

- Basic properties
  - linear transformation (no need to divide at the end)
  - parallel lines in 3D mapped to parallel lines in 2D

Angles are not preserved!



Summary of projection models

- Perspective (11 degrees of freedom): – under normal condition

$$M = \begin{pmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{pmatrix}$$

- Affine (8 degrees of freedom): – when the perspective is too weak

$$M = \begin{pmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Scaled orthographic (6 degrees of freedom): – when very far away

$$M = \begin{pmatrix} r_{11} & r_{12} & r_{13} & \Delta X \\ r_{21} & r_{22} & r_{23} & \Delta Y \\ 0 & 0 & 0 & Z_0 \end{pmatrix}$$

Note: All these are just approximations that assume a pin-hole camera. The real world is always more complex.

Radial distortion

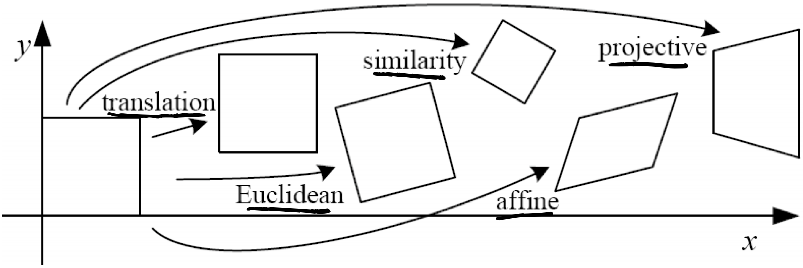
Due to the thickness of the lens, straight lines are no longer straight.

It is common to first apply an image transformation to straighten them out, then use the corrected images instead. An easiest model:

$$\begin{cases} x_c = x(1 + \kappa_1 r^2 + \kappa_2 r^4) \\ y_c = y(1 + \kappa_1 r^2 + \kappa_2 r^4) \end{cases}$$

where $r^2 = x^2 + y^2$, but models can be much more complex.

Image transformations: mappings from 2D to 2D



Degrees of freedom:

- Translation: 2 dof
- Euclidean (rotation, translation): 3 dof
- Similarity (rotation, translation, scaling): 4 dof
- Affine (rotation, translation, scaling, shear): 6 dof
- Projective (rotation, trans., scaling, shear, foreshortening): 8 dof

Image warping

Resample image $f(x, y)$ to get a new image $g(u, v)$, using a coordinate transformation: $u = u(x, y)$, $v = v(x, y)$.

Examples of transformations:

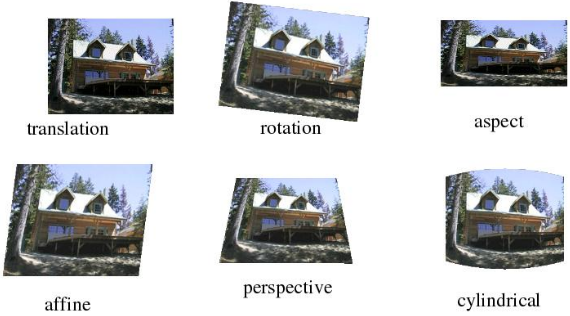
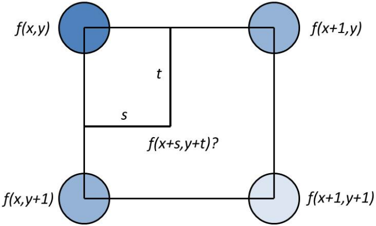


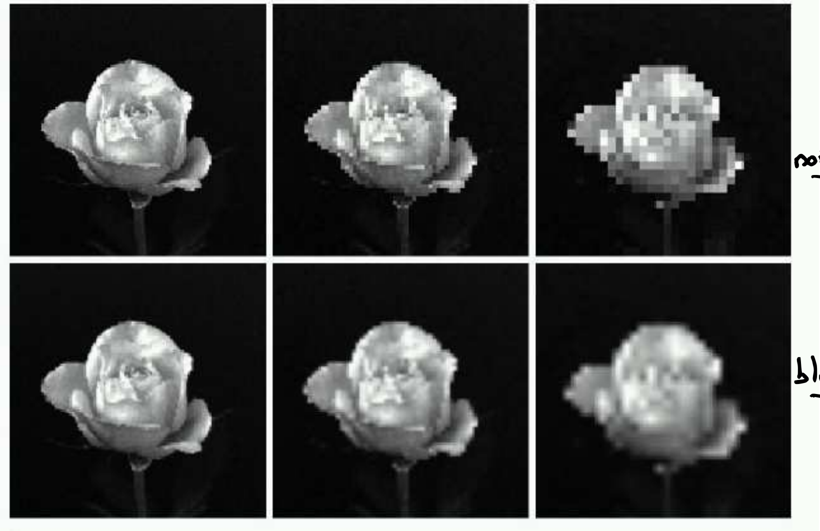
Image Warping

- For each grid point in (u, v) domain compute corresponding (x, y) values.
  Note: transformation is inverted to avoid holes in result.
- Create $g(u, v)$ by sampling from $f(x, y)$ either by:
  - Nearest neighbour look-up (noisy result)
  - Bilinear interpolation (blurry result)



