Lecture 10

Tracking using histograms

Wed. Oct. 7, 2020

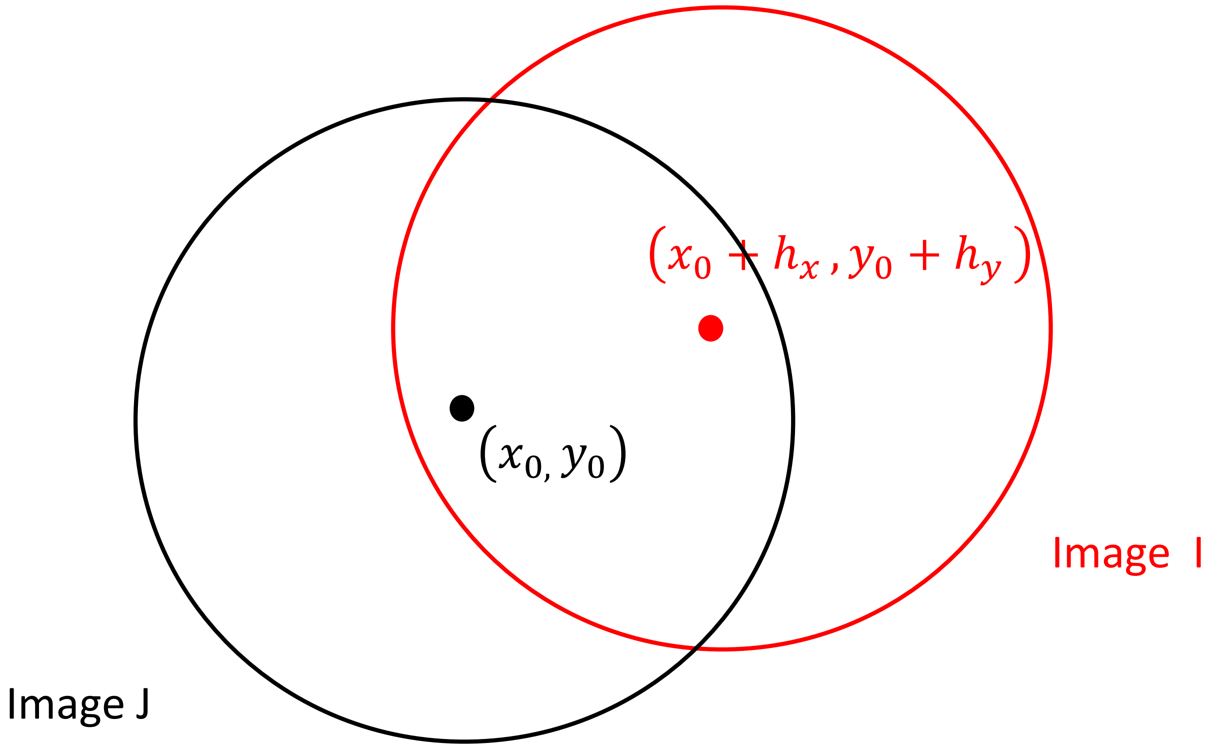
Reminder lecture recording

Recall the image registration problem (lecture 8):

For each $(x_{0,}\, y_0)$, find the $(h_{x,}\, h_y)$ that minimizes:

$$\sum_{(x,y)\in Ngd(x_0,y_0)} \{I(x+h_x, y+h_y) - J(x,y)\}^2$$

For each $(x_{0,} y_0)$, find the $(h_{x,} h_y)$ that minimizes the sum of squared differences of intensities:



Tracking

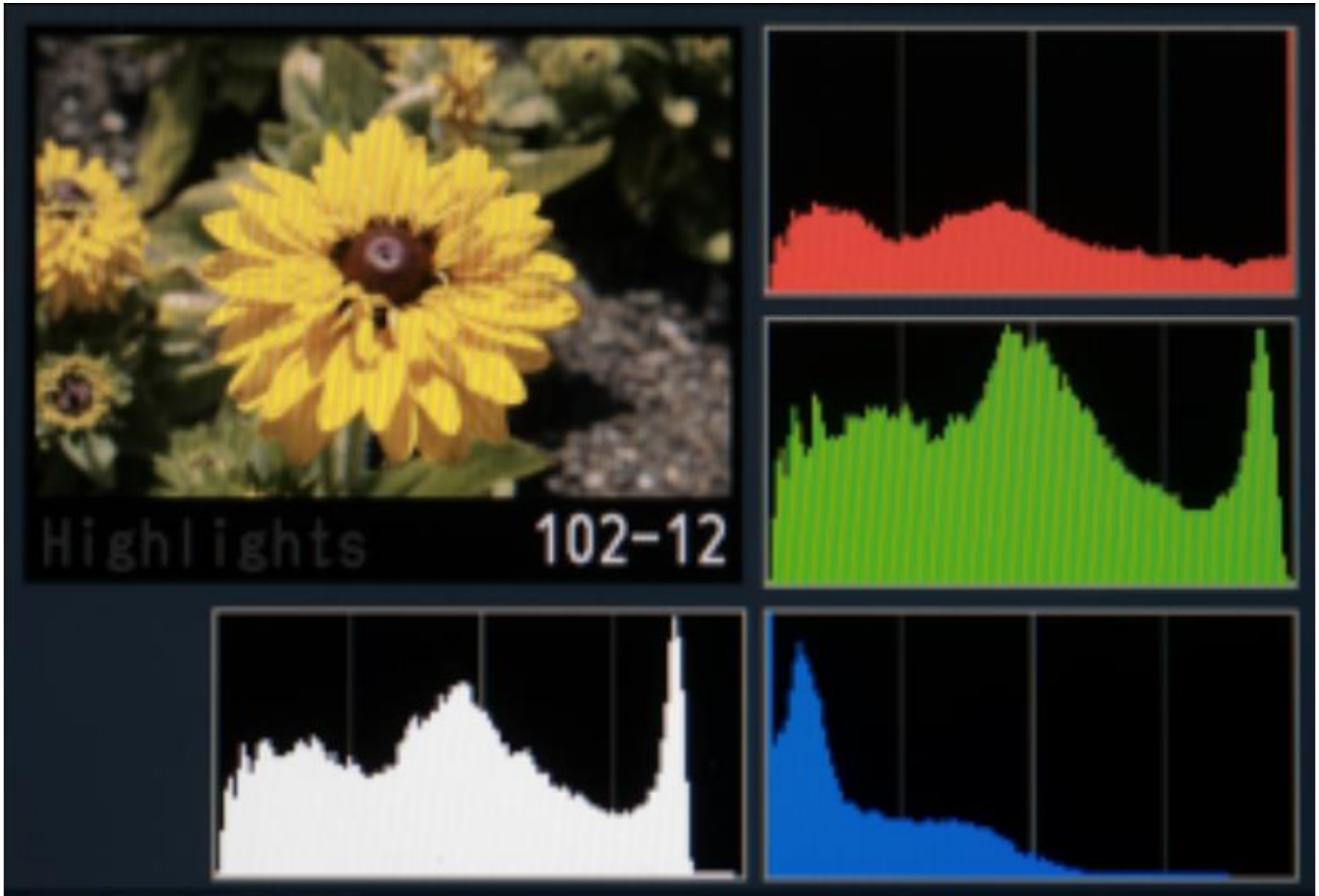
Perform frame-to-frame registration of a local patch: model how its translates and deforms over time.

LKT tracker follows the position of keypoint features (locally distinctive points) over multiple frames.

Registration-based tracking can fail when objects have moving parts.

Track this player

Today we'll look at a tracking approach that is based on RGB Histograms (recall lecture 2)

