Person Detection in Images using HoG + Gentleboost

Rahul Rajan
June 1st – July 15th
CMU-Q Robotics Lab

Introduction

- One of the goals of computer vision -
  - Object-class detection – car, animal, humans
    - Human Computer Interaction (Hala, roboceptionist)
    - Automatic analysis of digital content
    - Automated manufacturing processes
    - Smart autonomous vehicles
    - Pedestrian detection for pre-crash safety systems

Basics

- Image analysis/ feature extraction
  - Abstraction of an image
    - Edge detection/ Corner detection
    - Shape based – Template matching
    - Image Motion – Optical flow
- Statistical analysis/ machine learning
  - To create a classifier
    - SVM/ Decision Tree/ Neural Nets

Person Detection

- Challenge lies in discriminating human form despite:
  - Wide range of poses
  - Cluttered backgrounds
  - Varying illumination
- Existing Detectors use:
  - Haar, HoG, SIFT, Parts based methods
- HoG based detectors [12]:
  - Simpler architecture
  - Single detection window
  - Significantly higher performance

Roadmap

- Literature Survey
- HoG + Gentleboost System
  - HoG
  - Gentleboost
  - Results
- Cascade of Rejectors
  - Integral HoG
- Future Directions

Current State

- State-of-art is neither sufficient nor satisfactory for many applications - frequent false positives

