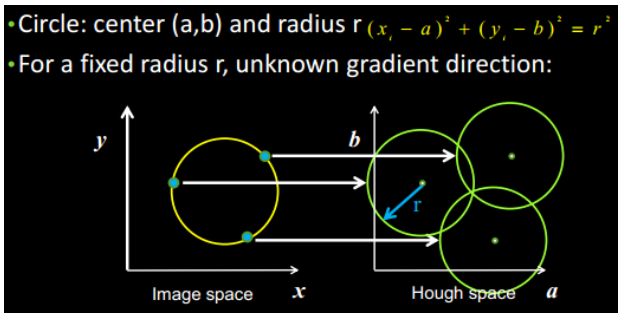
2B-L2 Hough transform - Circles

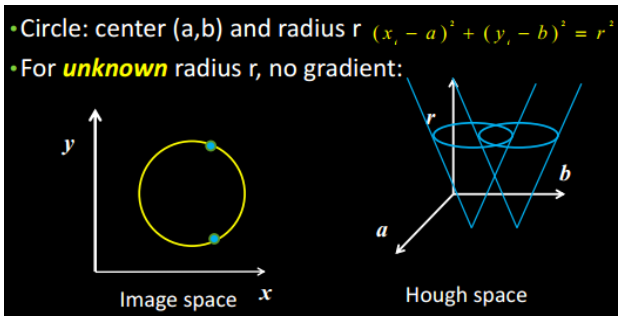
2017/11/13 02:59

1. Detecting Circles with Hough

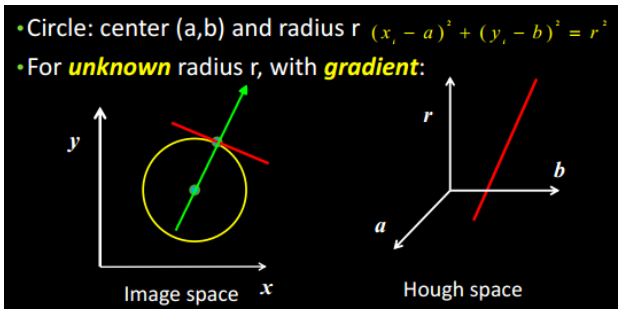


i. duality: one point in xy is a circle with r in ab; vise versa
ii. the intersection point of these three circles in ab defines a circle in xy

1. Hough Transform for Circles



i. when r is not know, the hough space become 3 dim, abr. still, to find the intersection. but compuationaly very expensive



1. the gradient of one point in the circle restrict the radius to be a line in the abr space

1. Algorithm 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(θ)
                b = y + r sin(θ)
                H[a,b,r] += 1
        end
    end
end
```

4. Voting Practical Tips
    a. **Minimize irrelevant tokens first (take edge points with significant gradient magnitude)**
    b. • Choose a good grid / discretization:
        i. • Too coarse: **large votes obtained when too many different lines correspond to a single bucket**
        ii. • Too fine: **miss lines because some points that are not exactly collinear cast votes for different buckets**
    c. Vote for neighboring bins (like smoothing in accumulator array)
    d. • **Utilize direction of edge to reduce free parameters by 1**
    e. • To read back which points voted for “winning” peaks, keep tags on the votes
5. 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

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

6. End
    a. it’s an old technique, but it still worth learning to extract structure with it