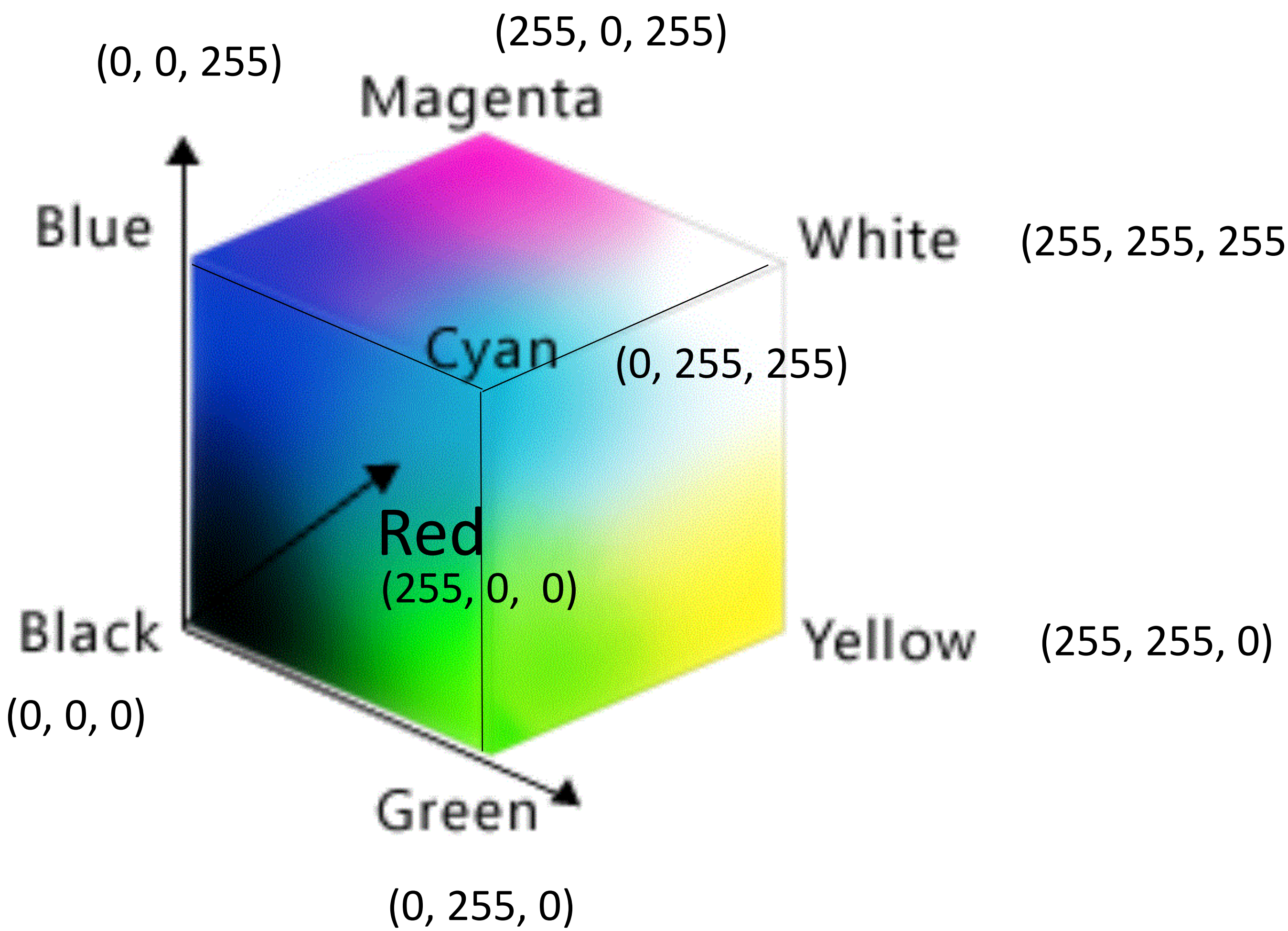


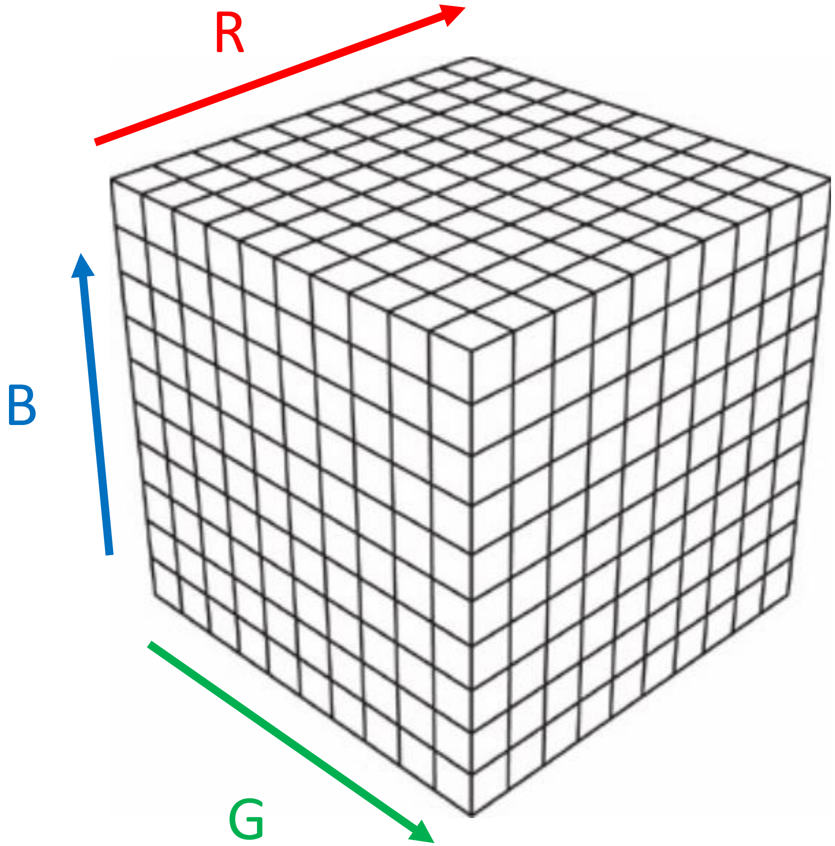
Histogram-based tracking

If you want to track a person over multiple frames of a video, but we don't care about exact position, it is often enough to use an RGB histogram.

How should we set up the problem ?

RGB values: 0 to 255 (8 bits)





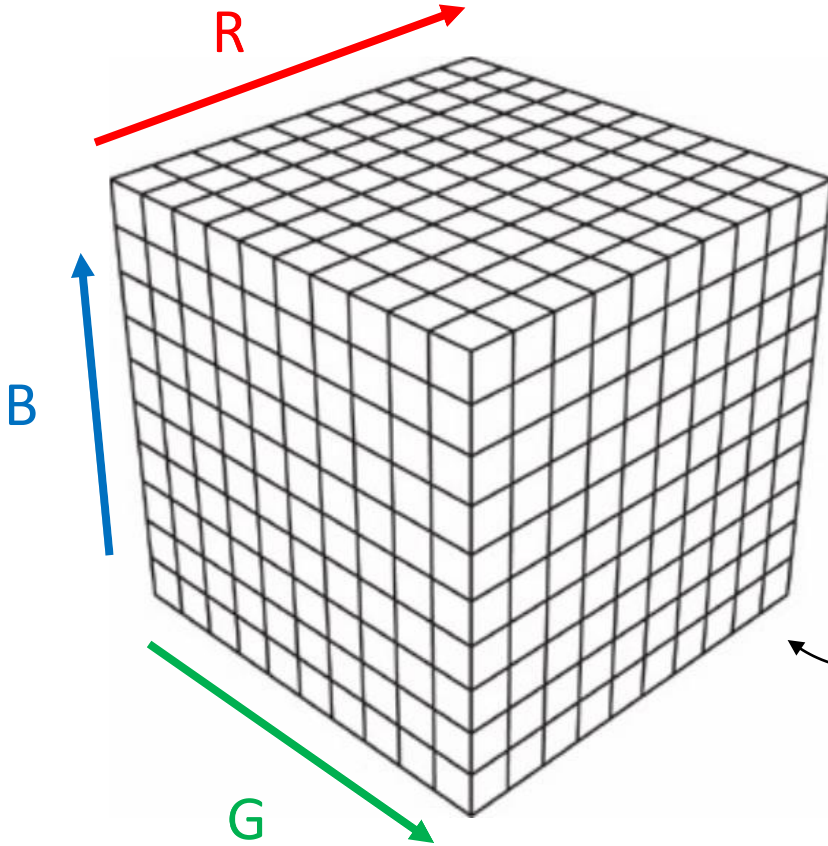
Suppose we partition each axis into 8 levels.
This would give 512 = 8*8*8 bins.
(Sorry the picture is 10*10*10.)

Each bin represents a range of 32 (256/8) levels of each R, G, B.

e.g.

R in [32,63], G in [224, 255], B in [96 ,127].

We partition each axis into 8 levels.
This would give 512 = 8*8*8 bins.



We index the bins by variable u.

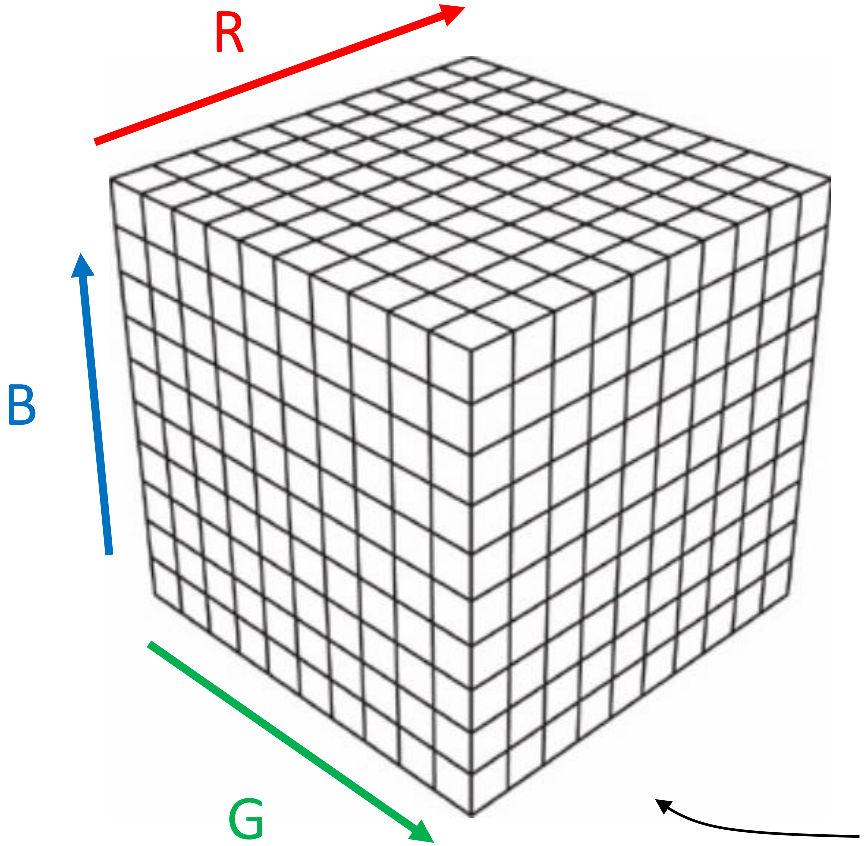
Suppose we have an RGB image $I(\boldsymbol{x})$ where $\boldsymbol{x}$ is a pixel.

$$u = bin\big(I(\boldsymbol{x})\big)$$

This maps pixel $\boldsymbol{x}$ to bin u.

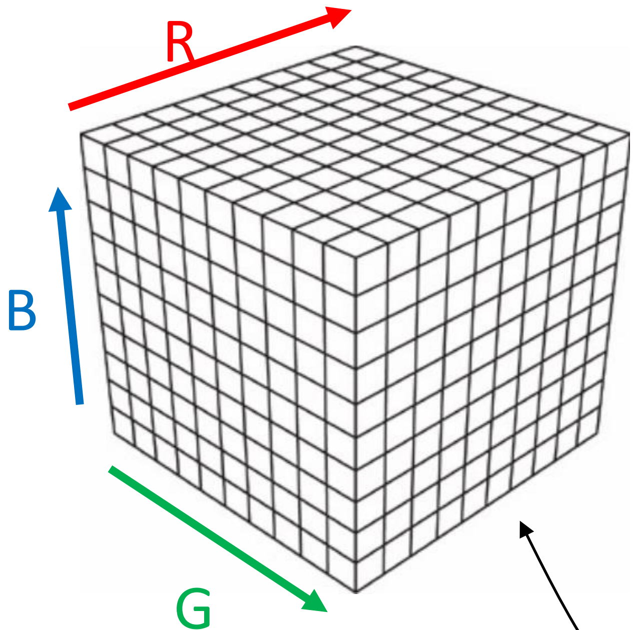
$\boldsymbol{x} \rightarrow I(\boldsymbol{x}) \rightarrow bin(I(\boldsymbol{x}))$

Define a *histogram* that counts the number of image pixels that map to each bin in RGB space.



Notation below is unconventional but it does work: it expresses the above definition.

$$hist(u) \equiv \sum_{\boldsymbol{x}} \delta(u - bin(I(\boldsymbol{x})))$$



We will define histograms over a ***region of interest*** **(ROI)**, centered at some pixel position $\boldsymbol{y}$.