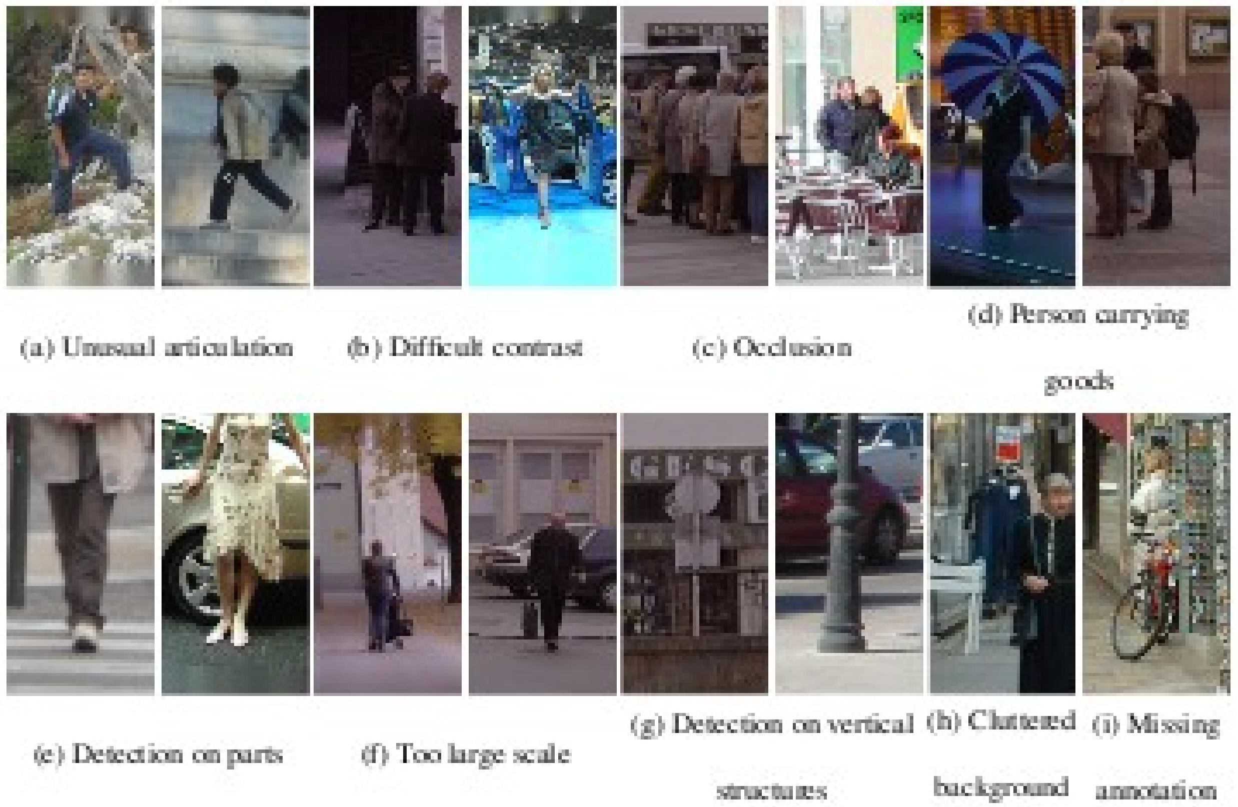


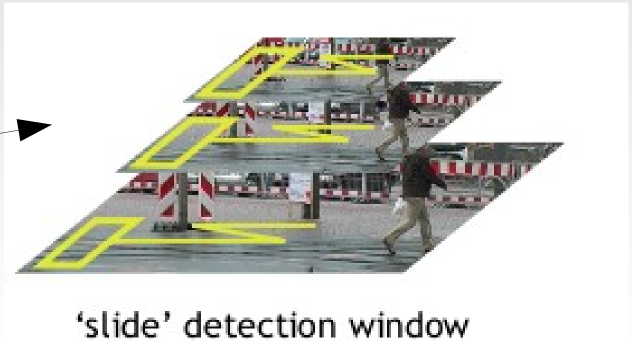
Figure 3. Missed recall (upper row) and false positive detections (lower row) at equal error rate

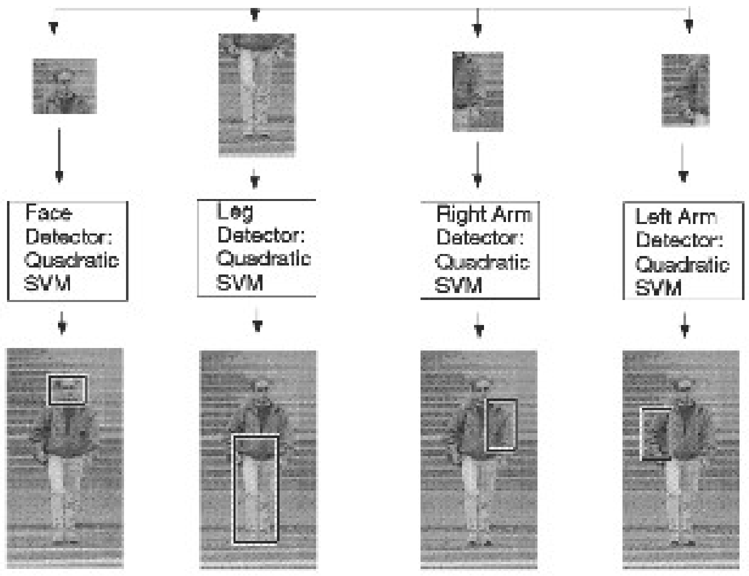
Fields of Research

- Detection in images – sliding window, parts based
- Improve false detection by detecting regions of interest (ROI)
  - Verify only ROI with classifier
  - Prior knowledge - object shape, size, color
  - Stereo - inverse perspective mapping (IMP)
  - Laser (Ladar system), Infrared
  - Motion cues - optical flow used commonly
    - Improves detection performance [7]
    - Impractical for real-time detection [6]
- Integration of detection and tracking

Recognition Approaches

- Two major types of approaches for people detection in still images:
  - Sliding Window
    - Features
    - Classifiers
  - Parts-based
    - Parts
    - Structure


'slide' detection window over all positions & scales



Face Detector: Quadratic SVM
Leg Detector: Quadratic SVM
Right Arm Detector: Quadratic SVM
Left Arm Detector: Quadratic SVM

Mohan, Papageorgiou, Poggio, '01

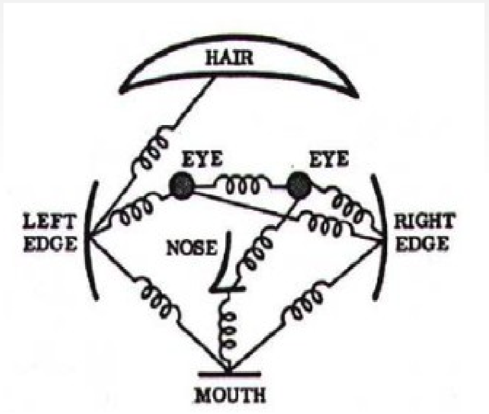
Sliding Window

- Feature + classifier
  - Features: Haar wavelets, HoG, Shape Context
  - Classifiers: SVM, Decision tree stumps + Adaboost framework
  - Multiple scales/position => computationally expensive
- HoG + Adaboost = Good Performance [2]*
- Multi-cue - Shape contexts + Haar better than HoG [2]*

*Tests done on a 4 different data sets including a challenging one compiled using a moving vehicle and on-board camera. Further more False Positives Per Image is used instead of False Positives Per Window

Parts-based Detection

- Part-based human body models
  - Low-level features or classifiers to model individual parts or limbs
  - Model topology of the human body to enable the accumulation of part evidence (configuration)
  - Parts-based people model can *outperform* sliding window based methods *in the presence of partial occlusion and significant articulations*



Combination?

