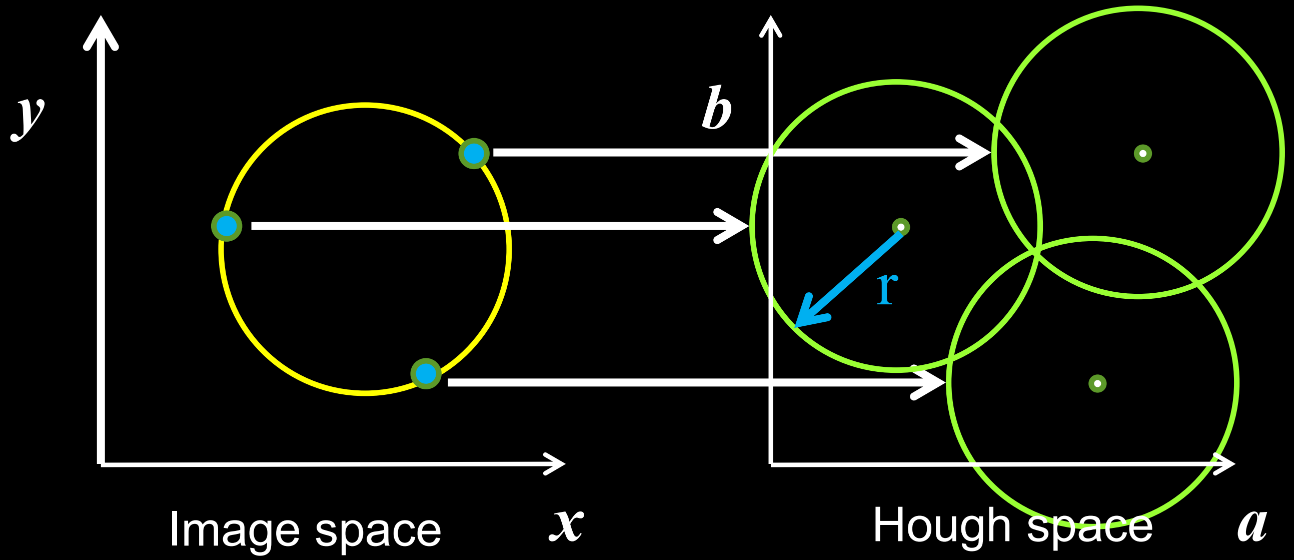
CS4495/6495
Introduction to Computer Vision