Notation: $\boldsymbol{x}$ and $\boldsymbol{y}$ here refer to different positions.

How many pixels have RGB value in bin u ?

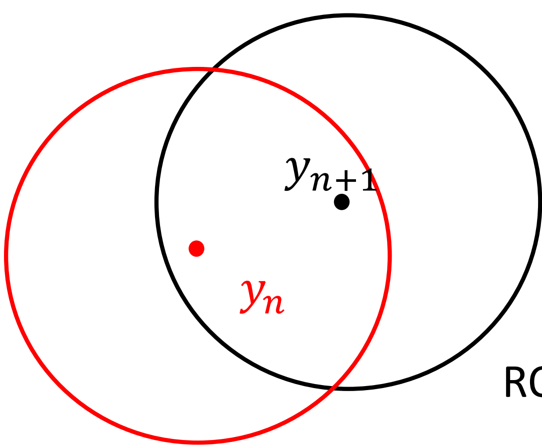
$$hist(u;\boldsymbol{y}) \equiv \sum_{\boldsymbol{x} \in \boldsymbol{ROI}(\boldsymbol{y})} \delta(u - bin(I(\boldsymbol{x})))$$

We want to track the object over many frames.

Let the image frames be $I_1, I_2, \ldots, I_n, I_{n+1}, \ldots\ldots$

Suppose we initialize a ROI at position y_1 in image 1

Given position y_n, estimate position y_{n+1}.

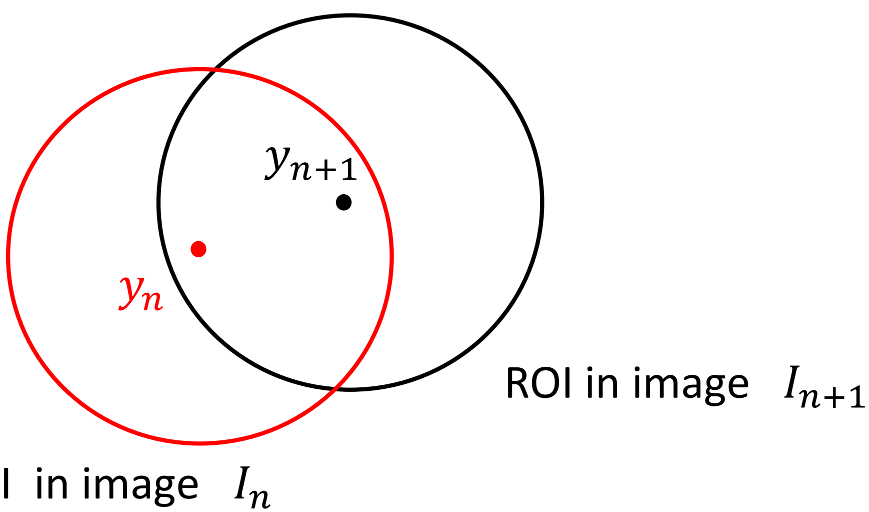


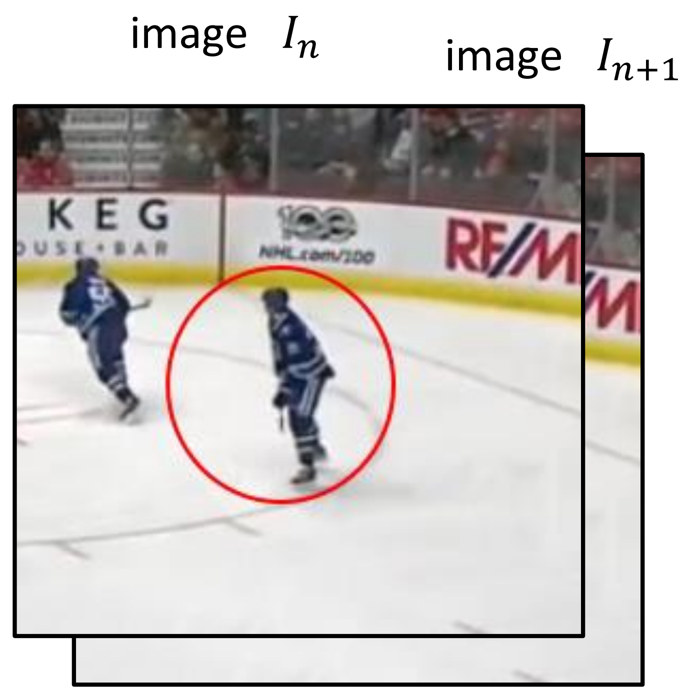
ROI in image I_{n+1}

ROI in image I_n

Given position y_n centered at ROI in frame I_n ,
find the nearby position y_{n+1} in frame I_{n+1} that
maximizes the similarity of the histograms.

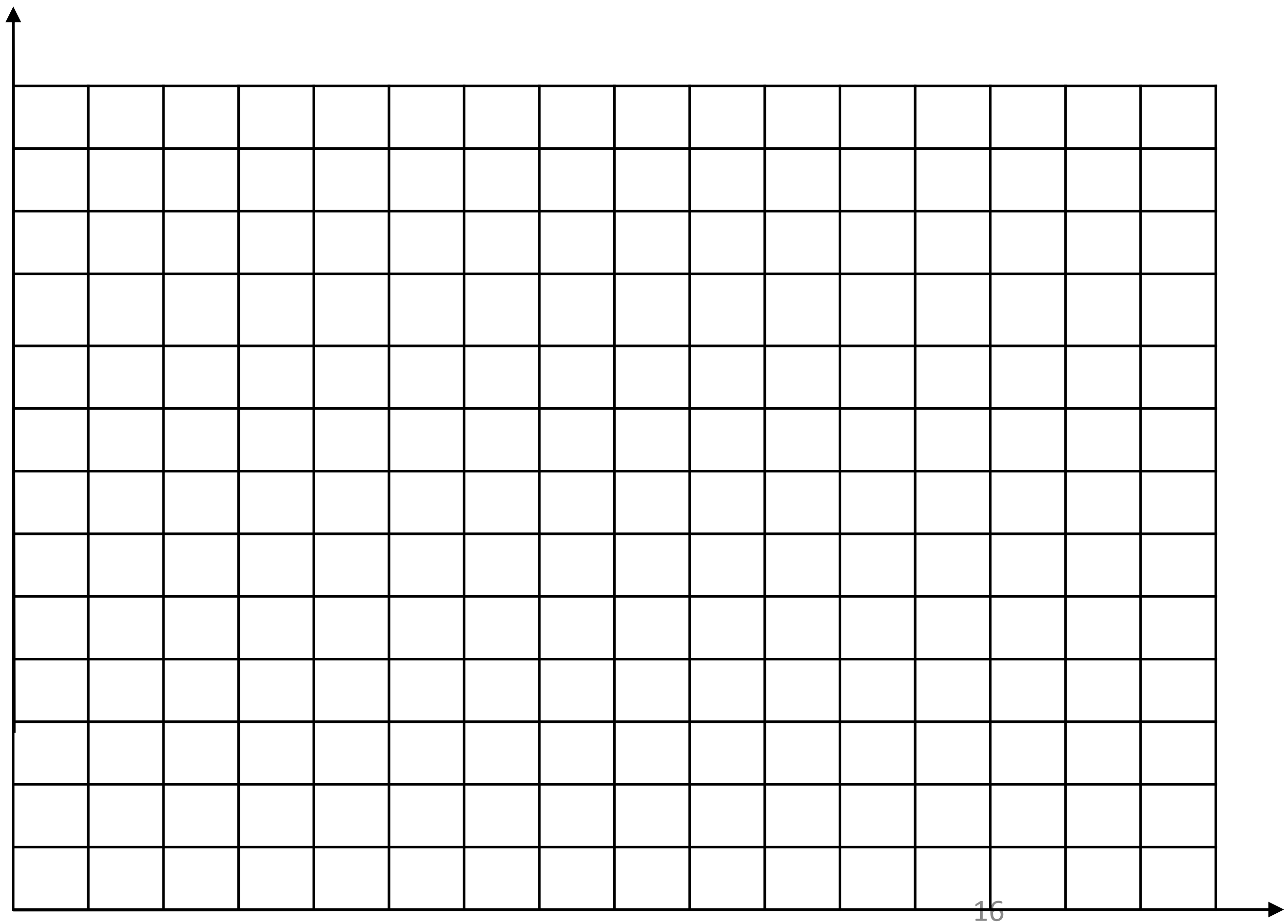
How do histograms vary with position y ?





Think of a 1D image. Let x be pixel positions. Let RGB bins be u.

How do we define histograms ?



For each image position, there is one RGB value, and so there is one bin value u. If $u = bin\big(I(x)\big)$, then we put a value 1 in that bin.

Note: each column sums to 1, but rows typically do not sum to 1.

u

			1												
		1						1	1	1					
1	1										1				
							1								
				1	1	1									
													1		
												1			
														1	1

Image position x

For each x, we have a very simple histogram:

$$hist(u; x) = \delta\left(u - bin\left(I(x)\right)\right).$$

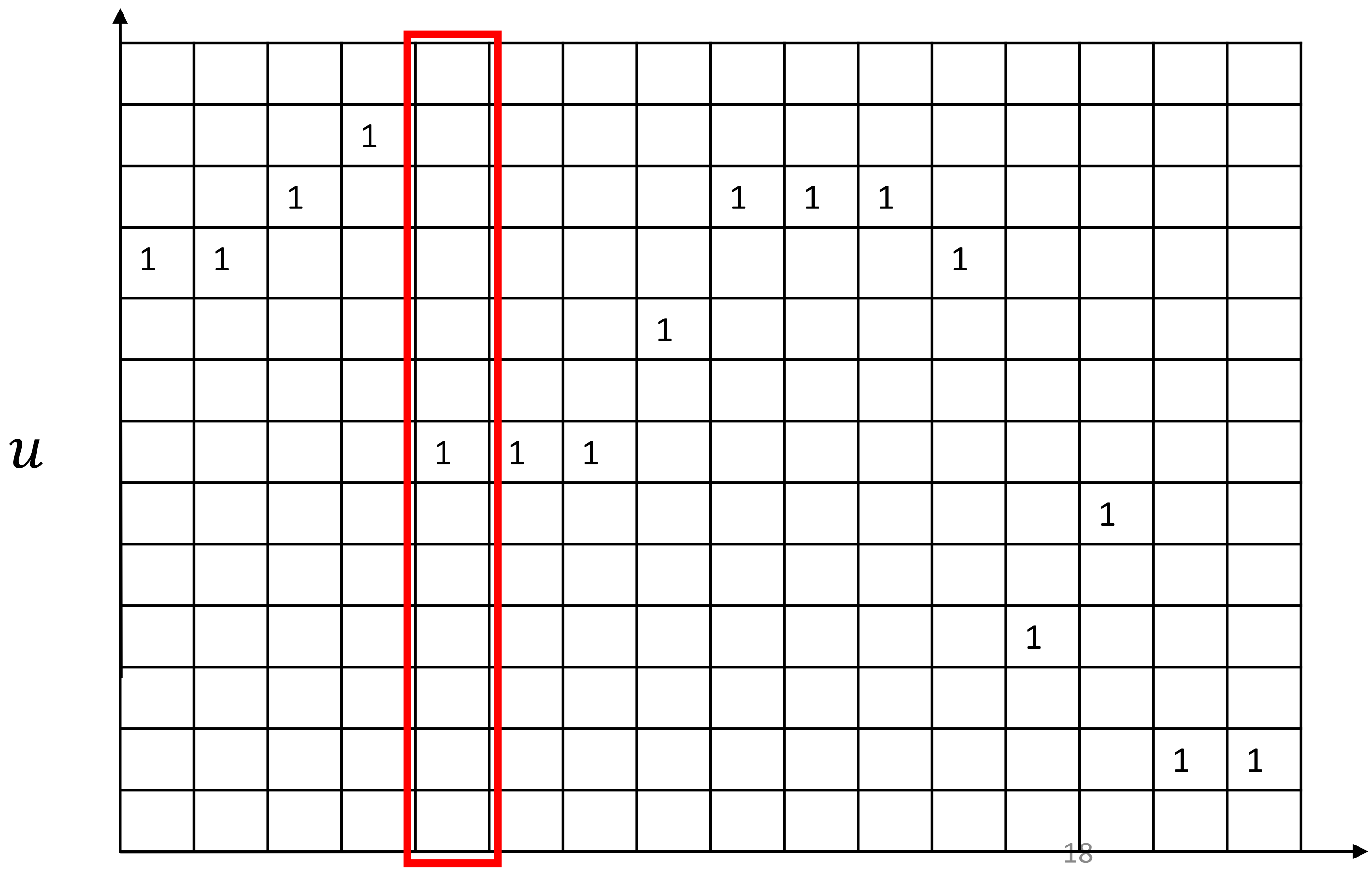


Image position x

Histogram for whole image.

u

			1												
		1						1	1	1					
1	1										1				
							1								
				1	1	1									
													1		
												1			
														1	1

0
1
4
3
1
0
3
1
0
1
0
2
0

Image position x

How do we define histograms on ROI's ?

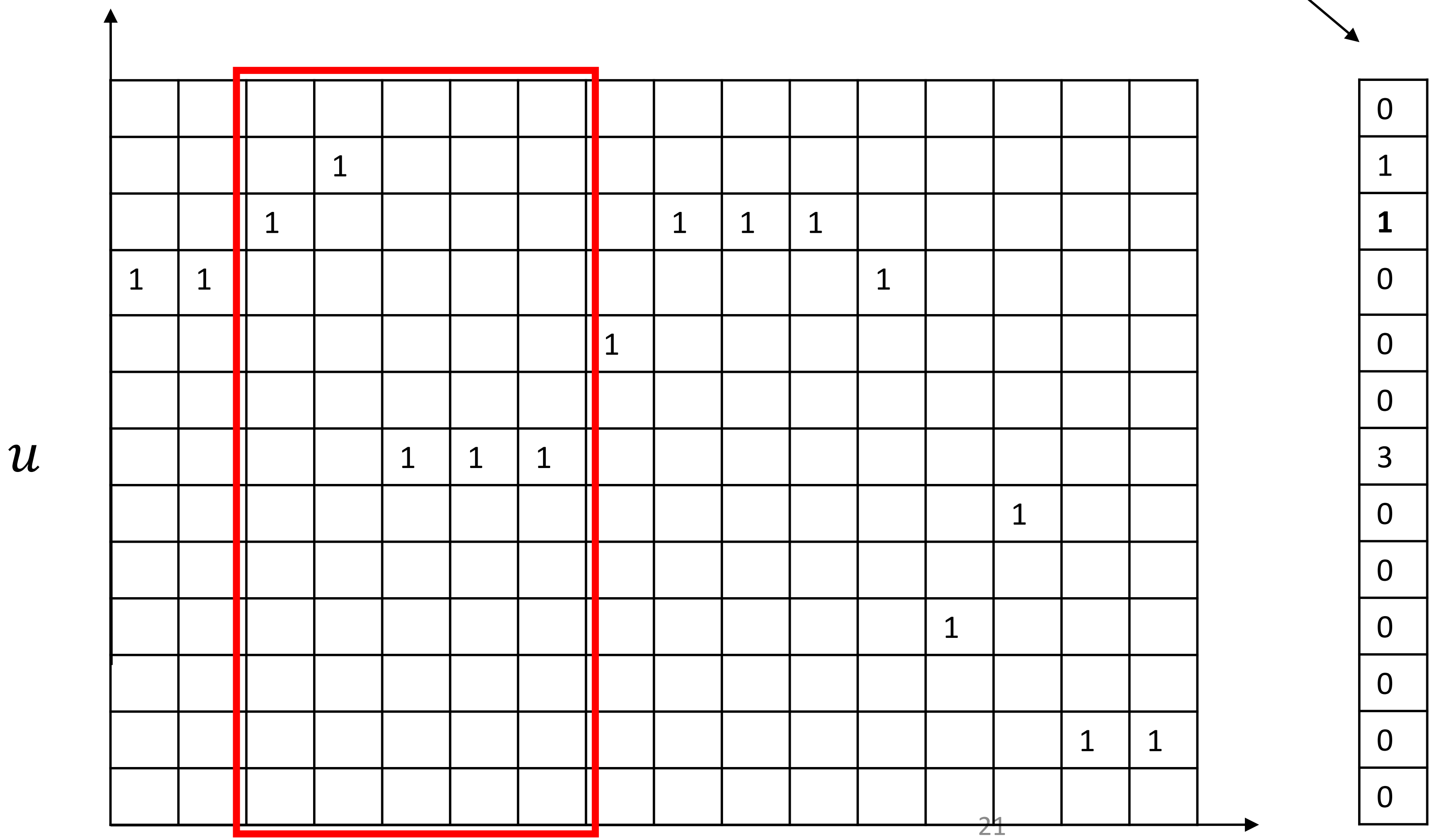
bin u

			1												
		1						1	1	1					
1	1										1				
							1								
				1	1	1									
													1		
												1			
														1	1

Image position x

$$hist(u;\mathbf{y}) = \sum_{\mathbf{x}\in ROI(\mathbf{y})} \delta(u - bin(I(\mathbf{x})))$$

Histogram for the ROI centered at location y.