- Very good object detector – Pascal Challenge '07

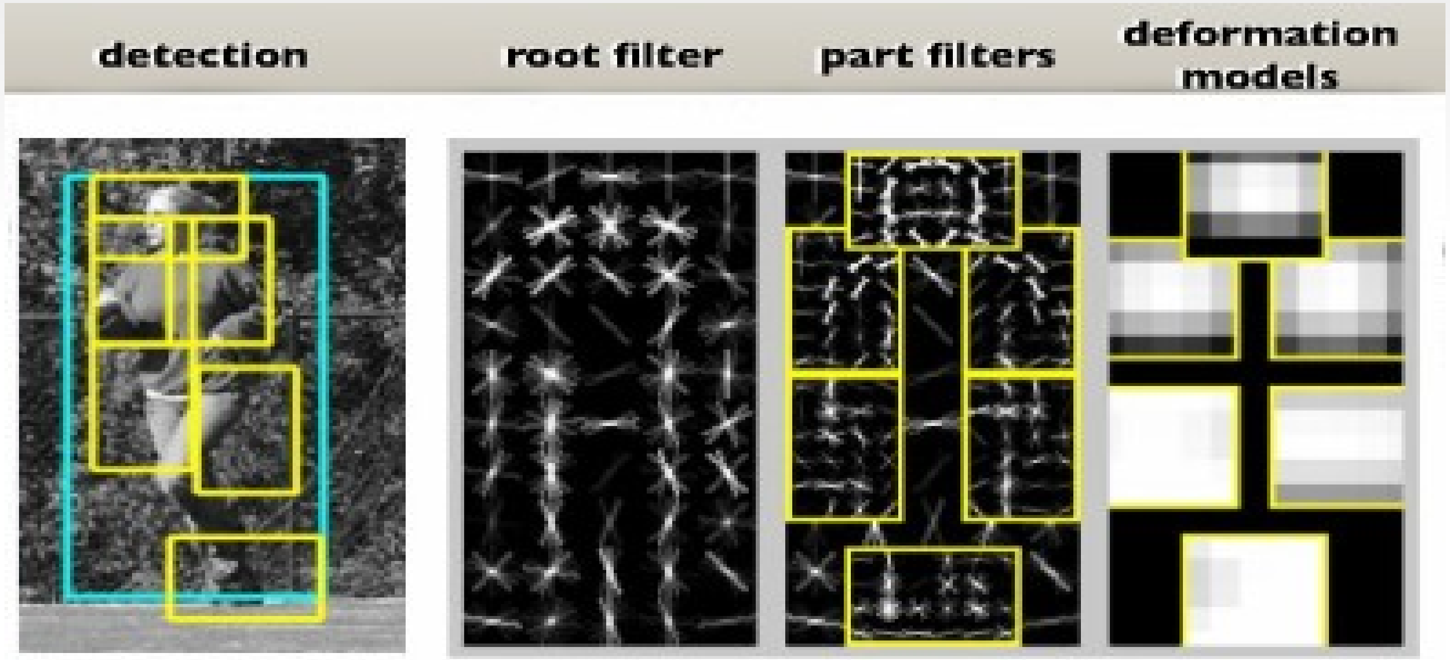


Figure 1. Example detection obtained with the person model. The model is defined by a coarse template, several higher resolution part templates and a spatial model for the location of each part.

Deformable Parts Model

- Adds a flexible part model to HoG
- Coarse global template, Higher resolution part templates
- Templates represent HoG features
- Latent variable SVM
- Seems to be effectively evaluating ROI through the root filter