2B-L2 *Hough transform: Circles*

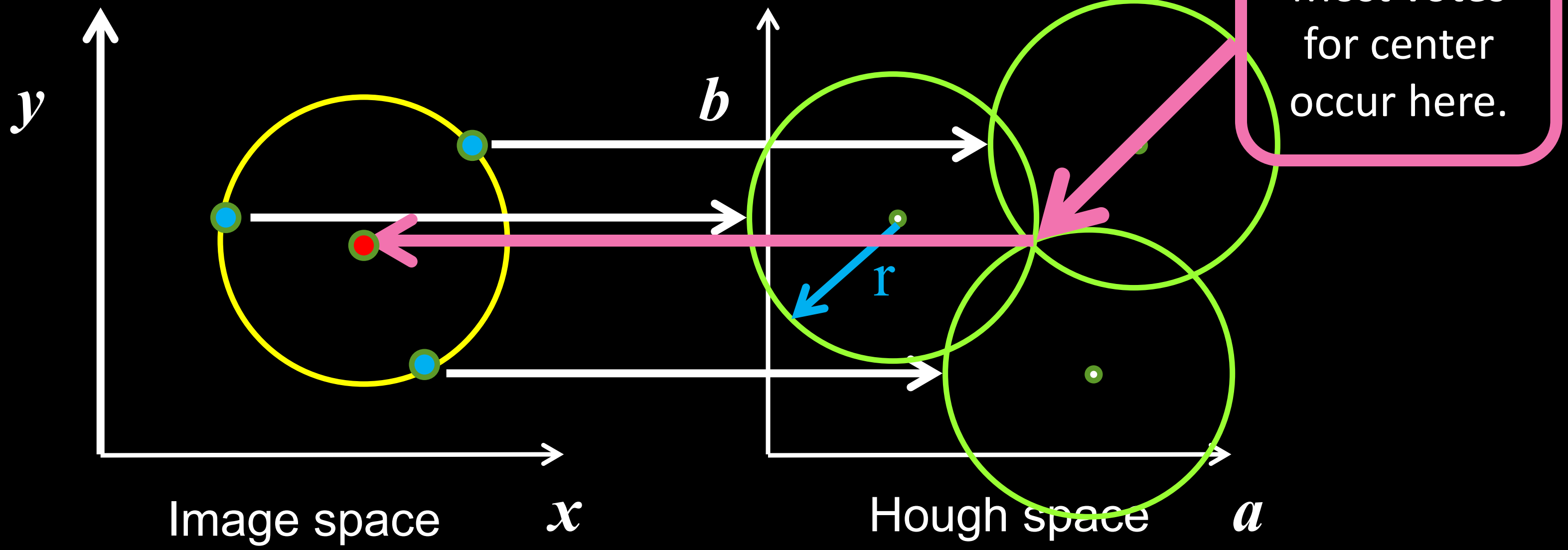
Hough transform for circles

- Circle: center (a,b) and radius r $(x_i - a)^2 + (y_i - b)^2 = r^2$
- For a fixed radius r, unknown gradient direction:



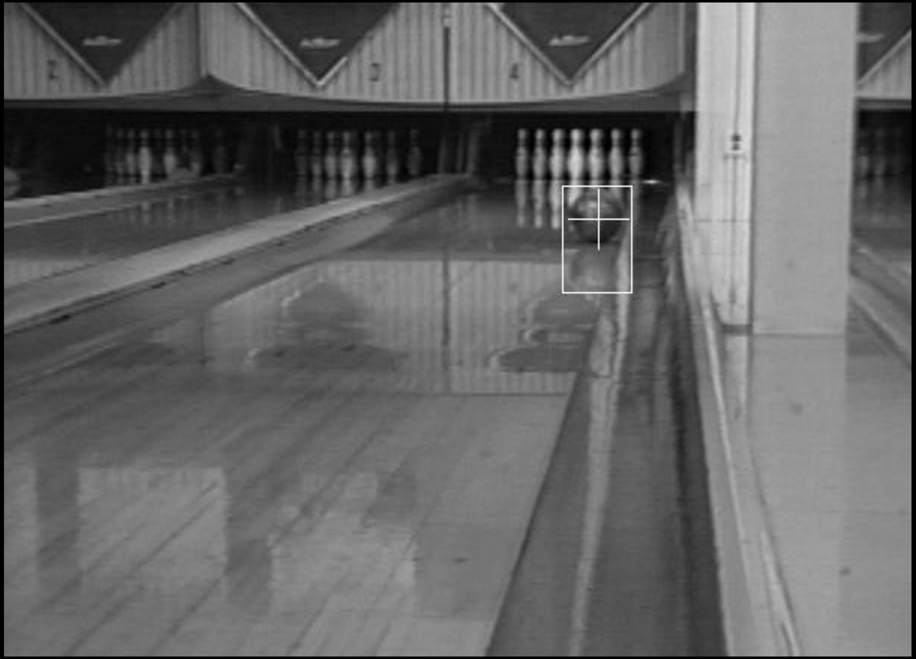
Hough transform for circles

- Circle: center (a,b) and radius r $(x_i - a)^2 + (y_i - b)^2 = r^2$
- For a fixed radius r, unknown gradient direction:



Most votes for center occur here.

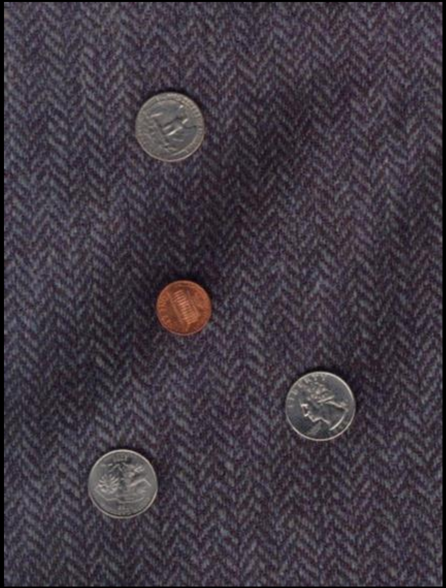
Example: detecting circles with Hough



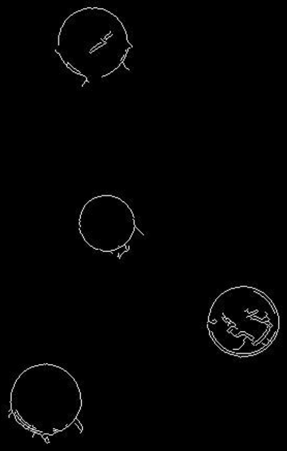
Crosshair indicates results of Hough transform; bounding box found via motion differencing.

