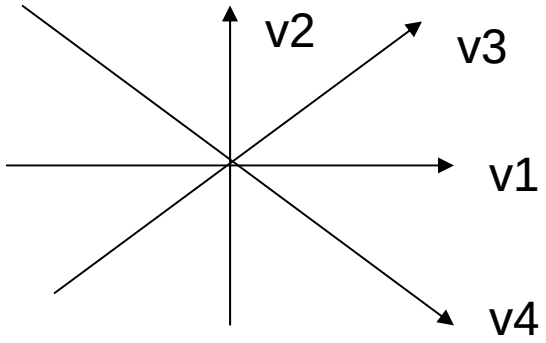
Image Patches

Point and Patch features

- Want to find distinctive points or patches in the image
  - Could match these features to a 3D model, for object recognition
  - Or track them from one image to another, for motion or structure estimation
- Want patches that are locally unique
  - Good types of features: bright dots, corners
  - Bad types of features: regions with constant value, or long straight edges
- We will look at
  - Moravec interest operator
  - KLT corner detector
  - How to match features between images
  - SIFT (Scale invariant feature transform)

Moravec Interest Operator

- Find points where the local variance in vertical, horizontal, and diagonal directions are all high

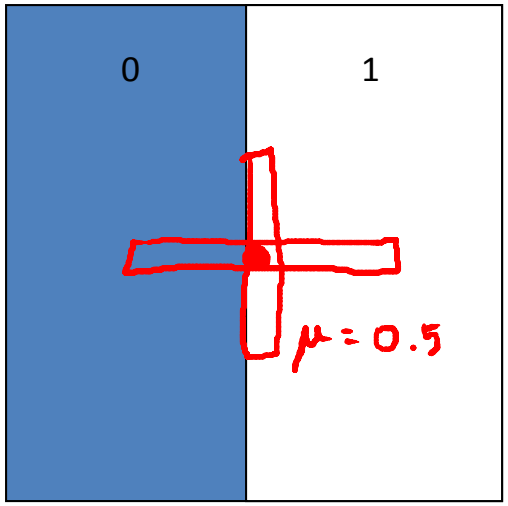


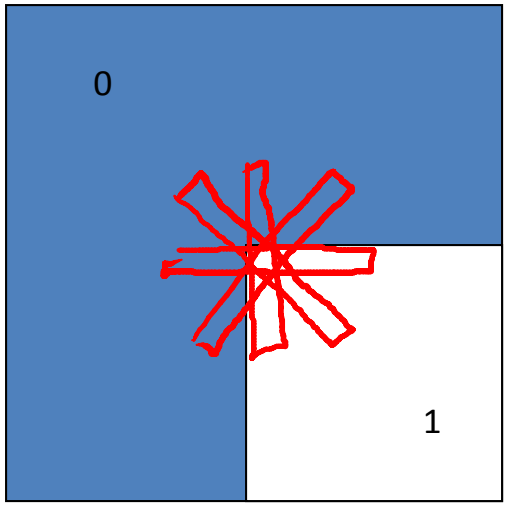
- Compute:
  - v1 = variance for horizontal pixels I(x-w,y):I(x+w,y)
  - v2 = variance for vertical pixels I(x,y-w):I(x,y+w)
  - v3 = variance for diagonal pixels I(x-w,y-w):I(x+w,y+w)
  - v4 = variance for diagonal pixels I(x+w,y-w):I(x-w,y+w)

- Interest value = min(v1,v2,v3,v4)