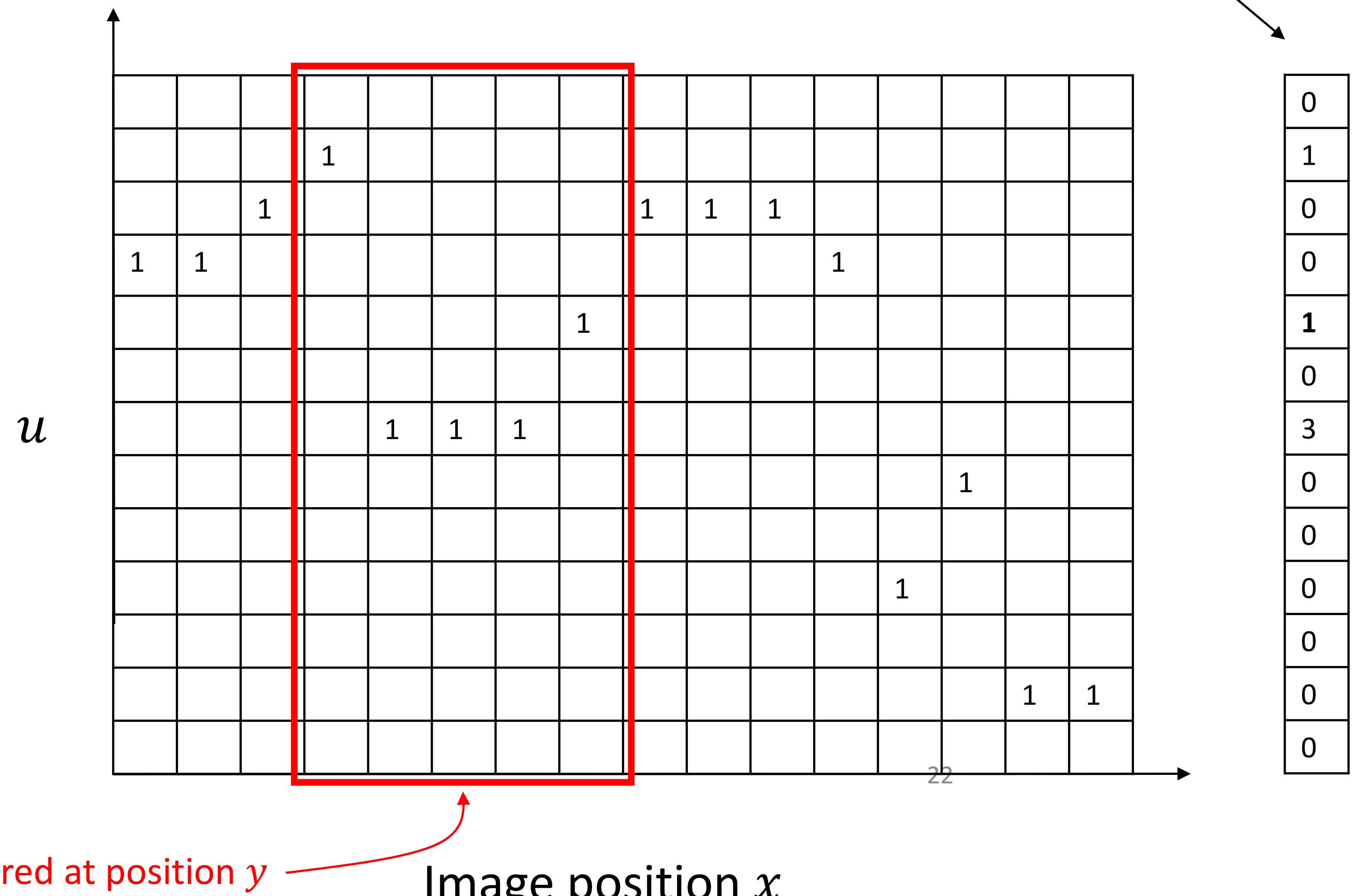


ROI centered at position y

Image position x

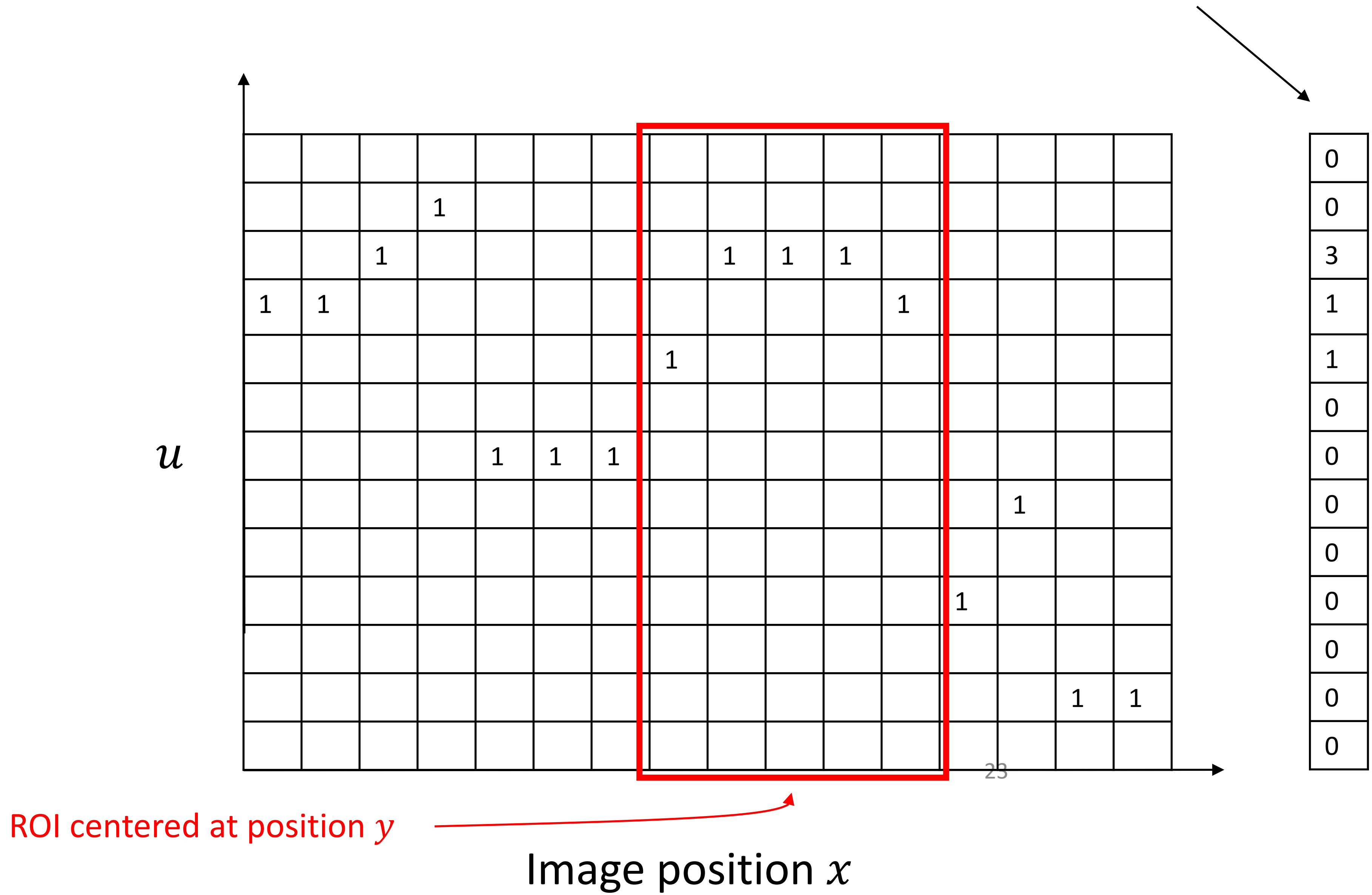
$$hist(u;\mathbf{y}) = \sum_{\mathbf{x}\in ROI(\mathbf{y})} \delta(u - bin(I(\mathbf{x})))$$

Histogram for the ROI centered at location y.



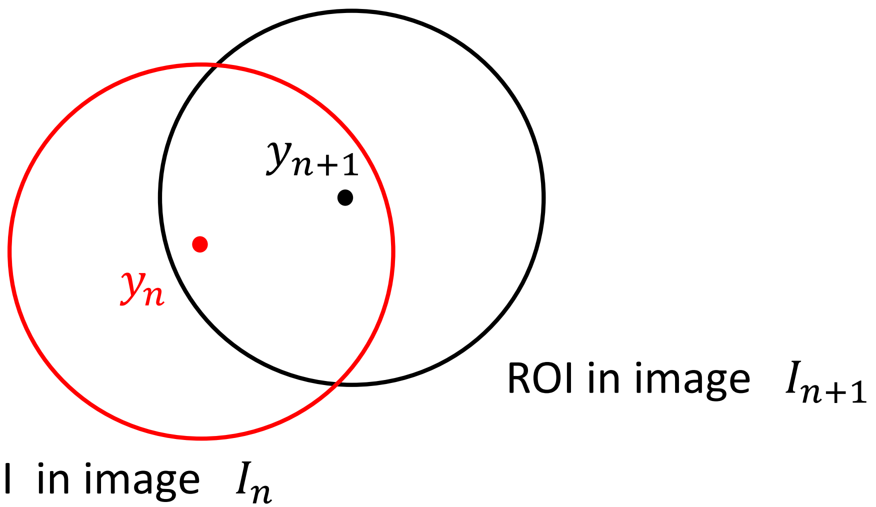
$$hist(u;\mathbf{y}) = \sum_{\mathbf{x}\in ROI(\mathbf{y})} \delta(u - bin(I(\mathbf{x})))$$

Histogram for the ROI centered at location y.



Given position y_n centered at ROI in frame I_n ,
find the nearby position y_{n+1} in frame I_{n+1} that *maximizes the similarity* of the ROI histograms.

How do we define similarity of histograms ?



Brute force tracking with ROI histogram comparison.

Histogram for ROI centered at $\boldsymbol{y}_n$ in frame I_n.

$$hist_n(u; \mathbf{y}_n) = \sum_{\mathbf{x} \in ROI(\mathbf{y}_n)} \delta(u - bin(I(\mathbf{x})))$$

Histogram for ROI centered at $\boldsymbol{y}$ in frame I_{n+1}.

$$hist_{n+1}(u; \mathbf{y}) = \sum_{\mathbf{x} \in ROI(\mathbf{y})} \delta(u - bin(I_{n+1}(\mathbf{x})))$$

Let y_{n+1} be the position y in frame I_{n+1} that *maximizes the similarity* of the histograms.

Histogram for ROI centered at y_n in frame I_n.

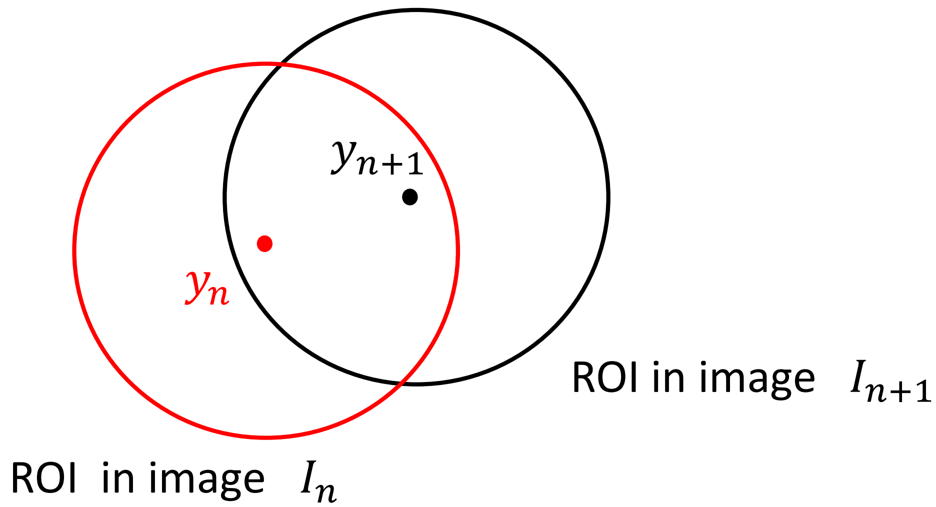
$$hist_n(u; \mathbf{y}_n) = \sum_{\mathbf{x} \in ROI(\mathbf{y}_n)} \delta(u - bin(I(\mathbf{x})))$$

Histogram for ROI centered at y in frame I_{n+1}.

$$hist_{n+1}(u; \mathbf{y}) = \sum_{\mathbf{x} \in ROI(\mathbf{y})} \delta(u - bin(I_{n+1}(\mathbf{x})))$$

Let y_{n+1} be the position y in frame I_{n+1} that e.g. *minimizes the sum of bin-wise differences of the histograms:*

