Introduction to Augmented Reality

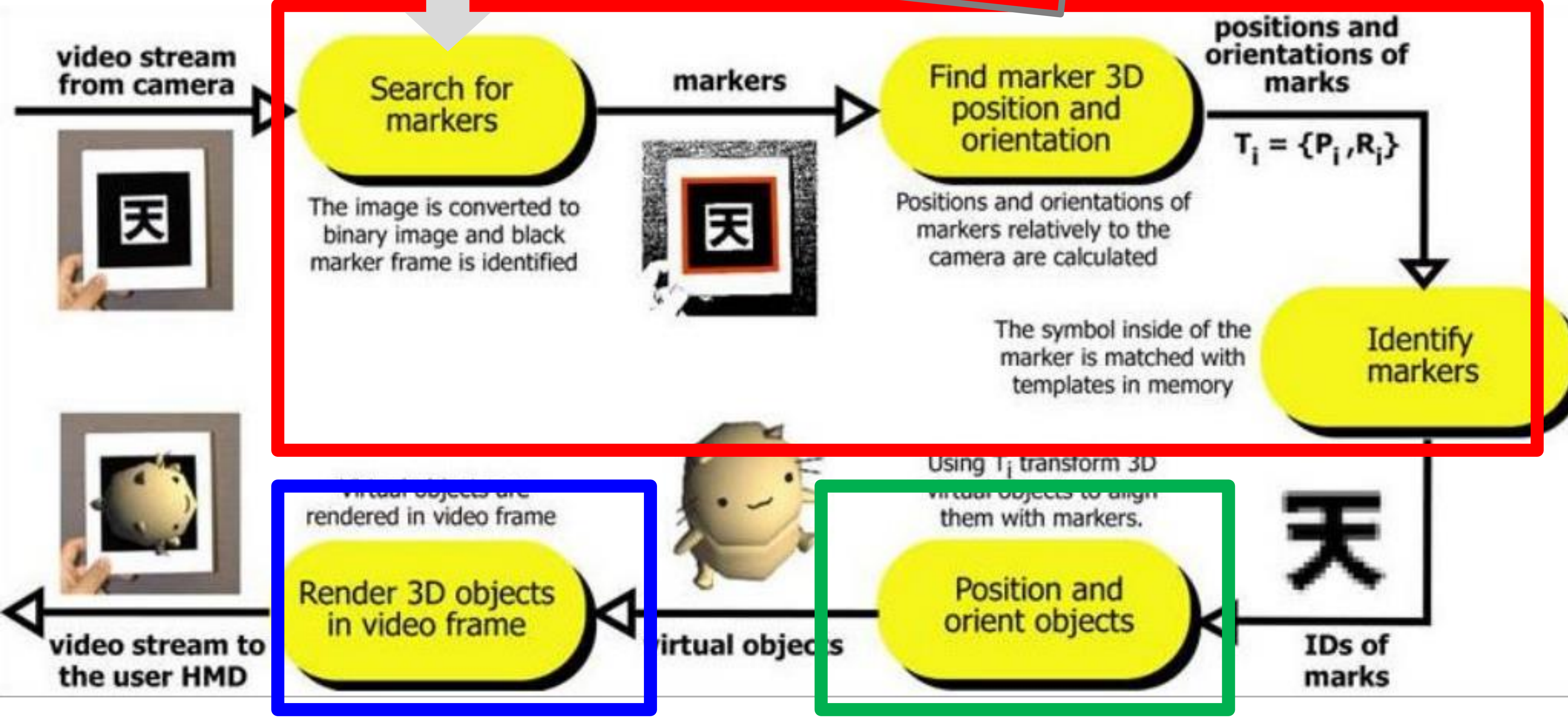
Tutorial 3: Marker Tracking Part 3
May 2nd 2018

Andreas Langbein, Adnane Jadid, David Plecher

Marker-based Tracking

today

Ex. 1~5



video stream from camera

Search for markers

The image is converted to binary image and black marker frame is identified

markers

Find marker 3D position and orientation

Positions and orientations of markers relatively to the camera are calculated

positions and orientations of marks

$T_i = \{P_i, R_i\}$

Identify markers

The symbol inside of the marker is matched with templates in memory

IDs of marks

Using T_i transform 3D virtual objects to align them with markers.