Example: detecting circles with Hough

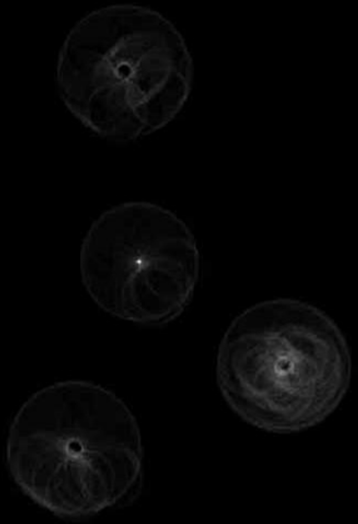
Original



Edges

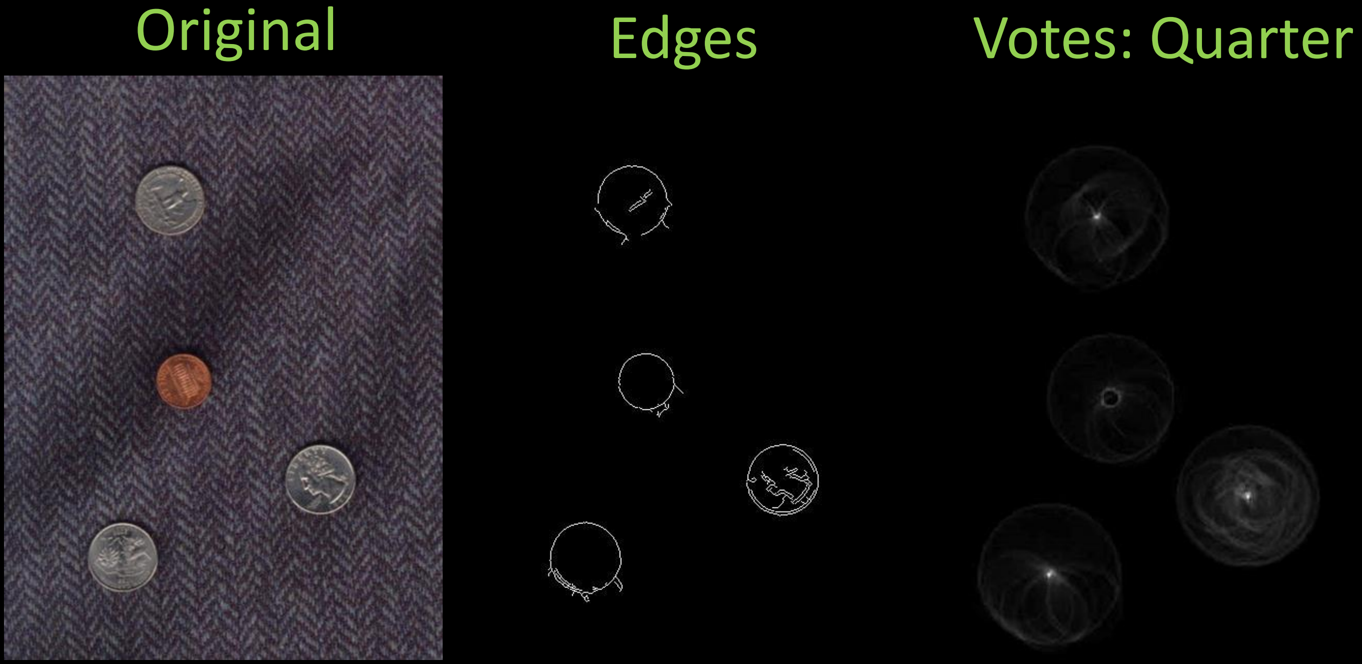


Votes: Penny



Note: a different Hough transform (with separate accumulators) was used for each circle radius (quarters vs. penny).