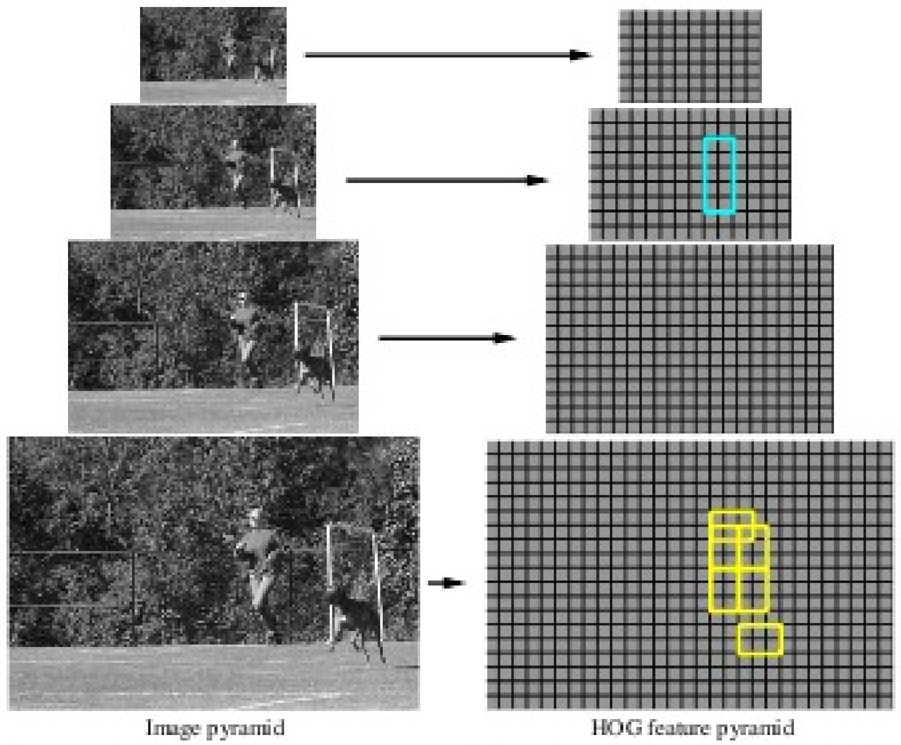
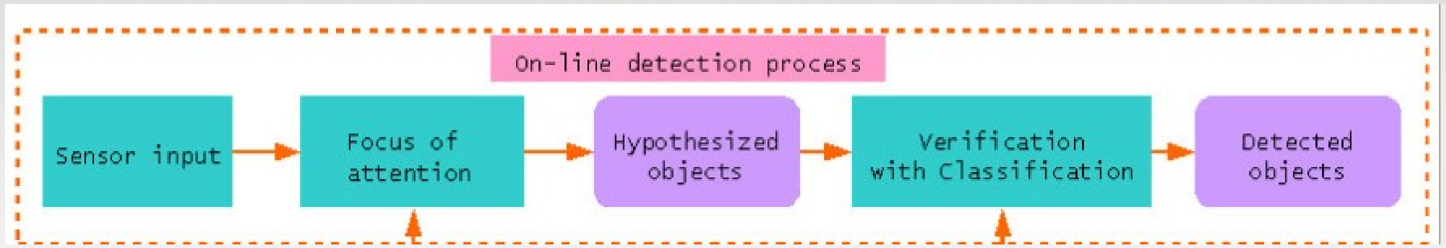


Figure 2. The HOG feature pyramid and an object hypothesis defined in terms of a placement of the root filter (near the top of the pyramid) and the part filters (near the bottom of the pyramid).

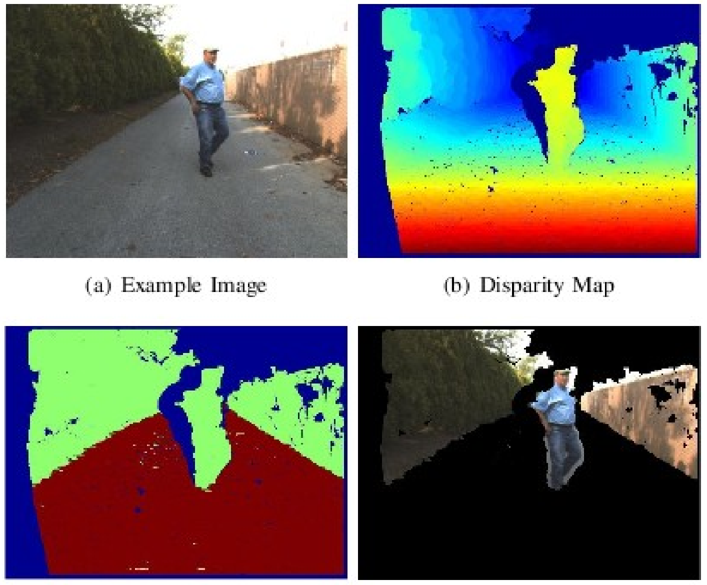
Fields of Research

- Detection in images – sliding window, parts based
- Improve false detection by detecting regions of interest (ROI)
  - Verify only ROI with classifier
  - Prior knowledge - object shape, size, color
  - Stereo - inverse perspective mapping (IMP)
  - Laser (Ladar system), Infrared
  - Motion cues - optical flow used commonly
    - Improves detection performance [7]
    - Impractical for real-time detection [6]
- Integration of detection and tracking

Using Stereo for ROI



- Stereo matching algorithm [6]
  - Remove ground plane from depth map


(a) Example Image
(b) Disparity Map
(c) Result of applying v-Disparity
(d) Image pixels of non-ground plane areas

Using Stereo for ROI

- Project remaining depth map to the XZ plane
  - Compute height and occupancy map