Position and orient objects

virtual objects

Virtual objects are rendered in video frame

Render 3D objects in video frame

video stream to the user HMD

ARToolKit

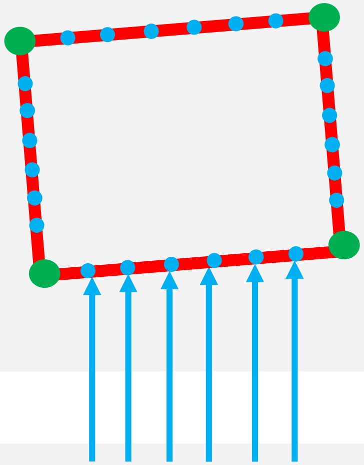
Ex. 8~9

Ex. 6~7

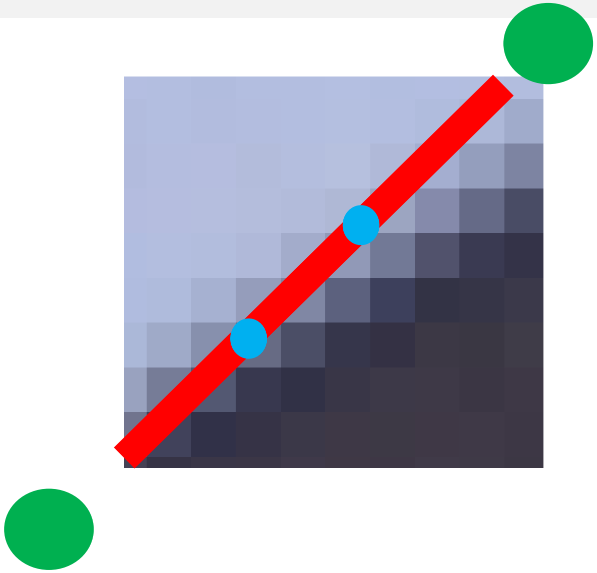
Solution for the Previous Tutorial

Ex. 2

Find marker in 2D



seven intervals and six delimiters

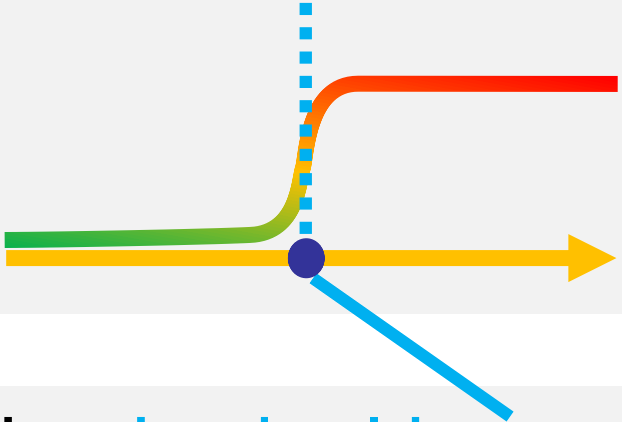


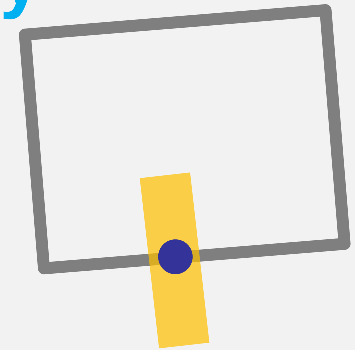
→ Code walkthrough

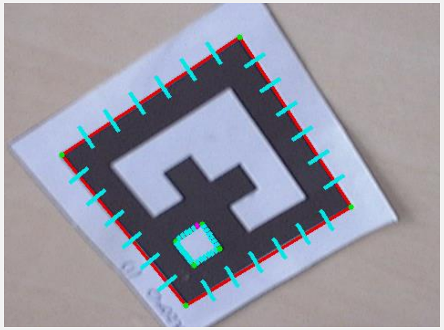
Today’s Tutorial

Ex. 3

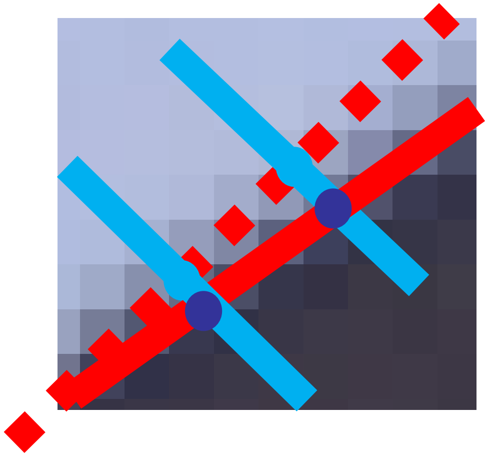
Find marker in 2D **precisely**







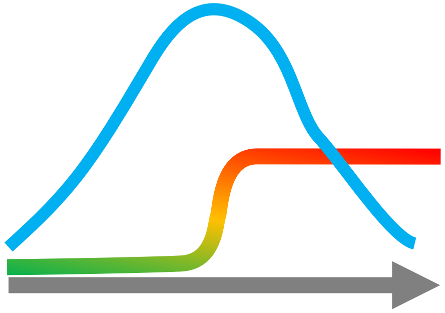
Find sub-pixel boundary of the edges

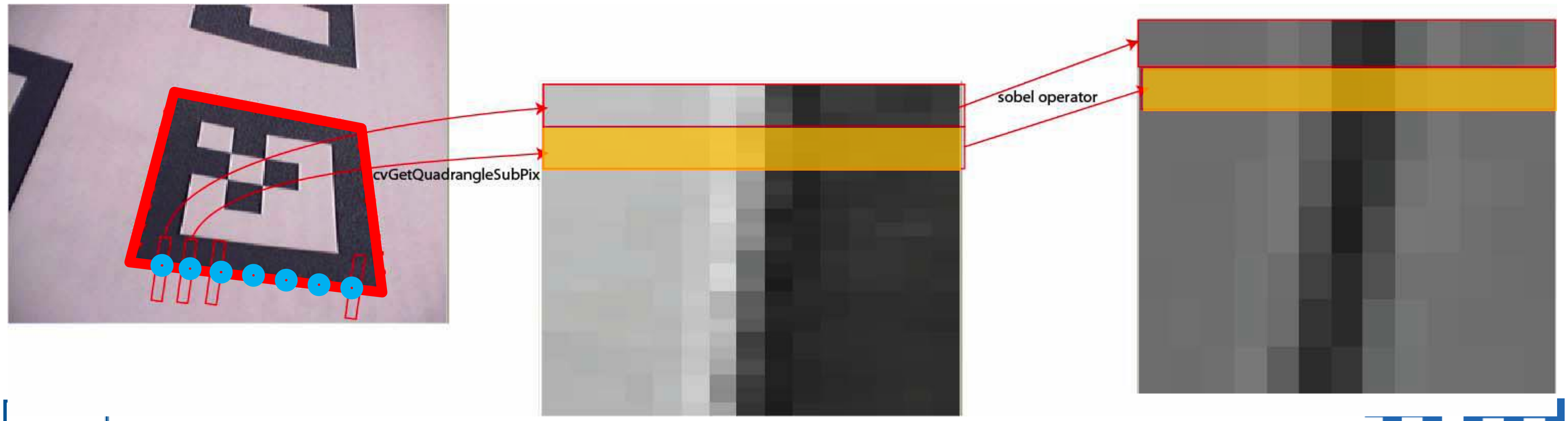


Extract stripes perpendicular to sides

1. Pick up 6 points along each sides
2. Extract **3 pixel-wide** stripes perpendicular to the sides
   - `cv::GetQuadrangleSubPix(…)`
3. Apply Sobel operator:
   - Intensity change

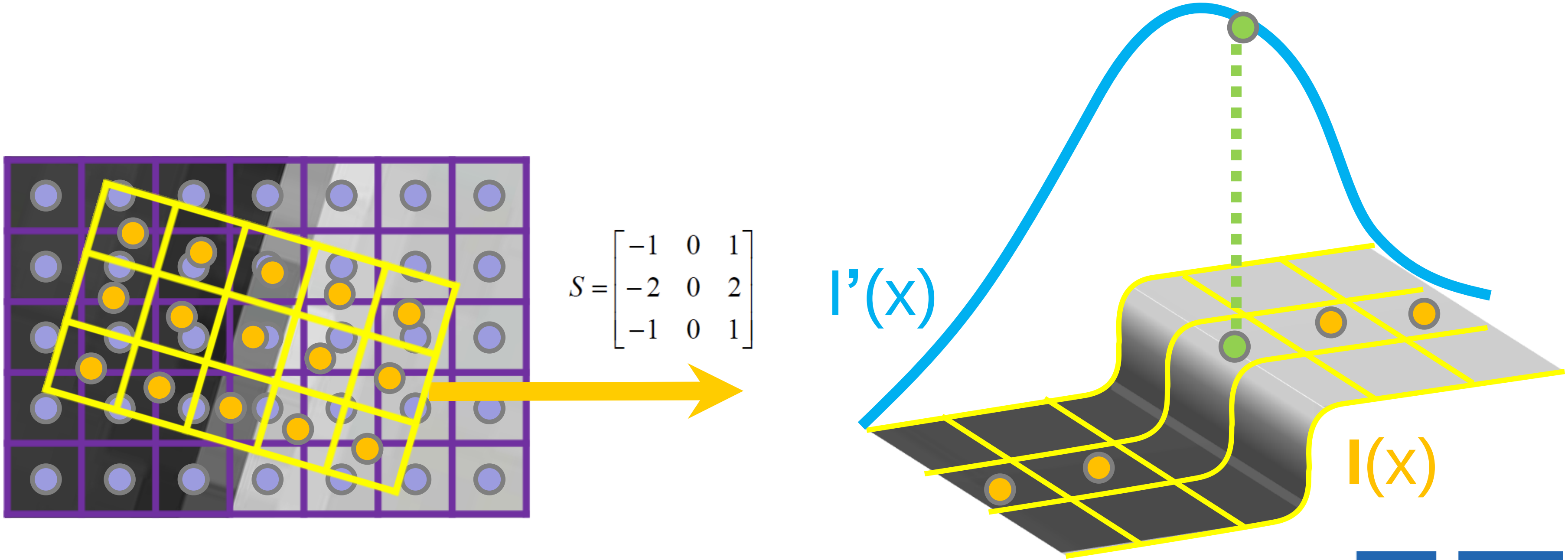
$$S = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$





