From Lowe,
IJCV 2004

Can you find the locomotive? Can a computer program?

# Invariant feature detection*

- Consider representing an image of an object with a collection of descriptive local features

- Most useful if these occur in “edgy” areas.

- Common modern strategy is do detect somewhat robust “interest points” and form a descriptor for the local area.

- Example descriptor is a histogram of edge orientations (local texture).

*Good reference is Lowe, IJCV, 2004

# Distinctive Key-Points

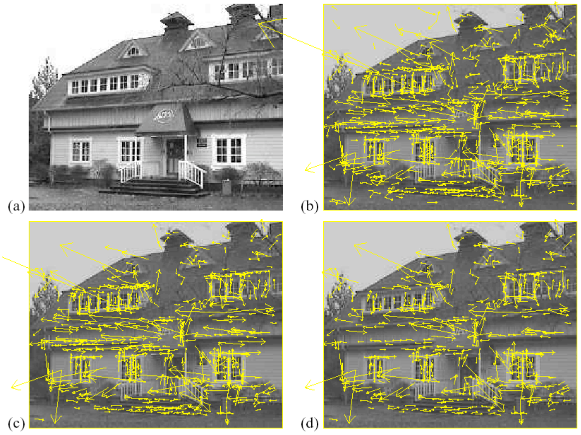
- Edges are interesting, but are they really distinctive?
  - Not for many applications because they do not have good localization

- More distinctive is a corner or a grid point

- Various strategies exist for finding “key-points” that are distinctive and localizable

- One idea is to look for edgy areas where one edge direction does NOT overly dominate the other
  - EG, a corner has both horizontal and vertical responses

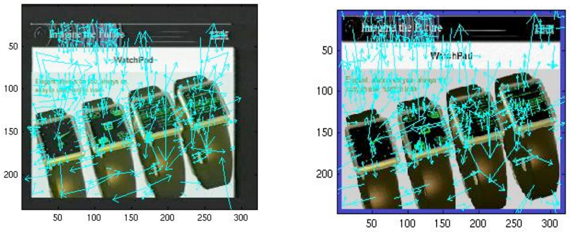
- Consider at different scales



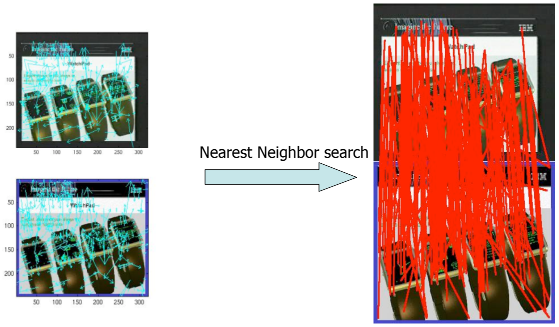
From Lowe, IJCV 2004

## Invariant feature detection

- To “find” the object, match the local features

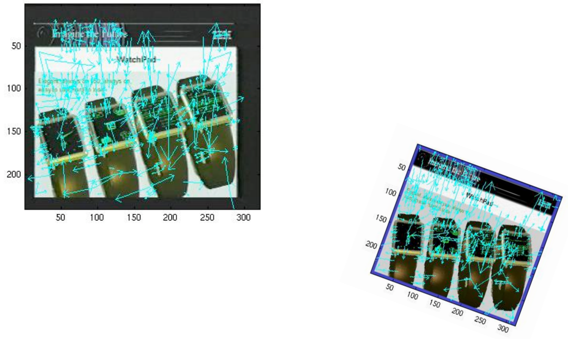


## Invariant feature detection



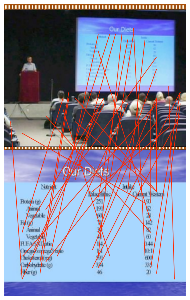
## Invariant feature detection

- Problems
  - Consistently determining which features goes with which
    - Covered later
  - Camera view changes
    - Approximately affine
    - Further approximated by scale and rotation

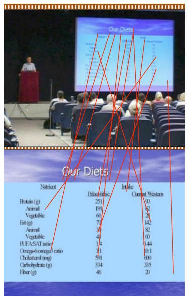


# Invariant feature detection

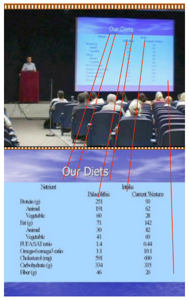
- Dealing with to camera view changes
  - Scaling and rotation can approximate camera view changes for small patches (locally planar)
  - Consider detector scale and direction (gradient)
  - This sets up a 2D coordinate system that is invariant to scale and rotation
  - One strategy is to make edge histogram grid with scaled bins and aligned with direction
  - Now, local feature description is invariant to scale and rotation.



*Initial matching*



*Constraining to correct part of image based on other information*



*After pruning outliers*

*(Covered later)*