$$
\begin{aligned}
f(x+s, y+t) &= (1-t)\cdot((1-s)\cdot f(x,y) + s\cdot f(x+1,y)) + \\
&+ t\cdot((1-s)\cdot f(x,y+1) + s\cdot f(x+1,y+1))
\end{aligned}
$$

Nearest Neighbor vs. Bilinear Interpolation



noisy

blurry

Reasoning about shape in binary images

- Images with two colours, black (0) and white (1 or 255).
  - Commonly referred to as 'background' and 'foreground'.
- Typically obtained from thresholding or image segmentation.

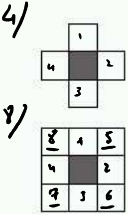
$$g(x,y) = \begin{cases} 1 & \text{if } f(x,y) > T \\ 0 & \text{otherwise} \end{cases}$$

Neighbourhood concepts

Many image processing operations work of neighbouring pixels. We need to define what it means that two pixels are neighbours.

Pixels are 4-neighbours if their distance is $D_4 = 1$

Pixels are 8-neighbours if their distance is $D_8 = 1$

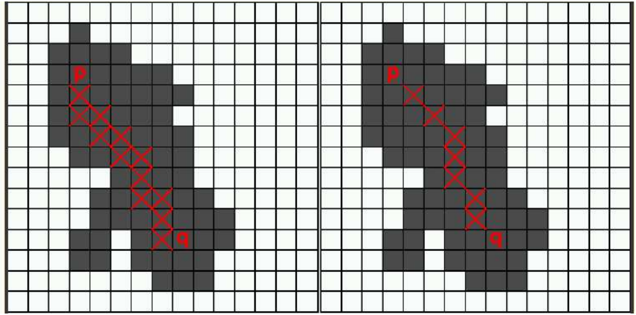


all 4-neighbours of center pixel

all 8-neighbours of center pixel

Connectivity

- Path: A path from p to q is a set of points $p_0 \dots p_n$, such that each point p_i is a neighbor of p_{i-1}.
- Connectivity: p is connected to q in S, if there is a path from p to q completely in S.

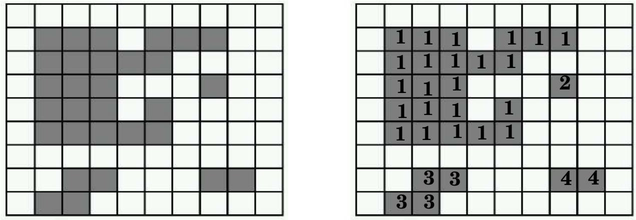


4-connectivity 8-connectivity

- For every p, the set of all points q connected to p is said to be its connected component.

Recursive procedure that scans entire image:

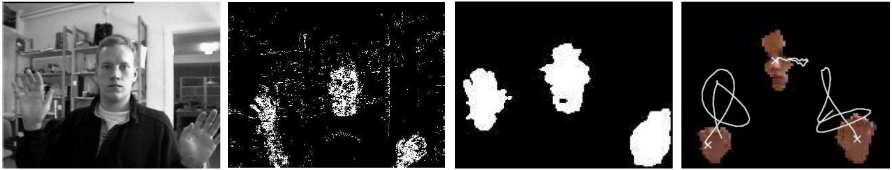
1. for each unlabeled foreground pixel, assign it a new label L
2. recursively assign label L to all neighboring foreground pixels
3. stop if there is no unlabeled foreground pixels



Connected component labeling

Regions (connected components) are often denoted by labels.

- statistics of regions (size, shape, gray-level statistics)
- size filtering (suppress objects of size $<$ threshold)

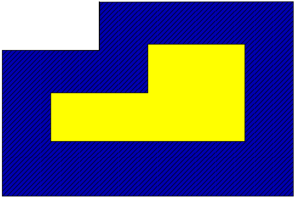


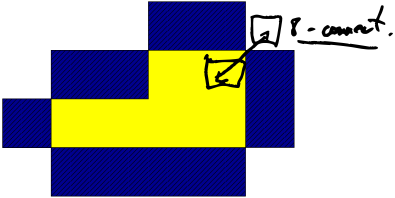
Connected components video example

Pixels are classified based on their colours, resulting in connected components. The largest such component is then tracked.