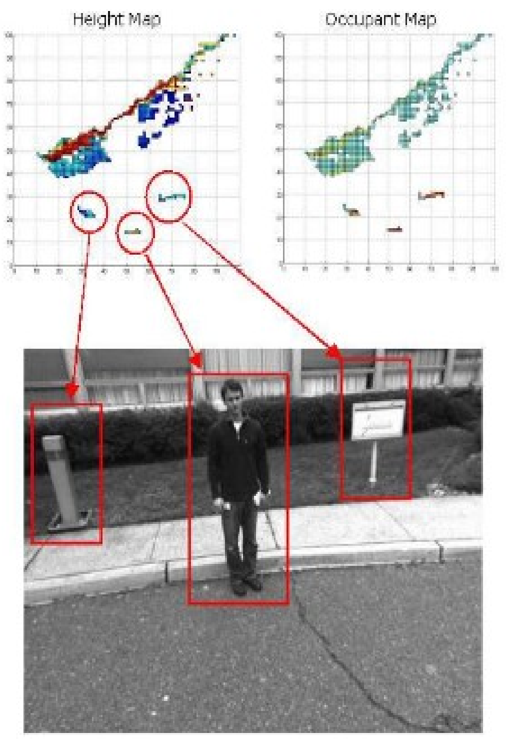


Fig. 3. Generate target hypothesis based on the height map and occupancy map.

Stereo/HoG details [6]

- Separate classifiers for front/rear view and side view
  - Apply in parallel and combine results
- Extended HoG: Incorporate spatial locality
  - Add one distance dimension to the angle dimension while binning
  - Distance is relative to the center of each subregion
  - Critical for highly structured objects like cars

Using Laser for ROI

- Visual + Laser [2, 5]
  - Use laser range information to constrain the search space of possible hypotheses
  - People walk on the floor => object scale is proportional to distance
  - Laser projected onto the image plane should hit the lower third (legs) of the detection window
- Stereo has wider field of vision
- Laser has longer range
- Combination?

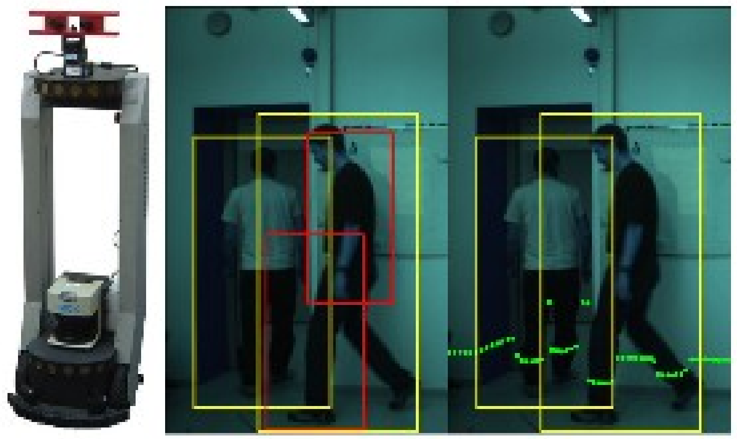


Figure 7. The PeopleBot Robot (a). Typical false positives from visual people detection (b). Rejection by simple range based constraints (c).

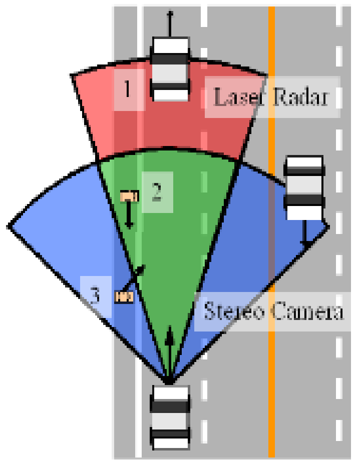


Fig. 3. Field of view

Multi-cue Pedestrian Detection

- Motion Cues -
  - Improves detection performance significantly [7]
  - Still have lots of room for improvement [2]
- Paper studies use of multiple cues and complementary features for detection
  - HoG, Haar, IMHwd (motion feature)
  - MPLBoost/MCBoost extension to Adaboost
    - Better perf. Static multi-viewpoint pedestrian detection
  - Improved learning and testing methods