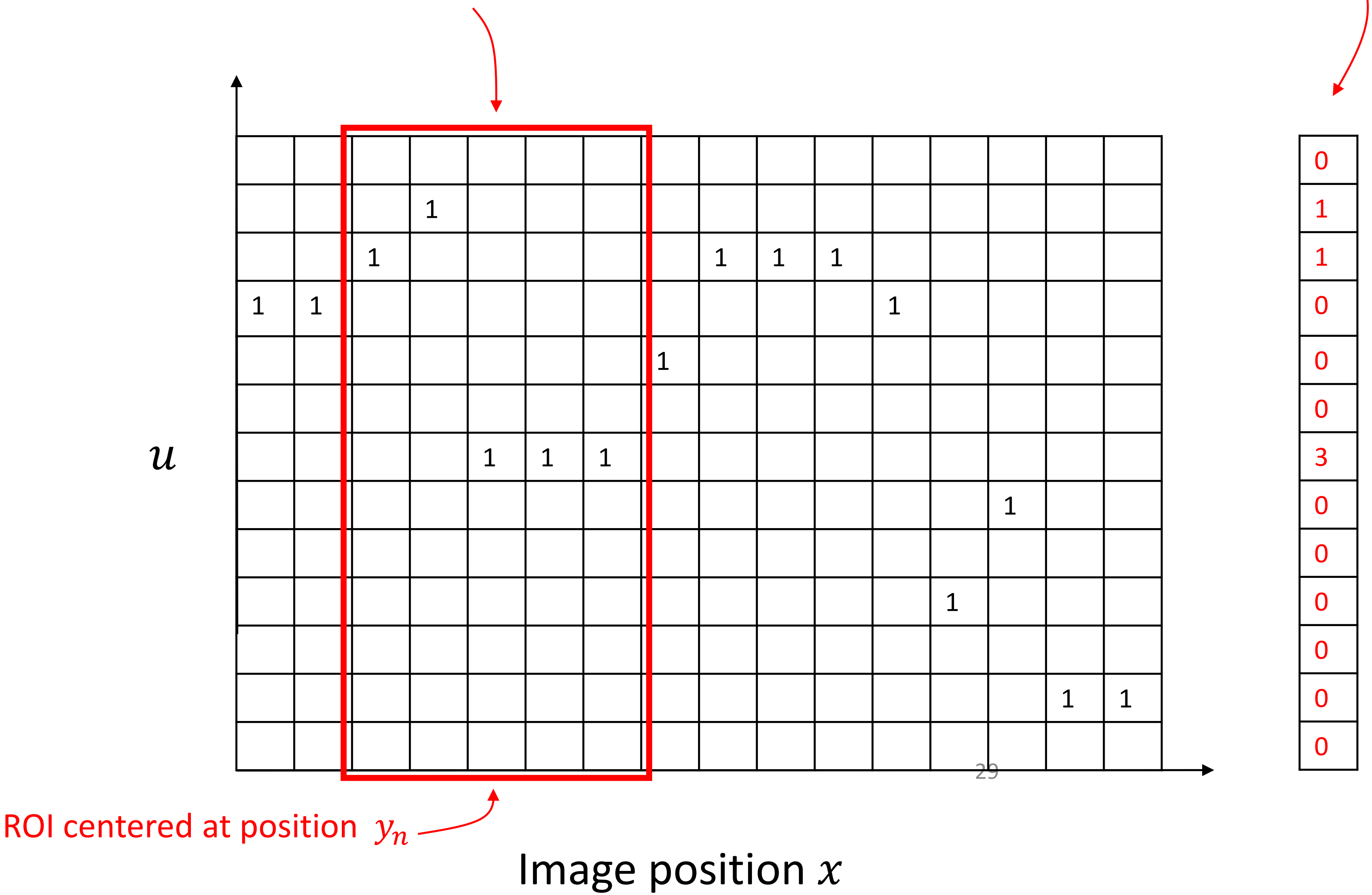
$$\sum_u |hist_{n+1}(u; \mathbf{y}) - hist_n(u; \mathbf{y}_n)|$$



Let’s try to visualize the problem…

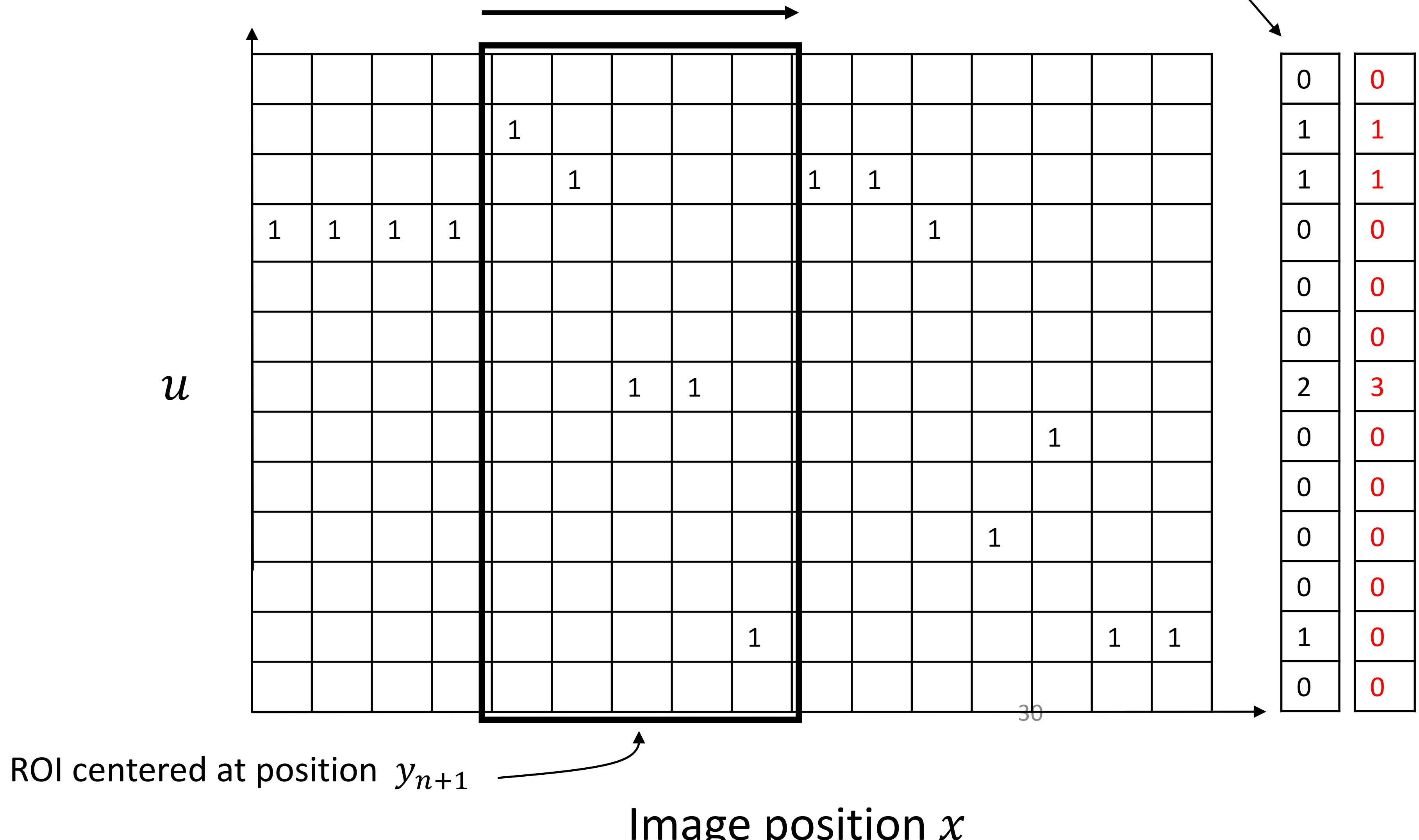
Suppose we have an ROI in image I_n centered at position y_n.

Histogram for the ROI in image I_n centered at y_n.



Find the position y_{n+1} in image I_{n+1} whose ROI histogram (right) is most similar to histogram from previous slide.

Histogram for the ROI in image I_{n+1} centered at y_{n+1}



Two problems with brute force tracking with ROI histogram comparison.

1) We should give more weight to the pixels near the center of the ROI.

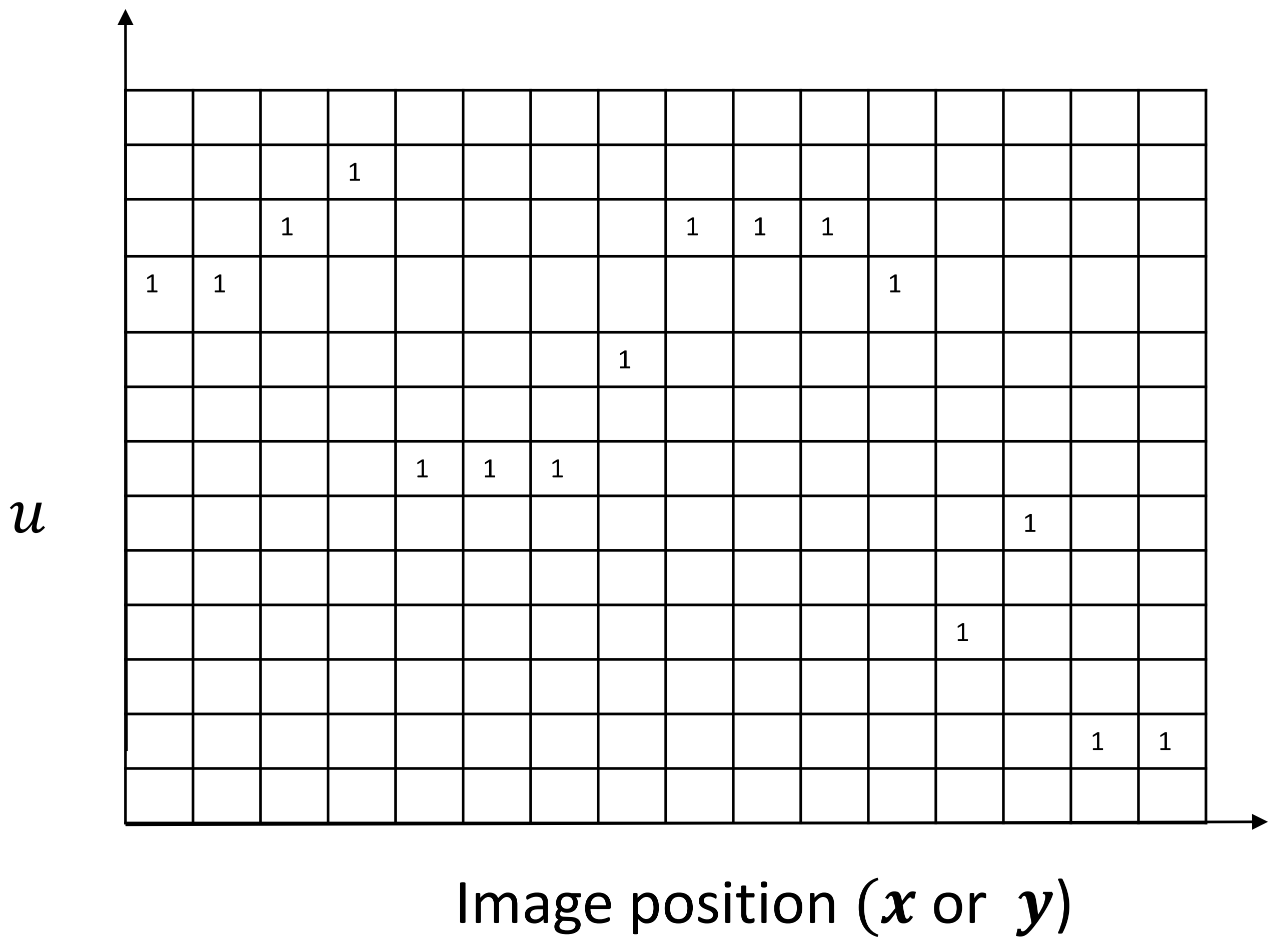
How to do so ?