Subpixel-precise localization of edges

1. Finding the **point** with highest intensity change in each stripe

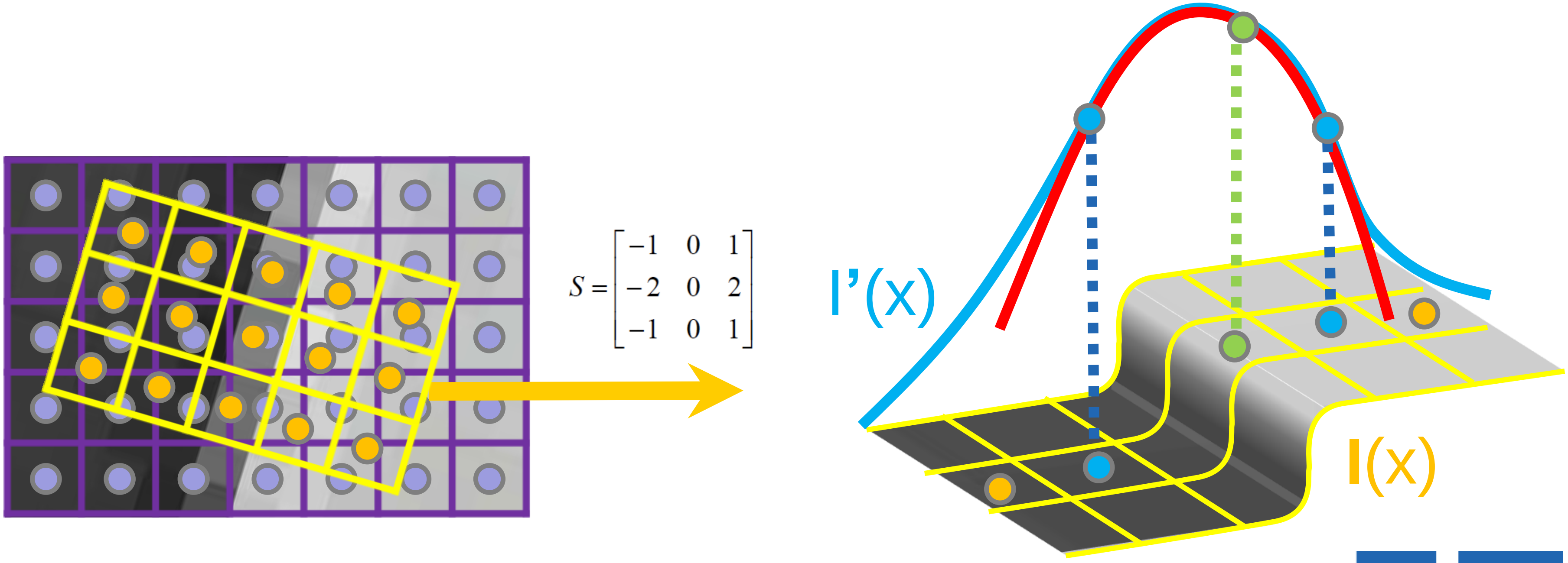


$$S = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Subpixel-precise localization of edges

1. Finding the **point** with highest intensity change in each stripe
2. Fit parabola **f** through the **point** and two **neighbors**

$$f(x) = ax^2 + bx + c$$

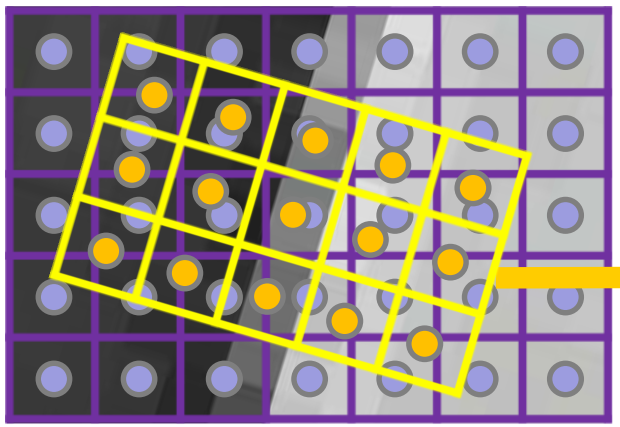


$$S = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

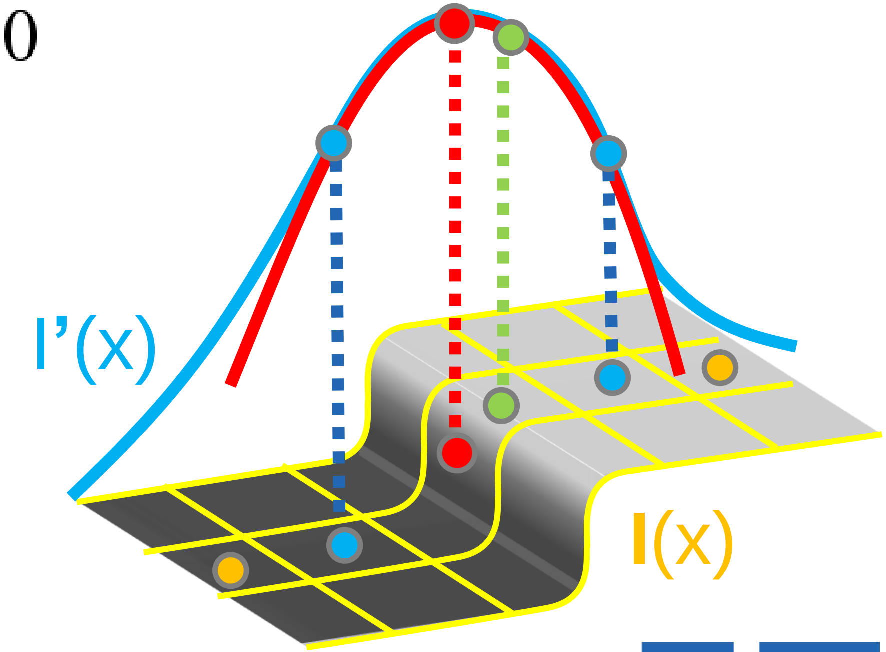
Subpixel-precise localization of edges

1. Finding the **point** with highest intensity change in each stripe
2. Fit parabola **f** through the **point** and two **neighbors** $f(x) = ax^2 + bx + c$
3. Find the peak (zero of first derivative)