Fields of Research

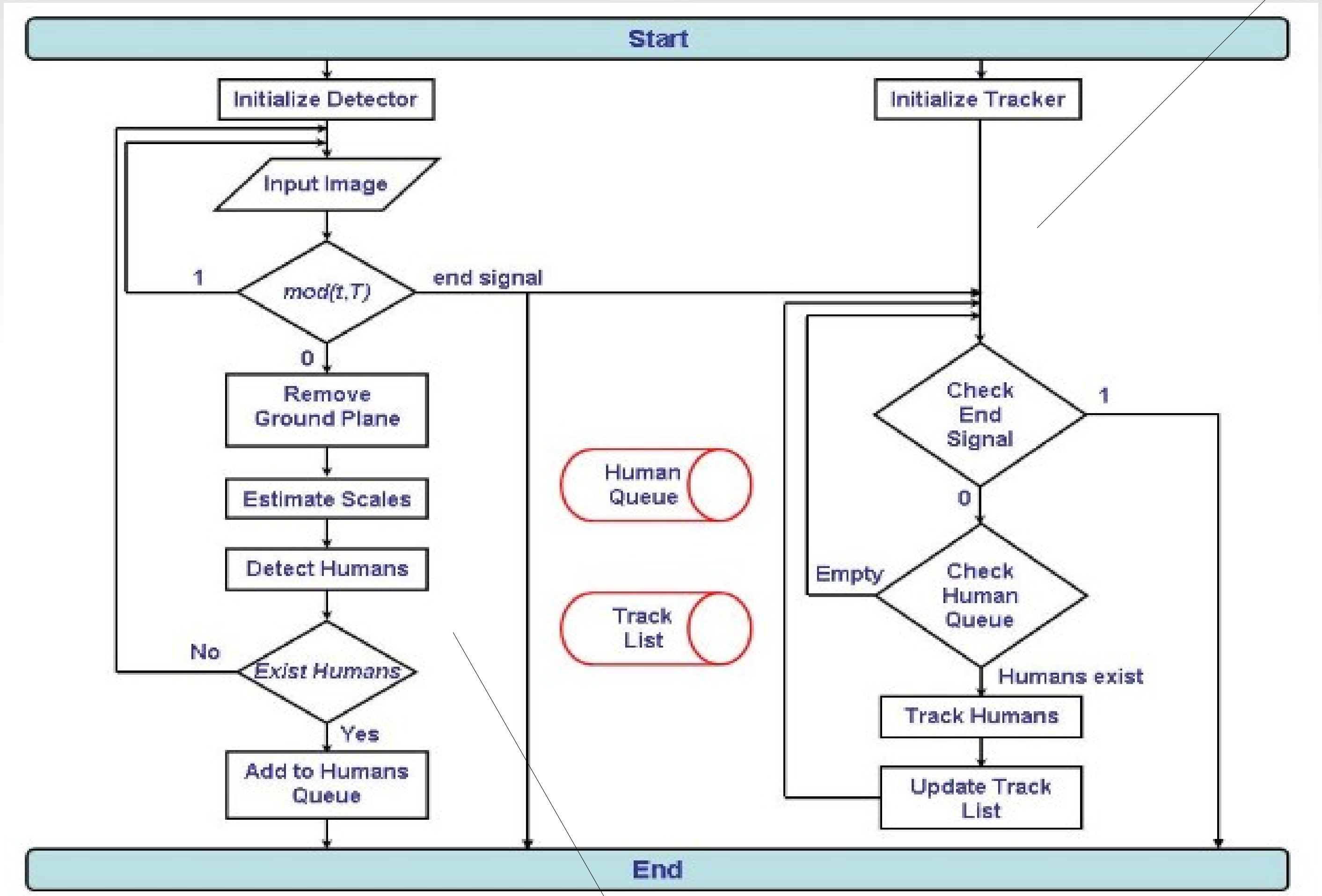
- Detection in images – sliding window, parts based
- Improve false detection by detecting regions of interest (ROI)
  - Verify only ROI with classifier
  - Prior knowledge - object shape, size, color
  - Stereo - inverse perspective mapping (IMP)
  - Laser (Ladar system), Infrared
  - Motion cues - optical flow used commonly
    - Improves detection performance [7]
    - Impractical for real-time detection [6]
- Integration of detection and tracking

Integrating Tracking

- Shape-based human detectors from still images
  - High false detection rate
    - Sensitivity to highly cluttered areas
  - Slow performance
- Motion-based human detectors
  - Tracking and analyzing motion patterns
    - Requires initialization
- Combining both is more effective [4]
  - Two concurrently running threads for detection and tracking that talk to each other

Detection and Tracking

Tracker runs at 30 fps on 320x480 images



Detector runs every 2 seconds

System Details [4]

- Builds on *integral HoG* implementation, Zhu et al.
  - Uses *Fisher Linear Discriminant* instead of SVM classifier in the cascaded Adaboost detector
    - Low complexity speeds up training and testing
  - Reduce false detections and speed up the process using *stereo cue* -
    - Estimate pixels that correspond to ground plane
      - v-Disparity algorithm
    - Cluster disparity values to determine search scale

Programming on GPU [8]