Example: detecting circles with Hough



Original

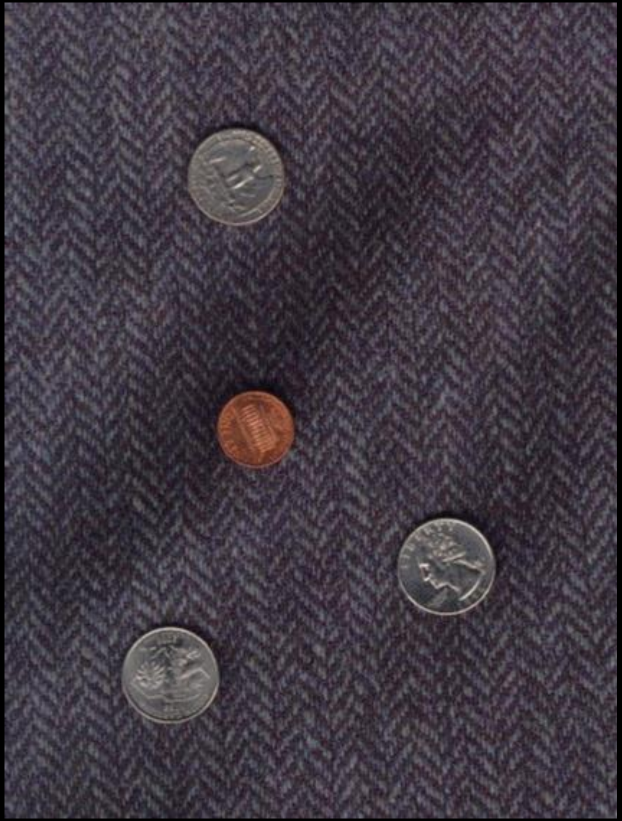
Edges

Votes: Quarter

Note: a different Hough transform (with separate accumulators) was used for each circle radius (quarters vs. penny).

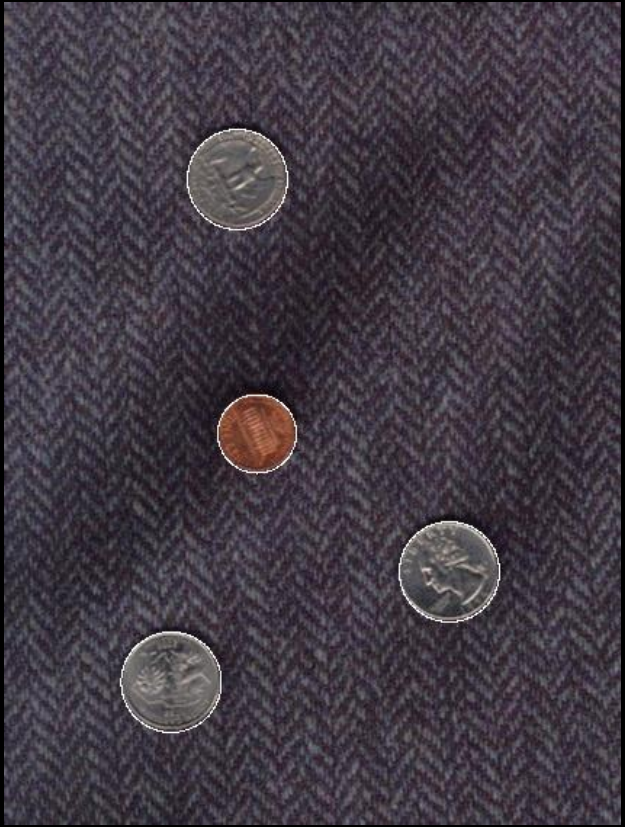
Example: detecting circles with Hough

Original



Example: detecting circles with Hough

Combined detections



Hough transform for circles

- Circle: center (a,b) and radius r $(x_i - a)^2 + (y_i - b)^2 = r^2$
- For ***unknown*** radius r, no gradient:

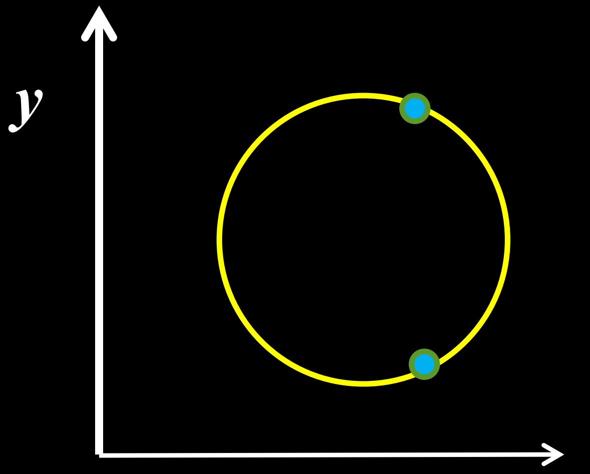
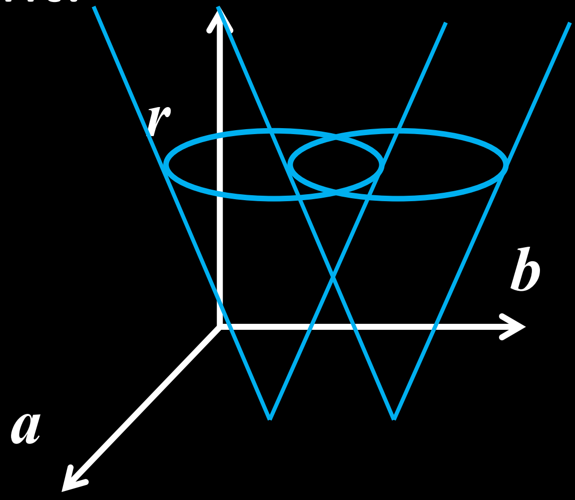


Image space



Hough space

Hough transform for circles

- Circle: center (a,b) and radius r $(x_i - a)^2 + (y_i - b)^2 = r^2$
- For ***unknown*** radius r, with ***gradient***:

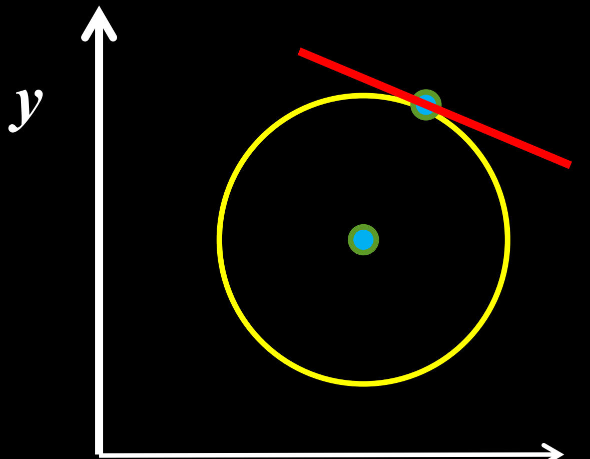
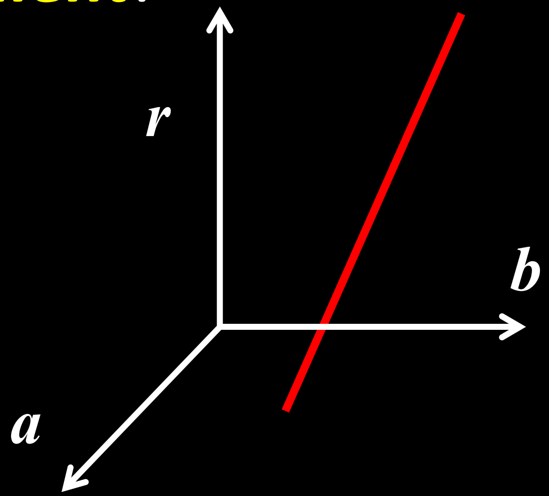


Image space



Hough space

Hough transform for circles

```
For every edge pixel (x,y) :
    For each possible radius value r:
        For each possible gradient direction θ:
            %% or use estimated gradient
                $a = x – r \cos(\theta)$
                $b = y + r \sin(\theta)$
                H[a,b,r] += 1
        end
    end
end
```

Voting: practical tips

- Minimize irrelevant tokens first (take edge points with significant gradient magnitude)
- Choose a good grid / discretization:
  - Too coarse: large votes obtained when too many different lines correspond to a single bucket
  - Too fine: miss lines because some points that are not exactly collinear cast votes for different buckets

Voting: practical tips

- Vote for neighboring bins (like smoothing in accumulator array)
- Utilize direction of edge to reduce free parameters by 1
- To read back which points voted for “winning” peaks, keep tags on the votes

Parameterized Hough transform: pros and cons

Pros

- All points are processed independently, so can cope with occlusion
- Some robustness to noise: noise points unlikely to contribute consistently to any single bin
- Can detect multiple instances of a model in a single pass

Parameterized Hough transform: pros and cons

Cons

- ***Complexity of search time increases exponentially with the number of model parameters***
- Non-target shapes can produce spurious peaks in parameter space
- Quantization: hard to pick a good grid size