f(x')=0 $2ax + b = 0$

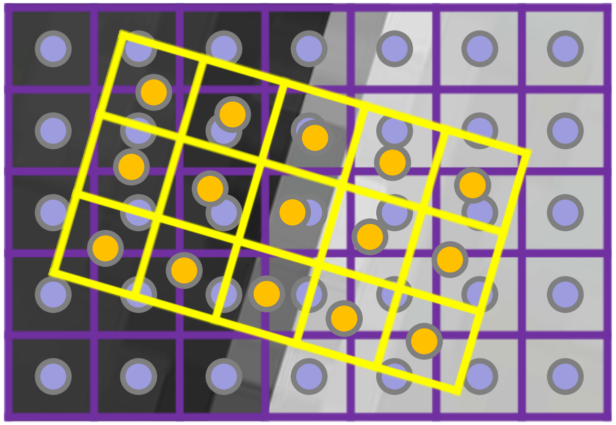


$$S = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

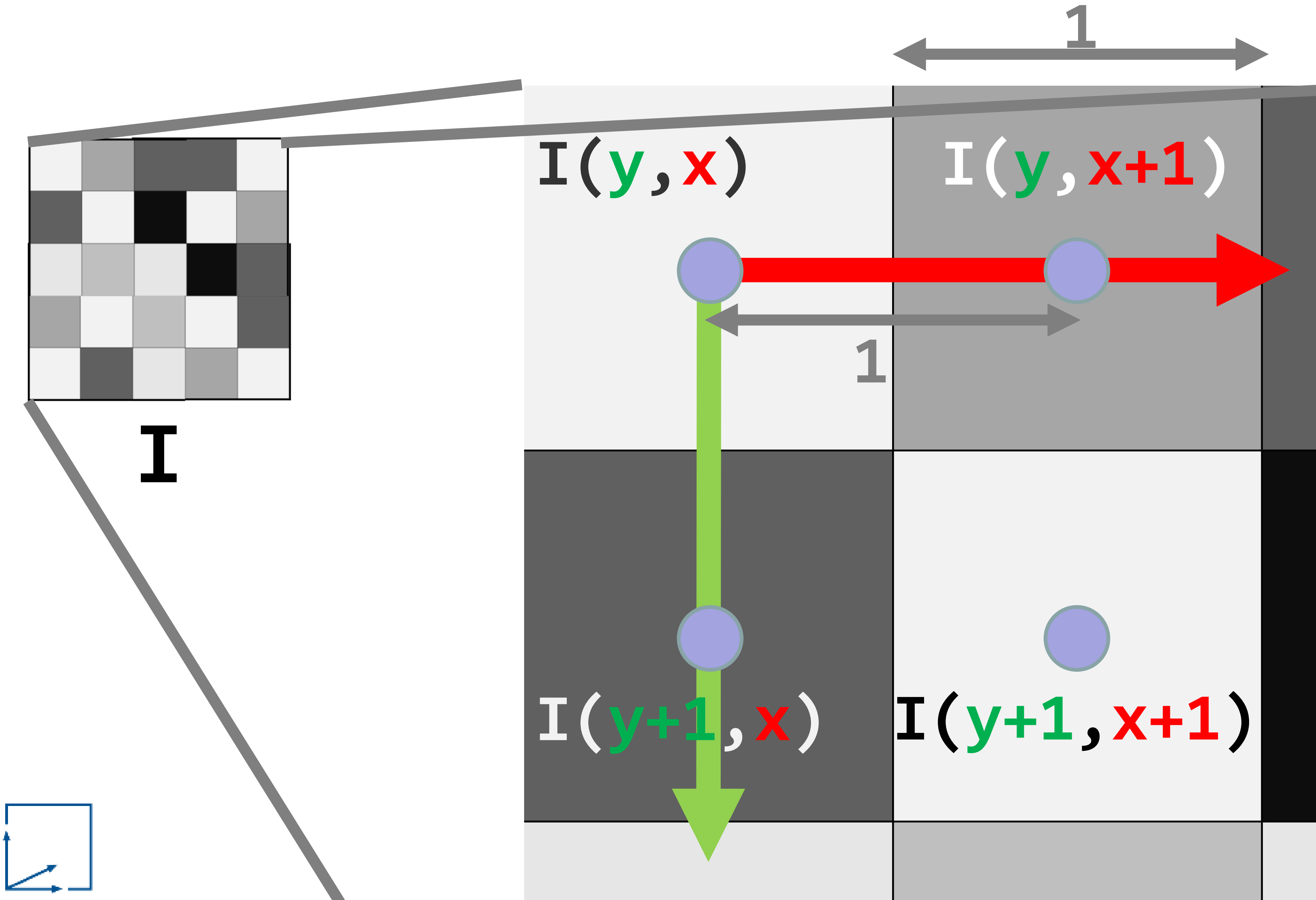


Subpixel-precise localization of edges

