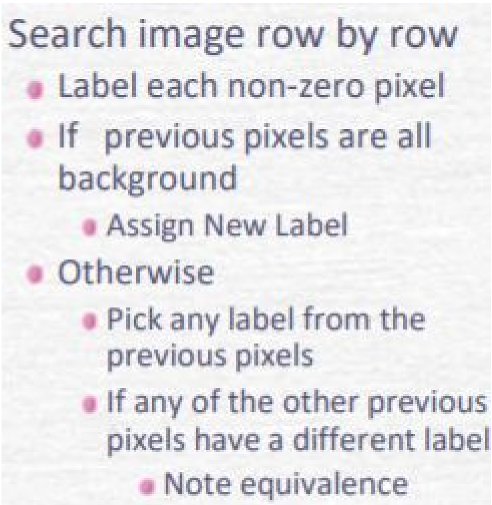
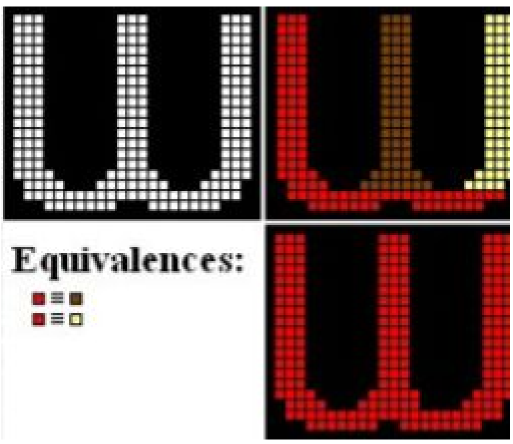
Connectivity

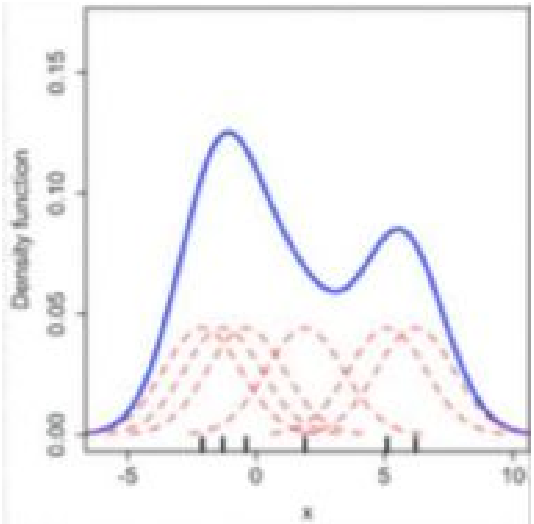
- **It is important to be able to reason about objects in a scene**
- **So we need to be able to tell which pixels are connected**
- Building regions using pixel adjacency
    - Objects
    - Backgrounds
    - Holes
- **4 adjacency vs 8 adjacency**
    - Need to use one of them
    - Using one or the other will give us bad results
        - Paradox: two crossing lines can appear continuous
    - **So a way around this is to alternate the adjacency**
        - 4-adjacency: Outer background
        - 8-adjacency: Outer object
        - 4-adjacency: Holes
        - 8-adjacency: Objects in holes
- **Connected Components Analysis**
    - Having alternated the adjacency to identify regions, we can use a single adjacency approach to label each non-zero pixel
    - 

    - **We then relabel all equivalent labels**
    - 

Segmentation

- Split an image into smaller parts

- Preferably corresponding to parts of particular objects
- Two main approaches:
  - **Region based**
  - **Edge based**
- Video segmentation
  - Breaking video into clips
  - Segmenting objects/regions consistently over time
- **Binary Regions**
  - Edge detection is easiest for binary images
- **K Means Clustering**
  - Covered in histogram notes
- **Watershed Segmentation**
  - In geology watersheds separate different catchment basins
  - **In computer vision we…**
    - Identify all regional minima
    - Label as different regions
    - Flood from minima extending the regions
    - Where two regions meet we have watershed lines
  - **Minimum of what?**
    - Greyscale values
    - Inverse of the gradient
    - Inverse of the chamfer distance
  - **Problems:**
    - Generally get too many regions
    - We can use pre-determined markers to identify objects and expand from there rather than from minima
- **Mean Shift Segmentation**
  - K-means clustering…
    - requires a pre-determined number of clusters to be known
    - Doesn't account for spatial location
  - **Mean shift doesn't have these restrictions**
  - **Kernel Density Estimation**
    - Given: a sparse dataset determine an estimate of density at each point
    - $$\hat{f}_h(x) = \frac{1}{n}\sum_{i=1}^{n} K_h(x - x_i) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$
      - Kernel Density K() function n samples x_i
      - h is the bandwidth
    - **This effectively smooths the data samples, and adds them all together**