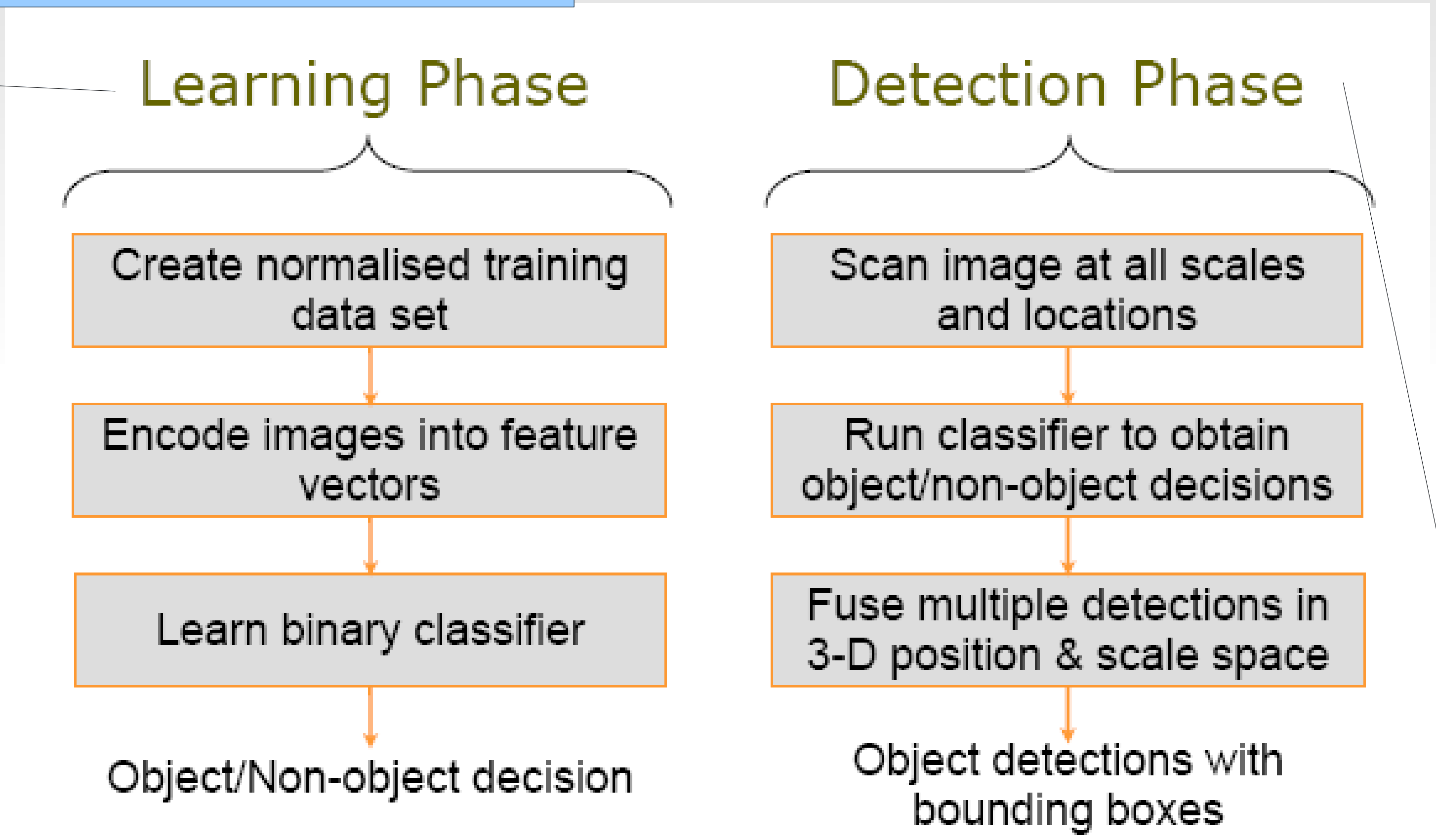
- Fast object class localization framework implemented on a data parallel architecture computer.
- Using this progamming model speeds up CPU only implementation by a factor of 34, making it *unnecessary to use early rejection cascades*.