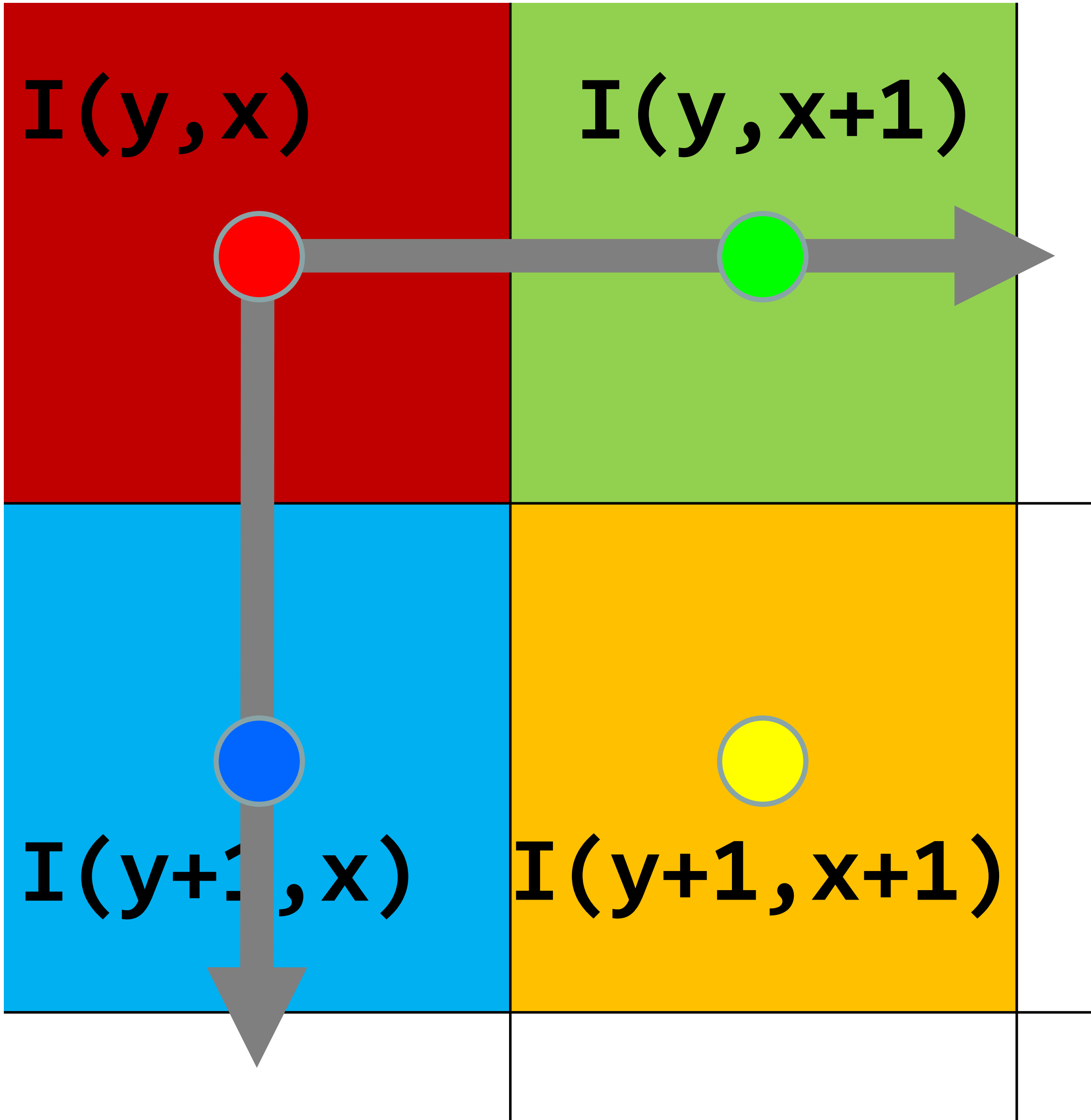
How to get **subpixel color** from **image pixels**?



Pixel Coordinates



Pixel Coordinates



I(y,x)

I(y,x+1)

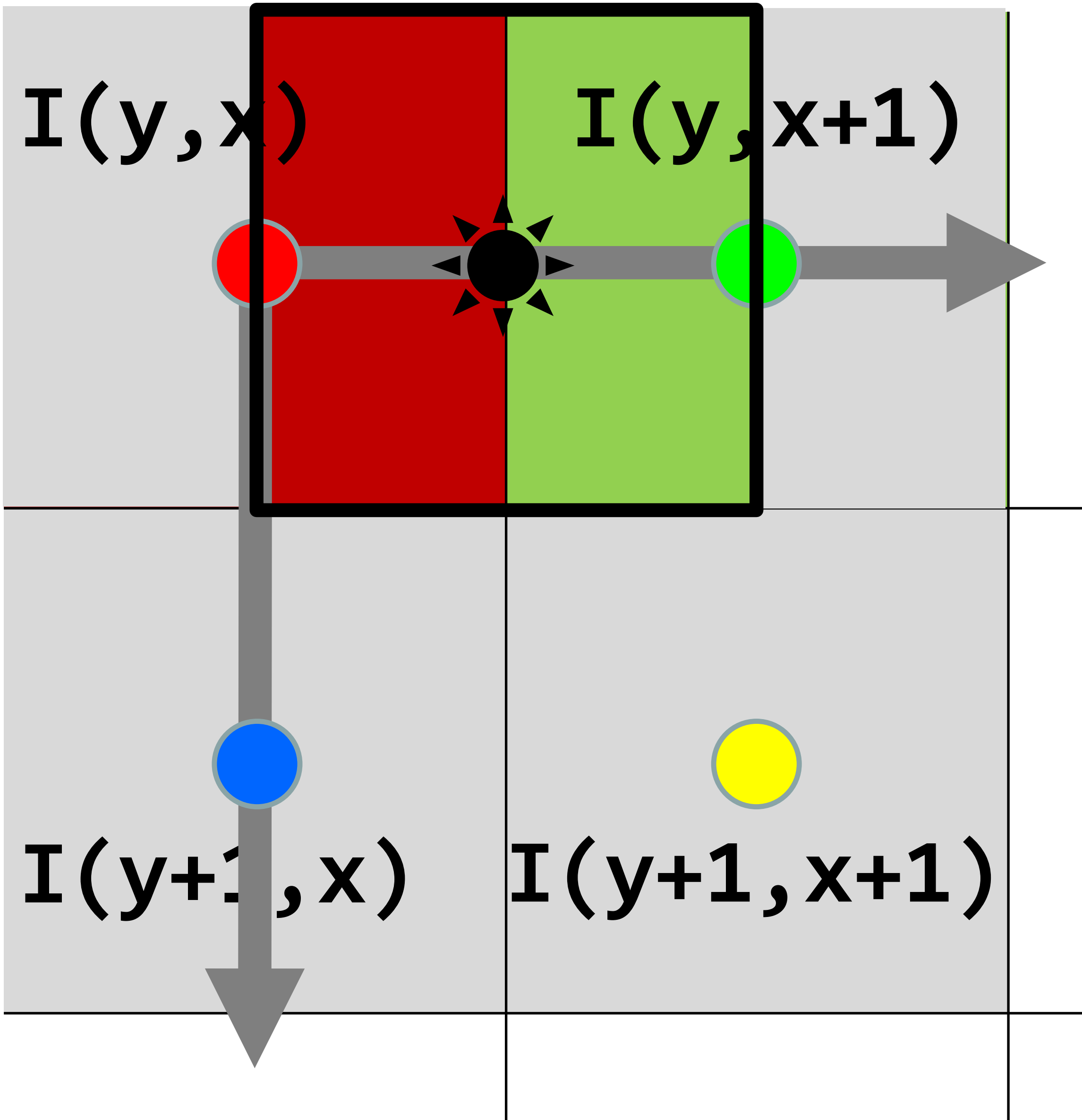
I(y+1,x)