        - 

    - If the dataset is multidimensional...
    - $$\hat{f}(\mathbf{x}) = \frac{1}{nh^d} \sum_{i=1}^{n} K\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right).$$
    - **There are many different kernels we can use**
    - They must integrate to one
    - We typically use…
        - **Uniform**
            - $$k_E(x) = \begin{cases} 1 - x & 0 \leq x \leq 1 \\ 0 & x > 1, \end{cases}$$
        - **Gaussian**
            - $$K_N(\mathbf{x}) = (2\pi)^{-d/2} \exp\left(-\frac{1}{2}\|\mathbf{x}\|^2\right)$$
- **So what is the mean shift algorithm?**
    - For each pixel…
        - Estimate the local kernel density and **the direction of the local increasing density (the mean shift vector)**
        - Shift the particle to the new mean
        - Re-compute until the location stabilizes
    - **Identify pixels that end up in the same location**
        - Mark these as members of a cluster
        - Determine the local mean of similar pixels
- **The points included in the kernel density estimate are limited based on distance and colour similarity to the current point**
    - $$K_{h_s,h_r}(\mathbf{x}) = \frac{C}{h_s^2 h_r^p} k\left(\left\|\frac{\mathbf{x}^s}{h_s}\right\|^2\right) k\left(\left\|\frac{\mathbf{x}^r}{h_r}\right\|^2\right)$$
    - This requires both a spatial kernel and a colour kernel
        - **Both can be gaussian**
        - Spatial kernel **limits/weights** the region to consider

- Colour kernel **limits/weights** the colour of the points to be included in the mean

- **The mean shift vector**

For radially symmetric kernels $K(\mathbf{x}) = c_{k,d}k(\|\mathbf{x}\|^2)$

So the KDE becomes $\hat{f}_{h,K}(\mathbf{x}) = \frac{c_{k,d}}{nh^d}\sum_{i=1}^{n} k\left(\left\|\frac{\mathbf{x}-\mathbf{x}_i}{h}\right\|^2\right)$ as $\hat{f}(\mathbf{x}) = \frac{1}{nh^d}\sum_{i=1}^{n} K\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right.$

We are interested in the rate of change...

$$\hat{\nabla} f_{h,K}(\mathbf{x}) \equiv \nabla \hat{f}_{h,K}(\mathbf{x}) = \frac{2c_{k,d}}{nh^{d+2}}\sum_{i=1}^{n}(\mathbf{x}-\mathbf{x}_i)k'\left(\left\|\frac{\mathbf{x}-\mathbf{x}_i}{h}\right\|^2\right)$$

Set $g(x) = -k'(x)$ so

$$= \frac{2c_{k,d}}{nh^{d+2}}\sum_{i=1}^{n}(\mathbf{x}_i-\mathbf{x})g\left(\left\|\frac{\mathbf{x}-\mathbf{x}_i}{h}\right\|^2\right)$$

$$= \frac{2c_{k,d}}{nh^{d+2}}\left[\sum_{i=1}^{n} g\left(\left\|\frac{\mathbf{x}-\mathbf{x}_i}{h}\right\|^2\right)\right]\left[\frac{\sum_{i=1}^{n}\mathbf{x}_i g\left(\left\|\frac{\mathbf{x}-\mathbf{x}_i}{h}\right\|^2\right)}{\sum_{i=1}^{n} g\left(\left\|\frac{\mathbf{x}-\mathbf{x}_i}{h}\right\|^2\right)} - \mathbf{x}\right.$$

Define the mean shift vector as

$$\mathbf{m}_{h,G}(\mathbf{x}) = \frac{\sum_{i=1}^{n}\mathbf{x}_i g\left(\left\|\frac{\mathbf{x}-\mathbf{x}_i}{h}\right\|^2\right)}{\sum_{i=1}^{n} g\left(\left\|\frac{\mathbf{x}-\mathbf{x}_i}{h}\right\|^2\right)} - \mathbf{x}$$

- **Pros of mean shift**
  - Do not need to know the number of clusters a priori
  - Provides segmentation which accounts for spatial and colour values
- **Cons of mean shift**
  - Selection of kernel widths can be very hard
  - It is slow, particularly when there are lots of clusters