1) Bruce force search is inefficient.

.

Define a symmetric weighting function $W(\boldsymbol{x})$, typically a Gaussian. Convolve each row u with $W(\boldsymbol{x})$:

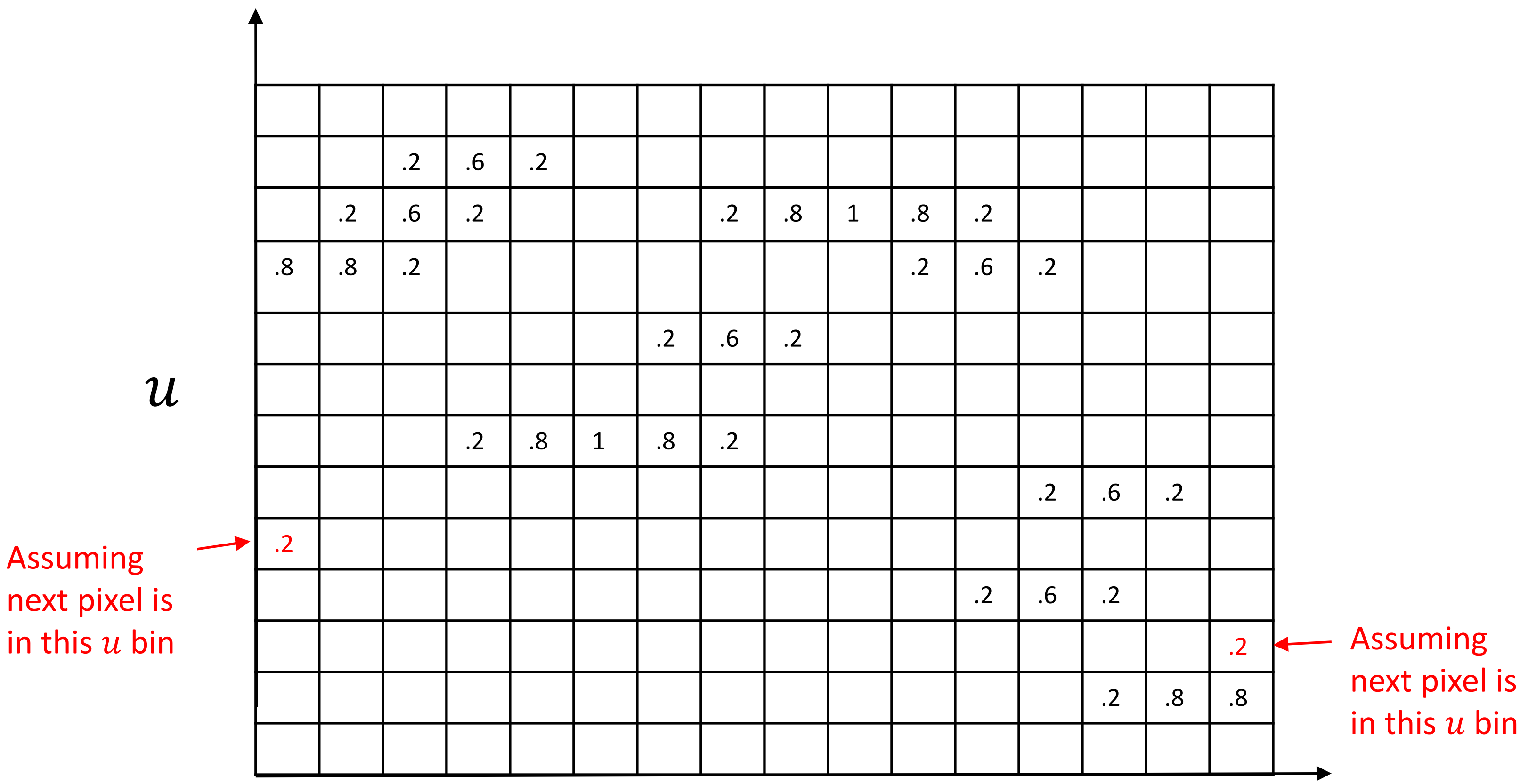
$$p(u;\mathbf{y}) = \sum_{\mathbf{x}} W(\mathbf{y}-\mathbf{x})\ \delta(u - bin(I(\mathbf{x})))$$



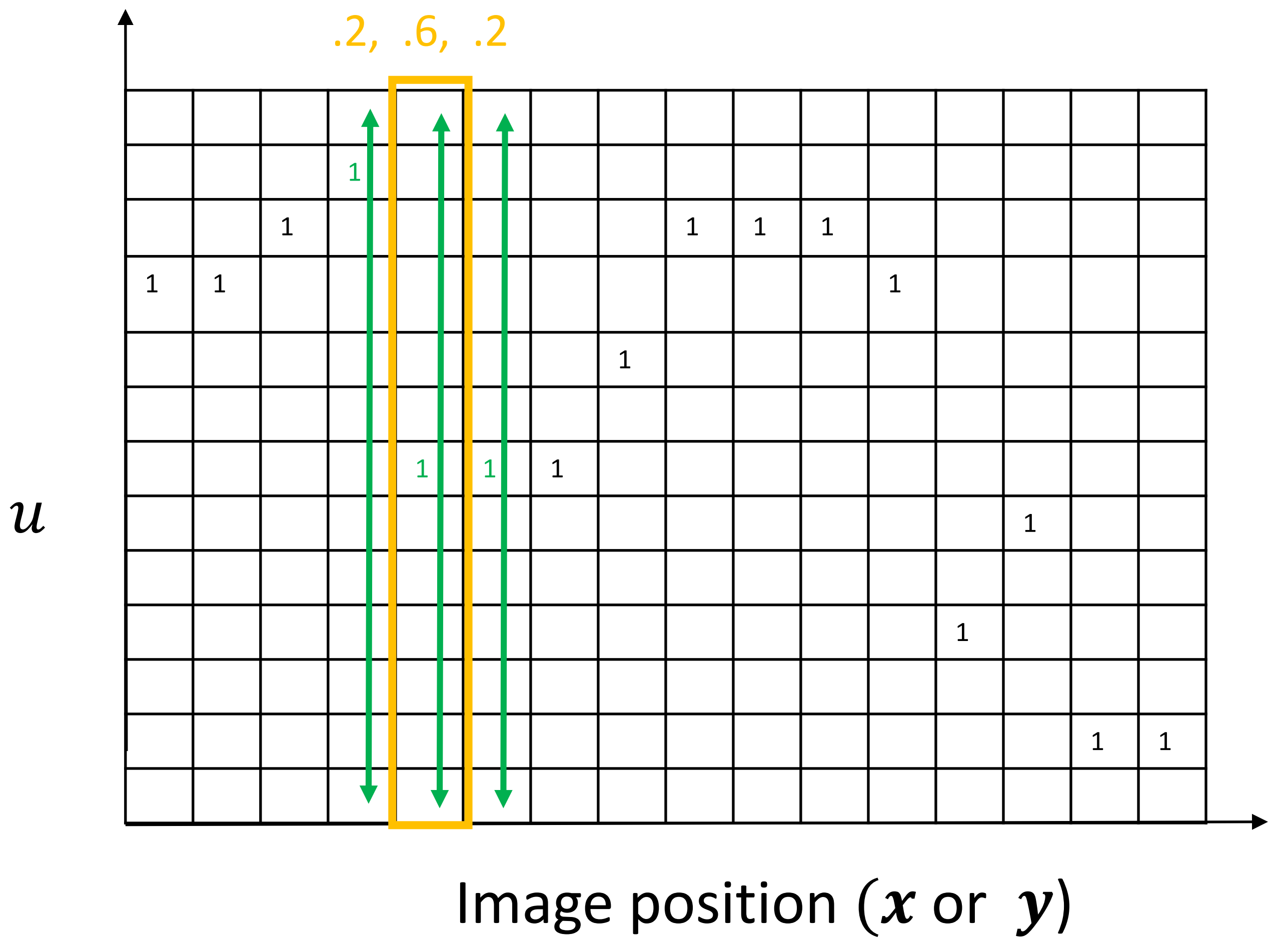
For the example below, we use $W(\boldsymbol{x})$ = (.2, .6, .2).

After convolving each row u with $W(\boldsymbol{x})$, we get the following.

Note that each column sums to 1, as before. Why?