I(y+1,x+1)

I(y,x+0.5)

= 1/2 🔴

+ 1/2 🟢



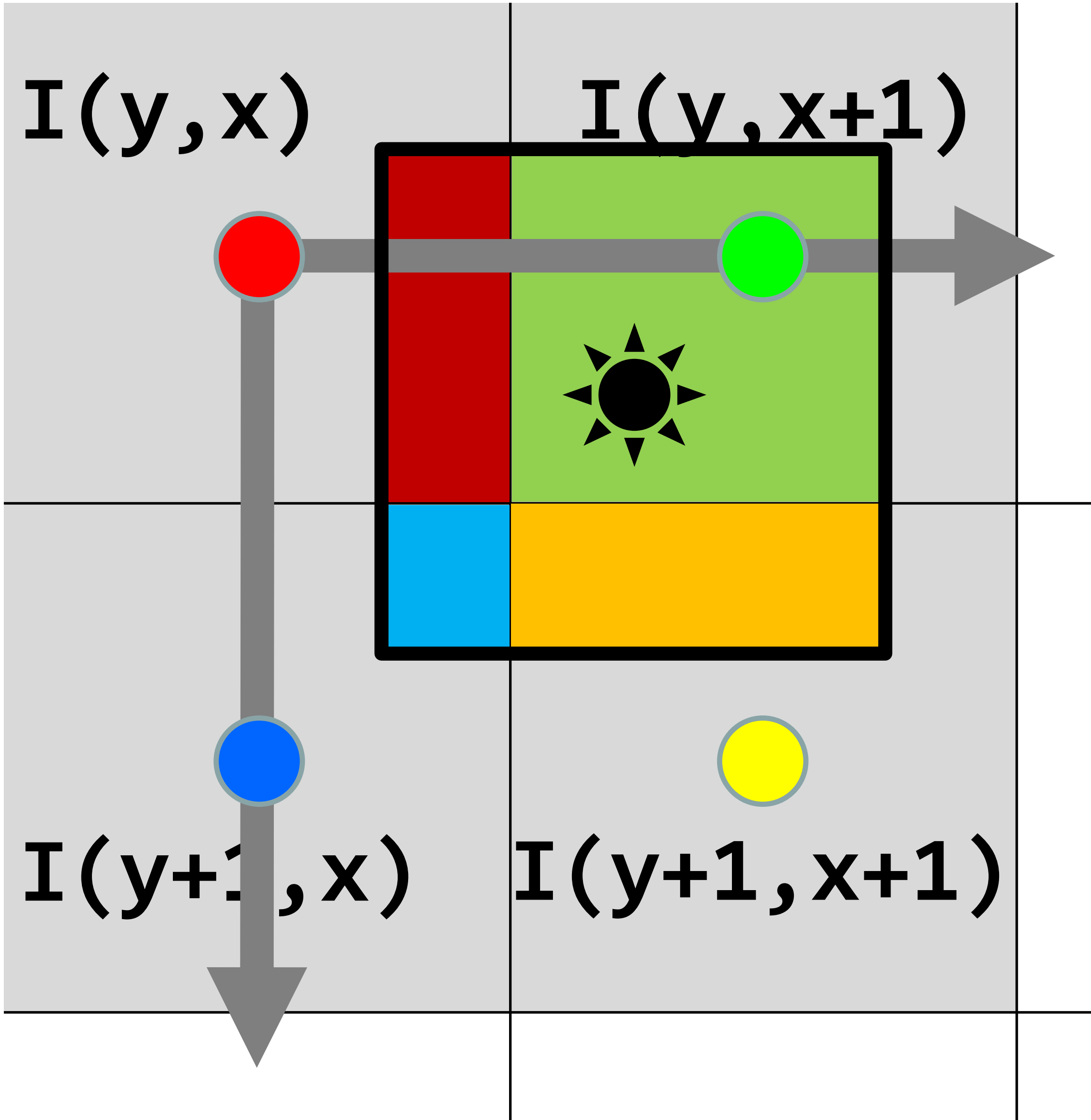
I(y+0.25,x+0.75)