Examples

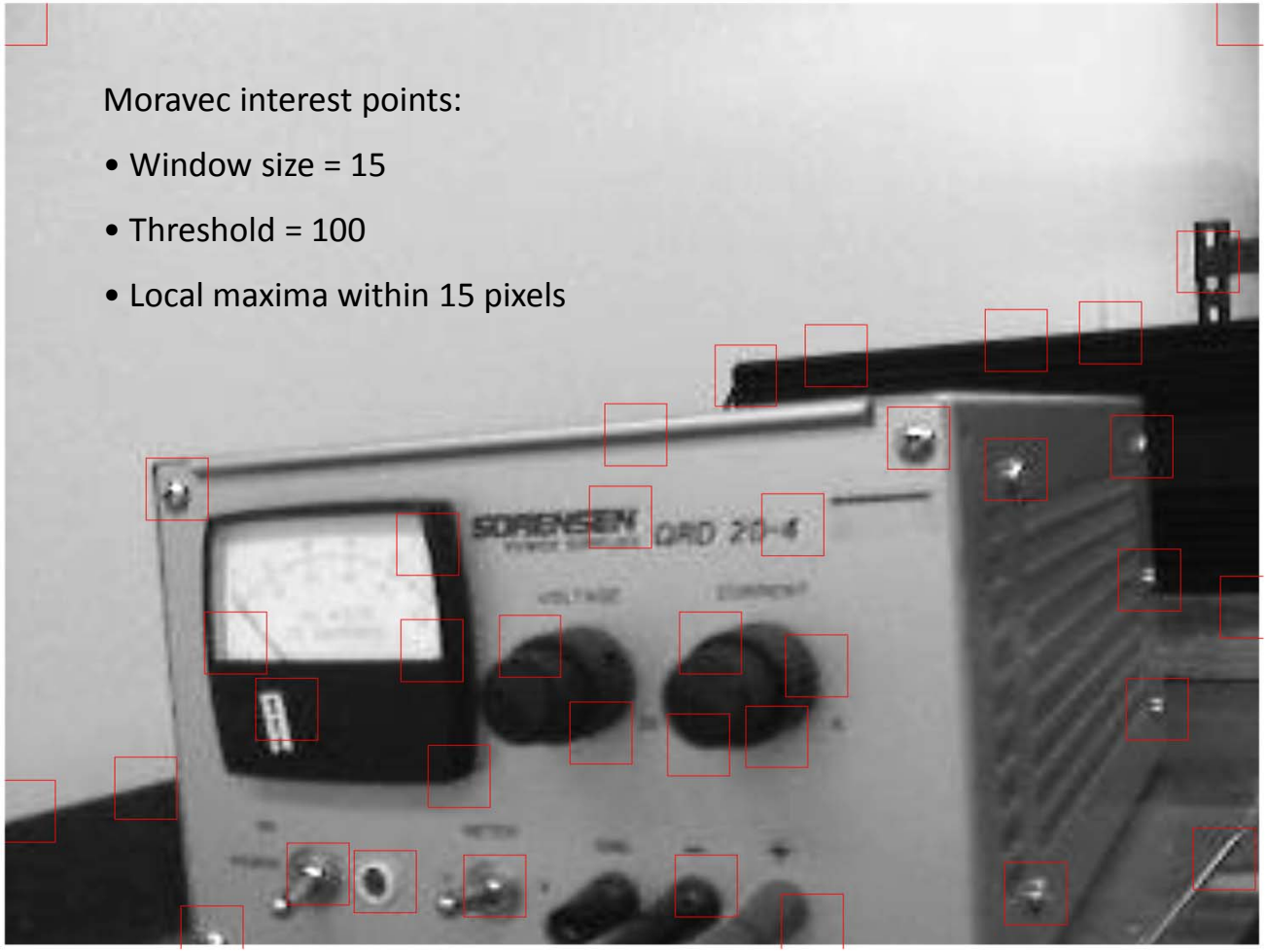
- Moravec interest operator score is low
  - In uniform regions
  - Along vertical or horizontal step edges
- The score is high at corners





$V_{HORIZ} = \frac{1}{N}\sum (x_i - \mu)^2 \qquad V_{VERT} = 0$

$= \frac{1}{N}\left(\frac{N}{2}\right)(-.5)^2 + \frac{1}{N}\left(\frac{N}{2}\right)(.5)^2 = .25$


Moravec interest points:
• Window size = 15
• Threshold = 100
• Local maxima within 15 pixels
SORENSEN
QRD 20-4

Notes on Implementation

- Recall definition of variance, and its alternative formulation
- Rather than numerous nested for-loops, can use Matlab's vector and array operators
- Approach:
  - Compute mean of local window at each point
  - Compute square at each point, and local sum
  - Combine to get local variance

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2 \qquad \mu = \frac{1}{N}\sum_{i=1}^{N} x_i$$

$$= \frac{1}{N}\sum_{i=1}^{N}\left(x_i^{\,2} - 2\mu x_i + \mu^2\right)$$

$$= \frac{1}{N}\sum_{i=1}^{N} x_i^{\,2} - \frac{2\mu}{N}\sum_{i=1}^{N} x_i + \frac{\mu^2}{N}\sum_{i=1}^{N} 1$$

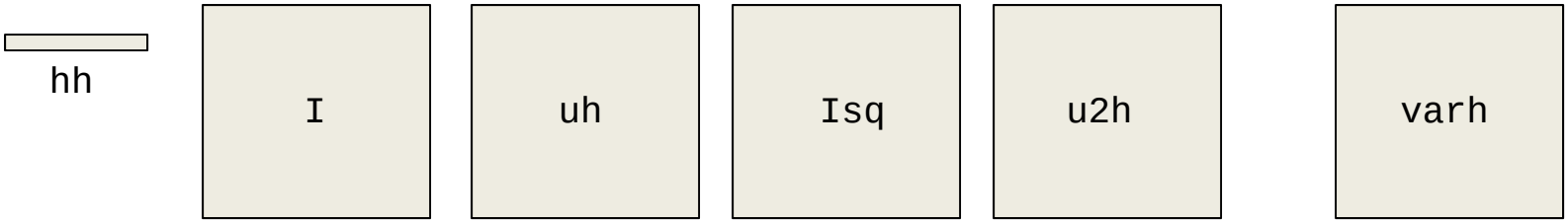
$$= \frac{1}{N}\sum_{i=1}^{N} x_i^{\,2} - 2\mu^2 + \mu^2$$