Duality of 4-connectivity and 8-connectivity

Outer boundary: background points with a neighbour on the object.





(left) based on 8-connectivity (right) based on 4-connectivity

- Jordan curve theorem (continuous case):
  Each closed curve divides plane into one region inside and one region outside.
- Note: Many region based methods, only store the boundary.

Distance measures

How to define distance between two points p and q?

Common distance measures:

- Euclidean distance $d(p,q) = \sqrt{(x-u)^2 + (y-v)^2}$
- City block distance $d(p,q) = \mid x-u \mid + \mid y-v \mid$
- Chessboard distance $d(p,q) = \max(\mid x-u \mid, \mid y-v \mid)$

All three measure satisfy metric axioms

- $d(p,q) \geq 0$
- $d(p,q) = d(q,p)$
- $d(p,r) \leq d(p,q) + d(q,r)$

Distance measures

Euclidean distance

$$\begin{matrix} \sqrt{8} & \sqrt{5} & 2 & \sqrt{5} & \sqrt{8} \\ \sqrt{5} & \sqrt{2} & 1 & \sqrt{2} & \sqrt{5} \\ 2 & 1 & 0 & 1 & 2 \\ \sqrt{5} & \sqrt{2} & 1 & \sqrt{2} & \sqrt{5} \\ \sqrt{8} & \sqrt{5} & 2 & \sqrt{5} & \sqrt{8} \end{matrix}$$

City block distance

$$\begin{matrix} 4 & 3 & 2 & 3 & 2 \\ 3 & 2 & 1 & 2 & 3 \\ 2 & 1 & 0 & 1 & 2 \\ 3 & 2 & 1 & 2 & 3 \\ 4 & 3 & 2 & 3 & 4 \end{matrix}$$

faster

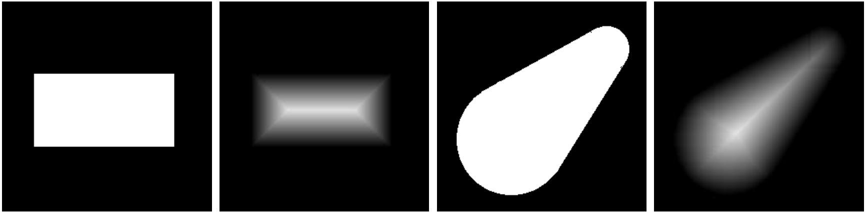
Chessboard distance

$$\begin{matrix} 2 & 2 & 2 & 2 & 2 \\ 2 & 1 & 1 & 1 & 2 \\ 2 & 1 & 0 & 1 & 2 \\ 2 & 1 & 1 & 1 & 2 \\ 2 & 2 & 2 & 2 & 2 \end{matrix}$$

The two last measures are usually faster, but equally useful.

Distance transform

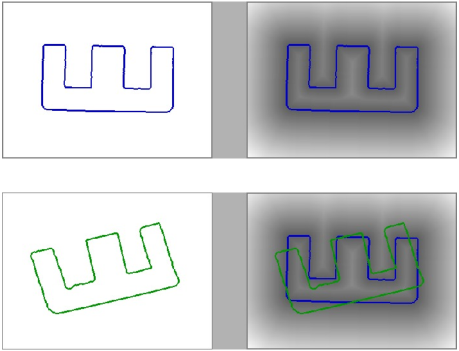
- The result is an image that shows the distance to the closest boundary from each point
- Useful for shape description, matching, skeletonization, etc



Example: Distance transform for aligning shapes

Goal: find the correct rotation and translation of a shape

- Create distance transform from model shape S_{model}.
- Extract new shape S_{image} from an image.
- Sum values in distance transform over edge points from S_{image}.
- Iteratively, rotate and translate S_{image} to minimize the sum.



Summary of good questions

- What is a pinhole camera model?
- What is the difference between intrinsic and extrinsic camera parameters?
- How does a 3D point get projected to a pixel with a perspective projection?
- What are homogeneous coordinates and what are they good for?
- What is a vanishing point and how do you find it?
- What is an affine camera model?
- What is a 4-neighbour and how is related to connectiveness?
- What kind of distance measures exist?

Readings

- Gonzalez and Woods: Chapters 2.4 - 2.5
- Szeliski: Chapters 2.1 and 3.6.1