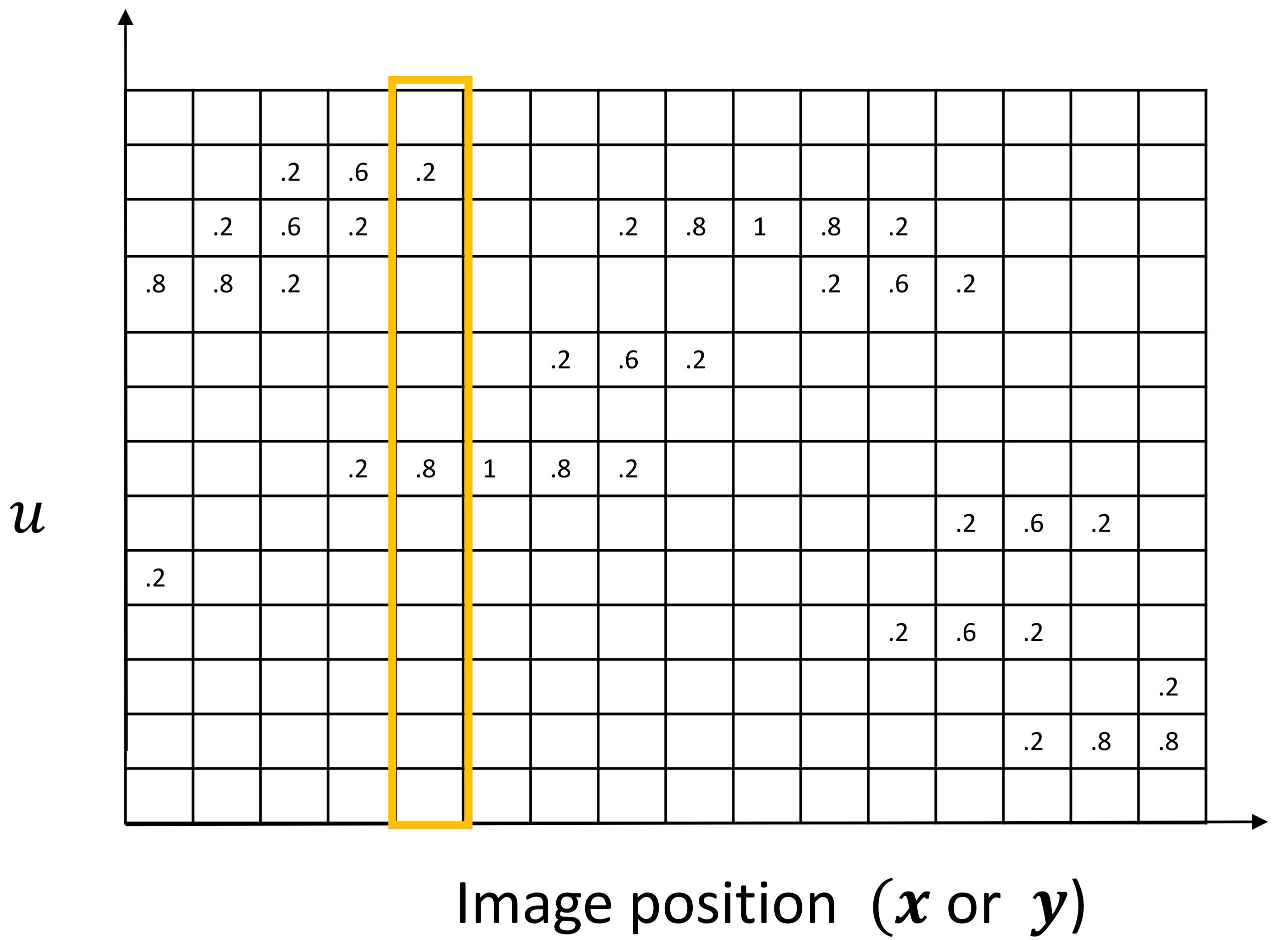


The convolution with $W(\boldsymbol{x})$ will result in each column receiving a contribution from its two neighbors and from itself. These contributions will sum to 1, regardless of which bins contains 1's.



$$p(u;\mathbf{y}) = \sum_{\mathbf{x}} W(\mathbf{y}-\mathbf{x})\ \delta(u - bin(I(\mathbf{x})))$$

Thus, $p(u;y)$ is a *probability function* for each image position $\boldsymbol{y}$.
That is, for each $\boldsymbol{y}$, $\sum_u p(u;y) = 1$ and $p(u;y) \geq 0$.



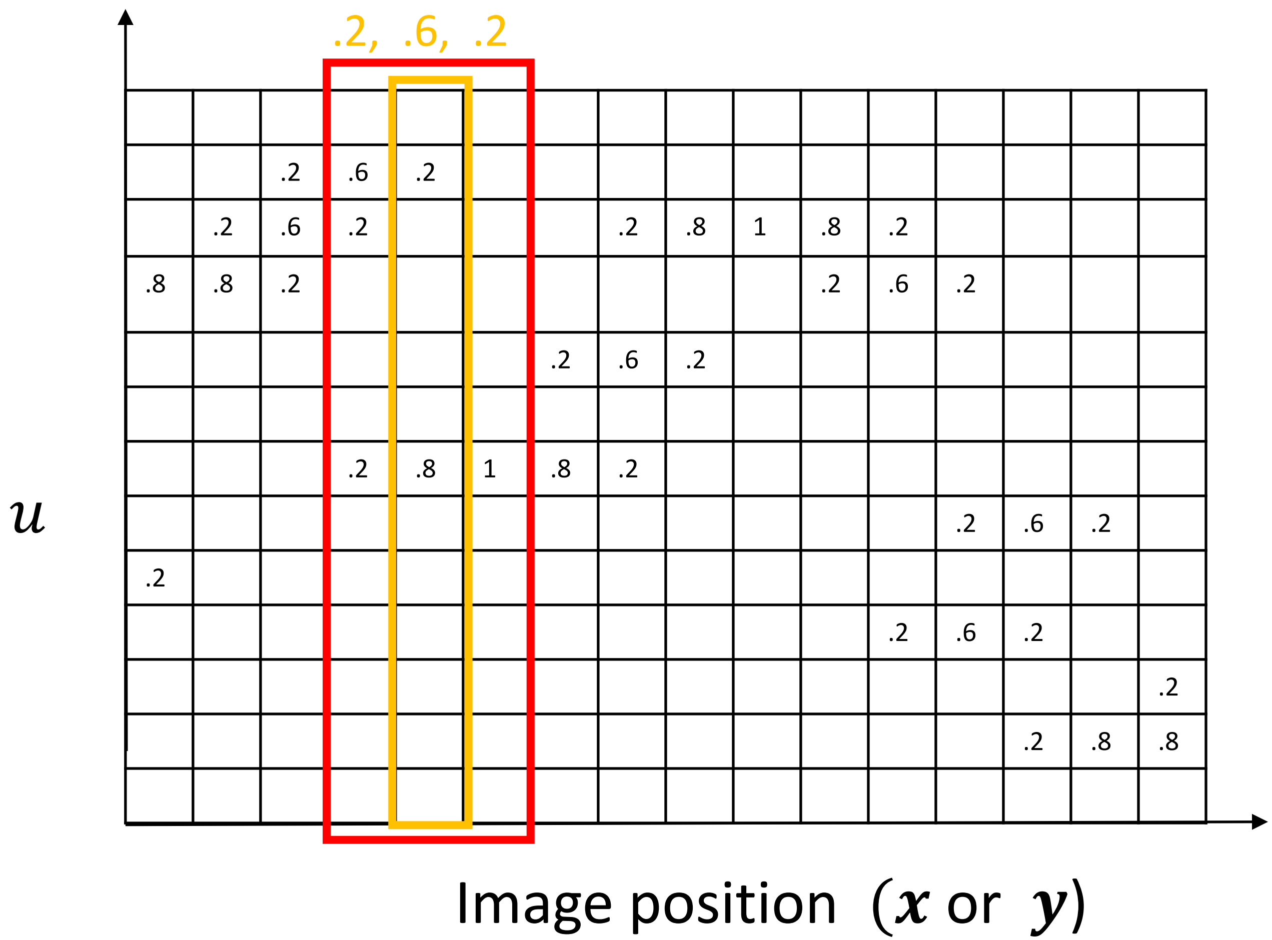
One typically uses a large neighborhood for $W(\boldsymbol{x})$.

We are *not* spatially blurring the RGB image.

Rather, we are computing a weighted histogram of the binned RGB triplets in the neighborhood of each image point $\boldsymbol{x}$.

$$p(u;\mathbf{y}) = \sum_{\mathbf{x}} W(\mathbf{y}-\mathbf{x})\ \delta(u - bin(I(\mathbf{x})))$$

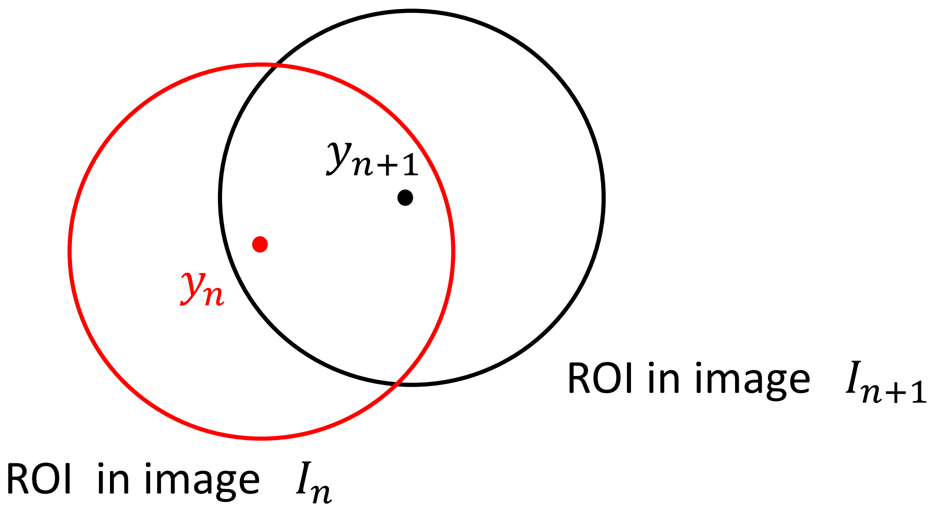
In our example, the neighborhood was of size 3, namely the width of the $W(\boldsymbol{x})$ function. But in practice, it will be larger!



Tracking problem:

Given position y_n centered at ROI in frame I_n,
find the nearby position y_{n+1} in frame I_{n+1} that *maximizes the similarity* of the weighted histograms.

How do we define this similarity ?



How to define the similarity of two probability functions?

Let $p(u)$ and $q(u)$ be two probability functions : $\sum_u p(u;y) = 1$
$\sum_u q(u;y) = 1$

The *Bhattacharya coefficient* is defined as:

$$BC(p,q) = \sum_u \sqrt{p(u)\,q(u)}$$

sum of "geometric means"

What is its value when $p(u) = q(u)$ for all u ?

What is its value when $p(u)\,q(u) = 0$ for all u ?

In general, its value ranges from 0 to 1.

Weighted histogram for ROI centered at y_n in frame I_n.

$$p_{n+1}(u;\mathbf{y}) = \sum_{\mathbf{x}} W(\mathbf{y}-\mathbf{x})\ \delta(u - bin(I_{n+1}(\mathbf{x})))$$

Weighted histogram for ROI centered at y in frame I_{n+1}.

$$p_n(u;\mathbf{y}_n) = \sum_{\mathbf{x}} W(\mathbf{y}_n-\mathbf{x})\ \delta(u - bin(I_n(\mathbf{x})))$$

Let $\boldsymbol{y_{n+1}}$ be the position $\boldsymbol{y}$ in frame $\boldsymbol{I_{n+1}}$ that *maximizes the Bhattacharya coefficient* :

$$BC(p_n(u;y_n), p_{n+1}(u;y)) = \sum_u \sqrt{p_n(u;y_n)\ p_{n+1}(u;y)}$$

Two problems with brute force tracking with ROI histogram comparison.

1) We should give more weight to the pixels near the center of the ROI. DONE.

2) Bruce force search is inefficient.

There is an algorithm called “mean shift” which can be used to solve this problem. (Details omitted. The basic idea is to do gradient descent on the Bhattacharya coefficient.)

See Mubarak Shah's video (start at 5 min in) if you are interested: https://www.youtube.com/watch?v=M8B3RZVqgOo

Summary

- When objects have moving parts, registration methods from lecture 8 don’t work.

- Instead, for any ROI in one frame, find ROI in next frame whose weighted histogram is most similar.

Reminders

- Assignment 1 due tonight at midnight
- Monday is Thanksgiving (no class)
- Quiz 2 is on Wed. Oct. 14
- Assignment 2 will be posted end of next week