= 3/16 ●

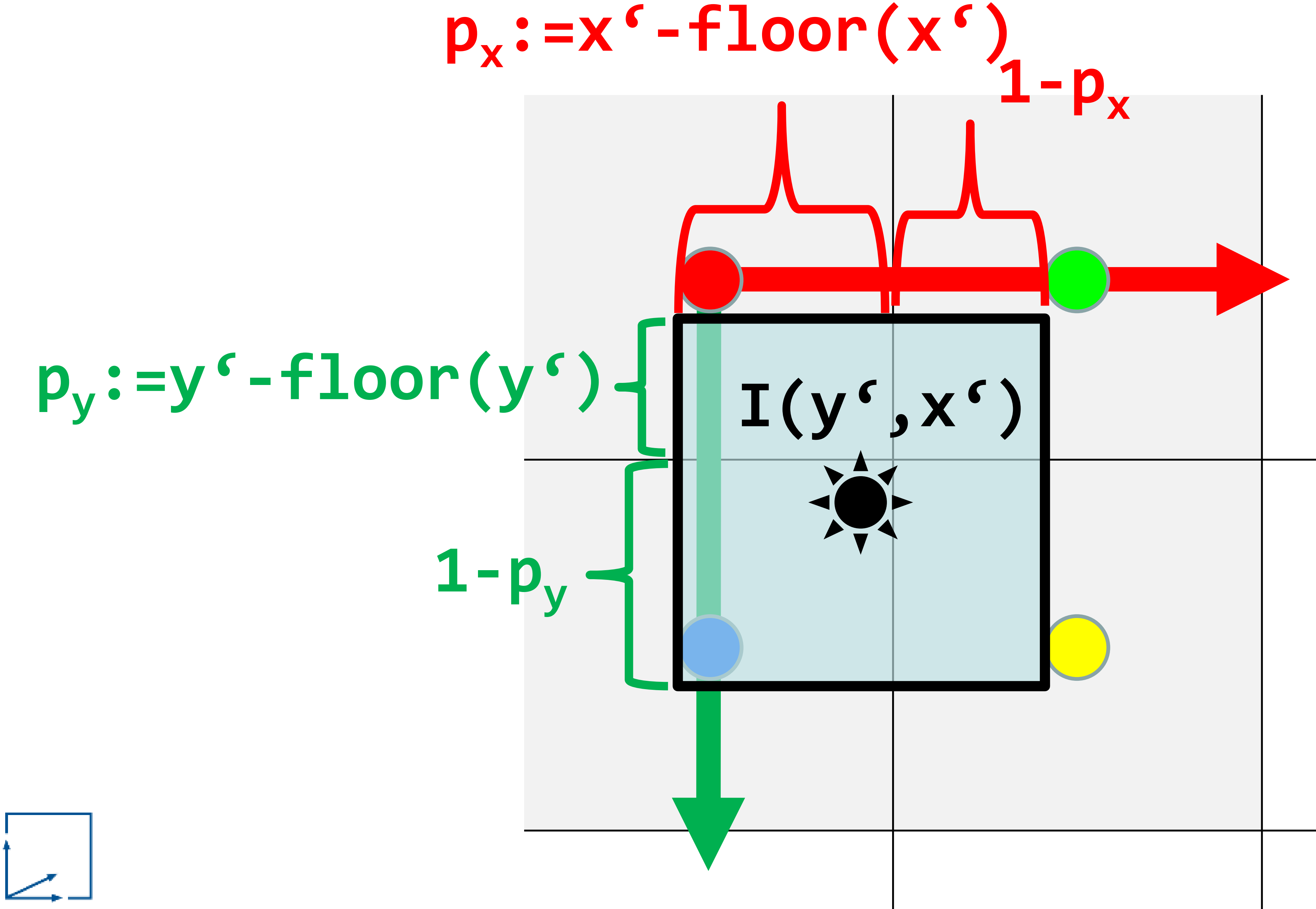
+ 9/16 ●

+ 1/16 ●

+ 3/16 ●



$I(y',x')$



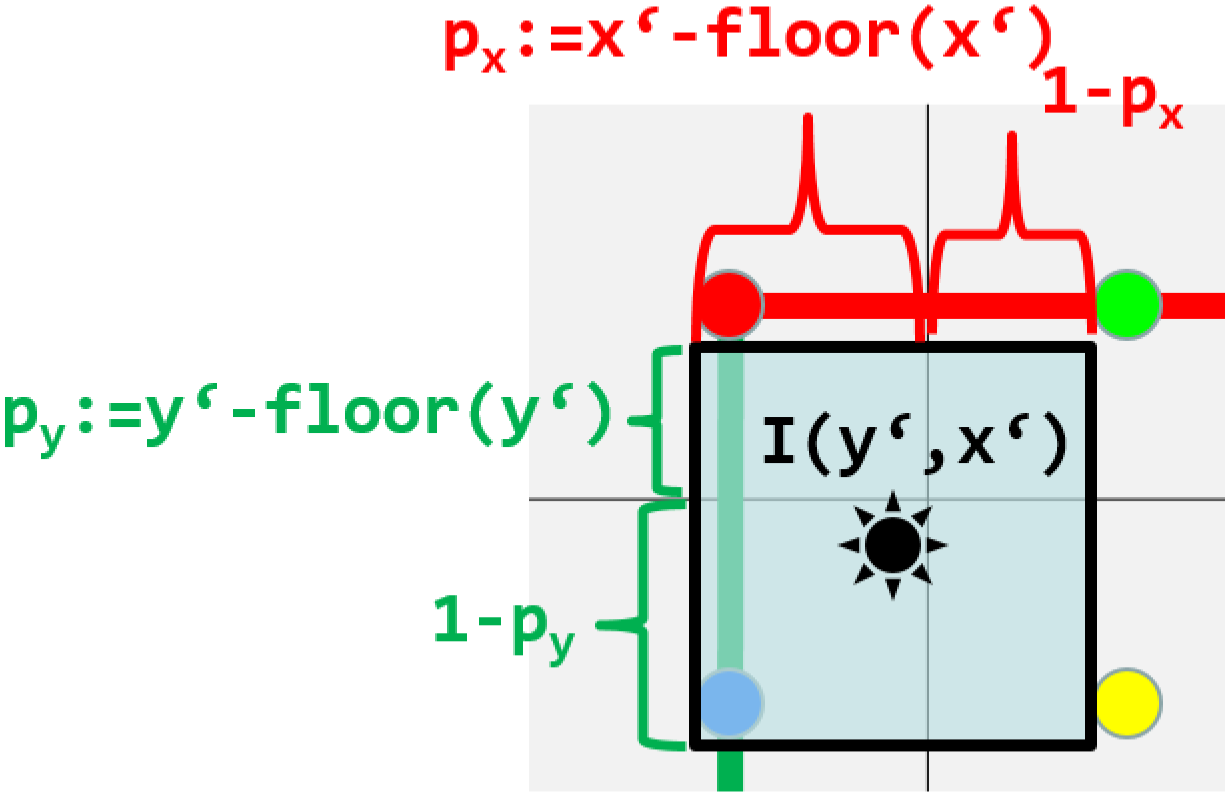
$p_x := x' - \text{floor}(x')$

$1 - p_x$

$p_y := y' - \text{floor}(y')$

$1 - p_y$

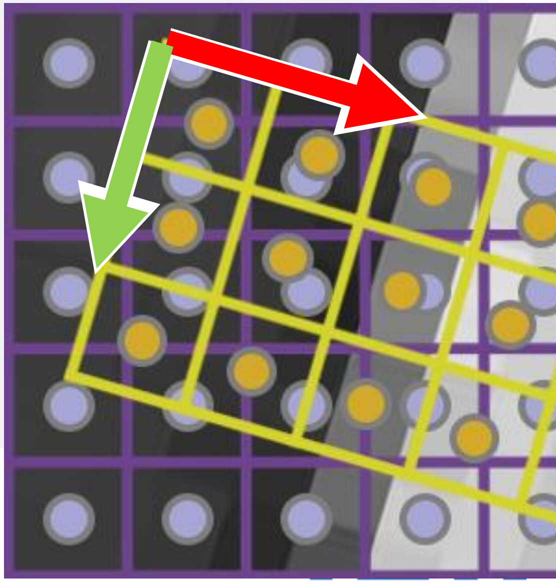
$I(y',x')$



```
fx=floor(x)
fy=floor(y)
px=1-fx
py=1-fy

I(x,y) =
(1-py)*((1-px) * I(fx,fy)  +px*I(fx+1, fy  ))+
   py *((1-px) * I(fx,fy+1)+px*I(fx+1, fy+1))
```