$$= \frac{1}{N}\sum_{i=1}^{N} x_i^{\,2} - \mu^2$$

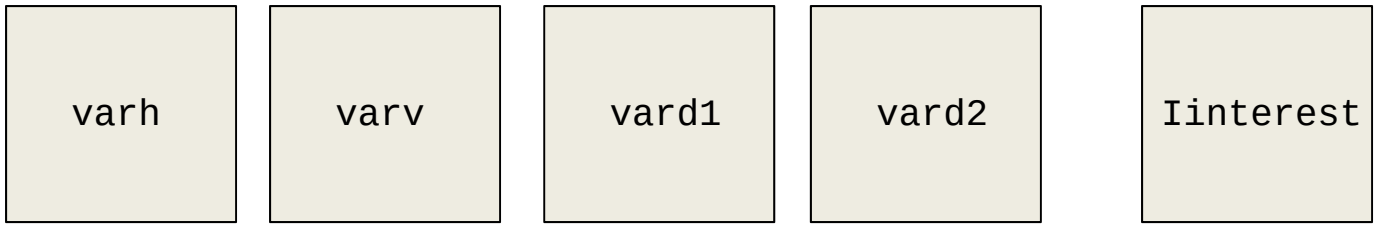
Matlab Implementation

- Example for horizontal (1xN) window

```
hh = ones(1,N);
uh = (1/N)*imfilter(I, hh);   % mean of horizontal
Isq = I .^ 2;
u2h = (1/N)*imfilter(Isq, hh);   % mean of squares
varh = u2h - uh.^2;          % variance of horizontal
```



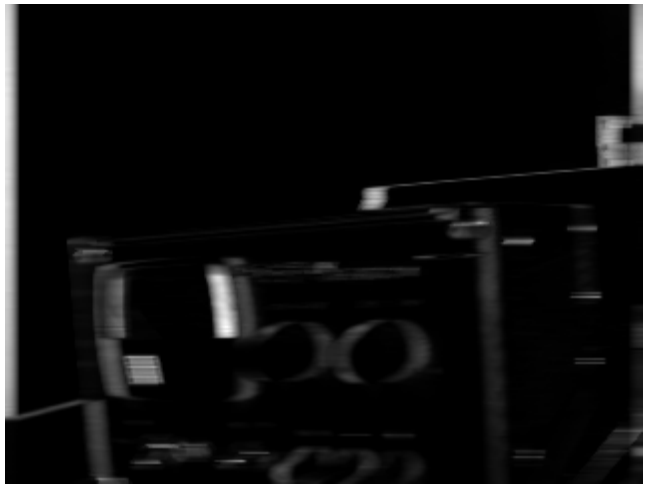
- Then compute variances in other directions, and take use Matlab's min function



- This will produce an image of interest point scores at each pixel

```
% Detect interest points using Moravec operator
clear all
close all

I1 = double(imread('test000.jpg'));

N = 15;
hh = ones(1,N);     % horizontal
hv = hh';           % vertical
hd1 = eye(N,N);     % diagonal1
hd2 = fliplr(hd1);  % diagonal2

uh = (1/N)*imfilter(I1, hh);   % mean of horizontal
uv = (1/N)*imfilter(I1, hv);   % mean of vertical
ud1 = (1/N)*imfilter(I1, hd1);   % mean of diagonal1
ud2 = (1/N)*imfilter(I1, hd2);   % mean of diagonal2

I1sq = I1 .^ 2;
u2h = (1/N)*imfilter(I1sq, hh);   % mean of horizontal squares
u2v = (1/N)*imfilter(I1sq, hv);   % mean of vertical squares
u2d1 = (1/N)*imfilter(I1sq, hd1);   % mean of diagonal1 squares
u2d2 = (1/N)*imfilter(I1sq, hd2);   % mean of diagonal2 squares

varh = u2h - uh.^2;         % variance of horizontal
varv = u2v - uv.^2;         % variance of vertical
vard1 = u2d1 - ud1.^2;      % variance of diagonal1
vard2 = u2d2 - ud2.^2;      % variance of diagonal2

Iinterest = min(min(varh,varv), min(vard1,vard2));
```

horiz



vert

diag1

diag2

Minima of the four directions

min

Problem with Moravec