Sketch Solution for Exercise 3

```
for (int i=0; i<4; ++i) {
    ...
    // Stripe size
    int stripeLength = (int)(0.8*sqrt (dx*dx+dy*dy));
    if (stripeLength < 5)
     stripeLength = 5;
    cv::Size stripeSize;
    stripeSize.width = 3;
    stripeSize.height = stripeLength;

    // Direction vectors
    cv::Point2f stripeVecX;
    cv::point2f stripeVecY;
    double diffLength = sqrt ( dx*dx+dy*dy );
    stripeVecX.x = dx / diffLength;
    stripeVecX.y = dy / diffLength;
    stripeVecY.x =  stripeVecX.y;
    stripeVecY.y = -stripeVecX.x;
    ...
}
```



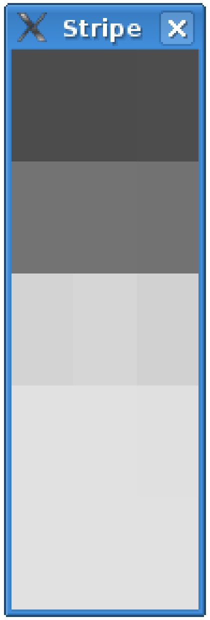
Sketch Solution for Exercise 3

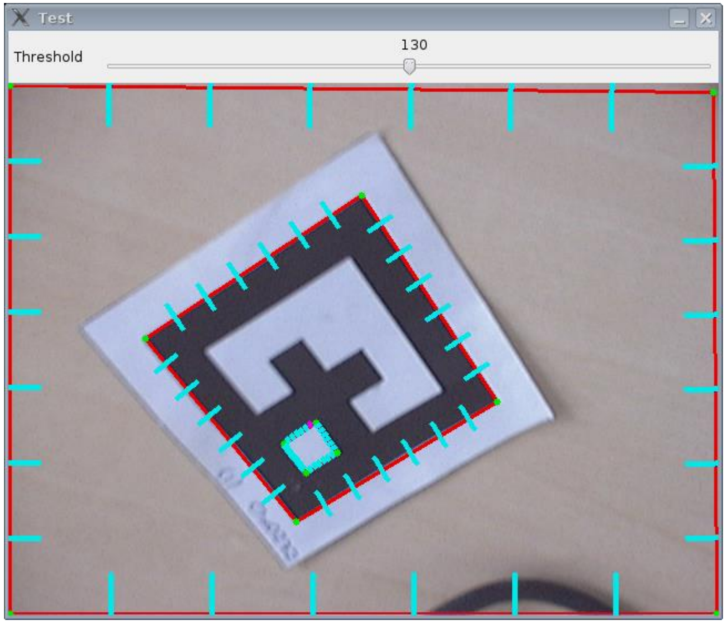
```
// For each stripe
for (int j=1; j<7; ++j)
{
...
cv::Mat iplStripe( stripeSize, CV_8UC1 );
// Stripe width
    for ( int m = -1; m <= 1; ++m )
    {
        // Stripe length
        for ( int n = nStart; n <= nStop; ++n )
        {
            cv::Point2f  subPixel;
            subPixel.x = ...;
            subPixel.y = ...;
            ...
            int pixel = subpixSampleSafe (img_gray, subPixel);
            int w = m + 1;
            int h = n + ( stripeLength >> 1 );

            iplStripe.at<uchar>(h,w) = (uchar)pixel;
        }
    }
}
```

Note: img.at<uchar>(y,x)

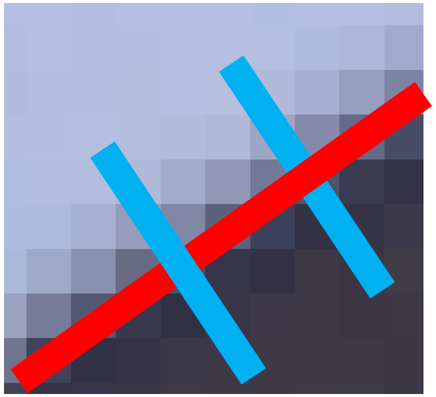
Homework

Find stripes with subpixel accuracy
(see homework sheet on Moodle)



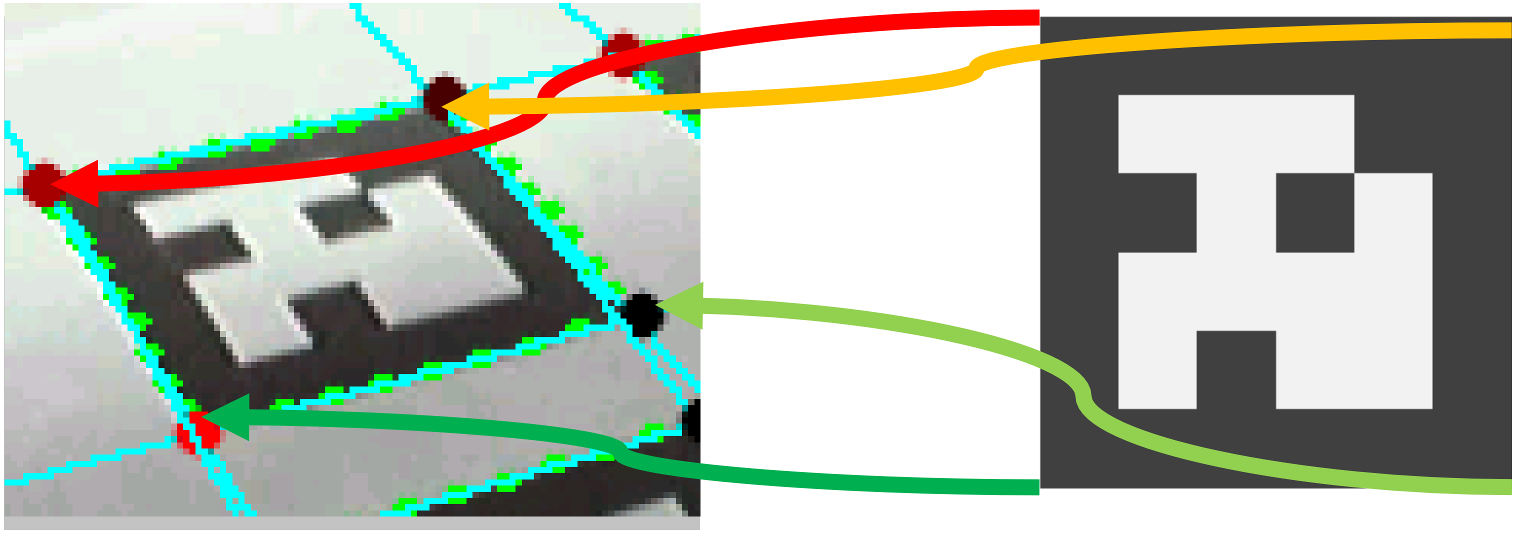


Threshold 130



Spoiler of the next tutorial

1. Compute exact corner points
   - Fit lines through all points and mark in picture (cyan lines)
2. Rectify contained image



That‘s it…

- Questions