Roadmap

- ✓ Literature Survey
- HoG + Gentleboost System
  - HoG
  - Gentleboost
  - Results
- Cascade of Rejectors
  - Integral HoG
- Future Directions

Overall Architecture

Size of learning images and detection window are both 64x128



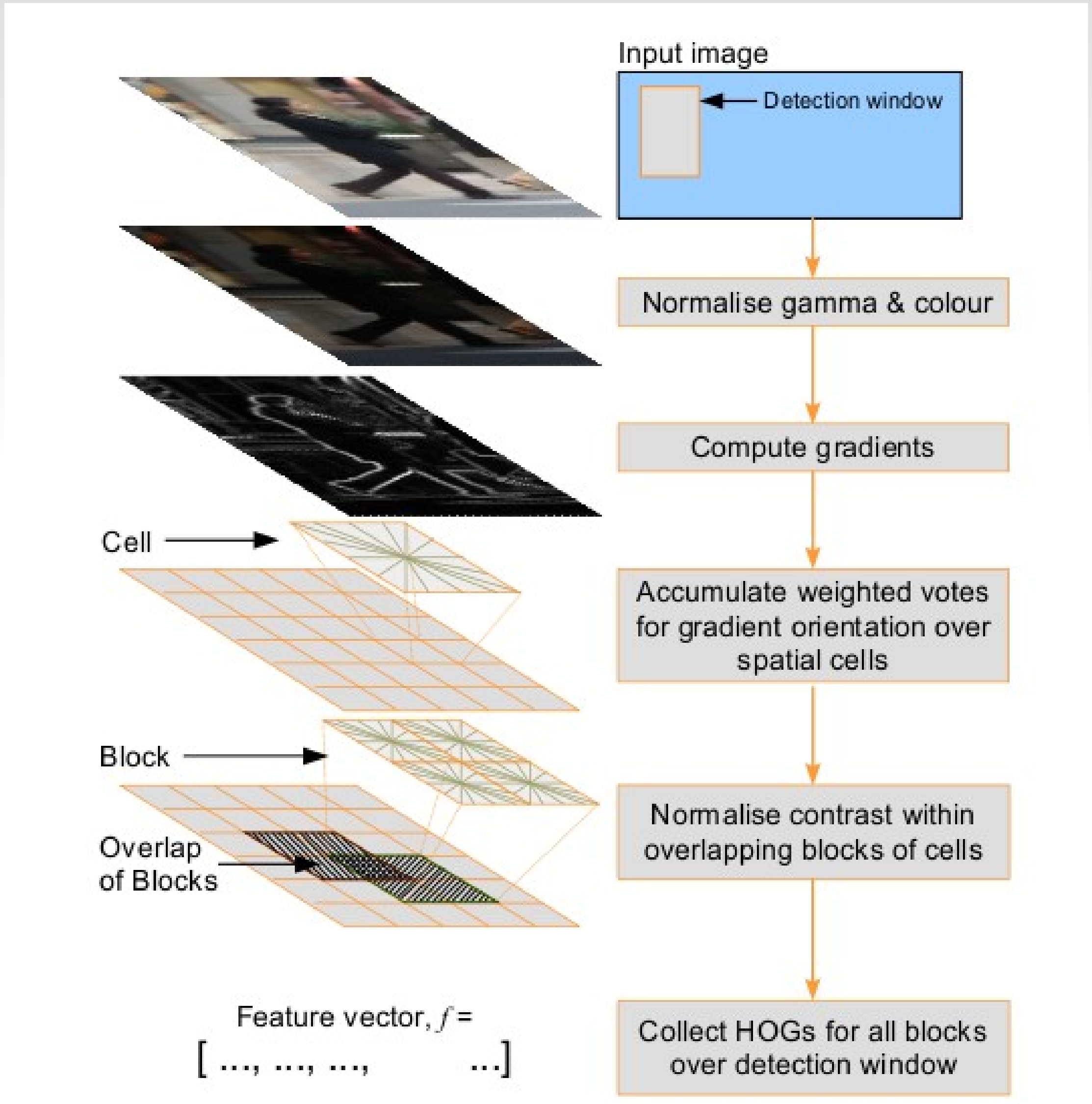
Learning Phase
Create normalised training data set
Encode images into feature vectors
Learn binary classifier
Object/Non-object decision
Detection Phase
Scan image at all scales and locations
Run classifier to obtain object/non-object decisions
Fuse multiple detections in 3-D position & scale space
Object detections with bounding boxes

Detection window is sill 64x128. It now scans through the image (any size) at all locations and multiple scales

HoG Features

- HoG Feature Descriptors (Histogram of Oriented Gradients) by Dalal and Triggs
  - Why is it effective?
    - Captures edge structure characteristic of local shapes in a manner that is *invariant to translations or rotations* smaller than orientation bin size (20 deg)
    - Coarse spatial sampling & fine orientation sampling allow for *invariance to pose* as long as they maintain upright position
    - Contrast-normalization improves *invariance to illumination*

HoG Extraction Overview


Input image
Detection window
Normalise gamma & colour
Compute gradients
Cell
Accumulate weighted votes for gradient orientation over spatial cells
Block
Overlap of Blocks
Normalise contrast within overlapping blocks of cells
Feature vector, f =
[..., ..., ..., ...]
Collect HOGs for all blocks over detection window

Creating HoG

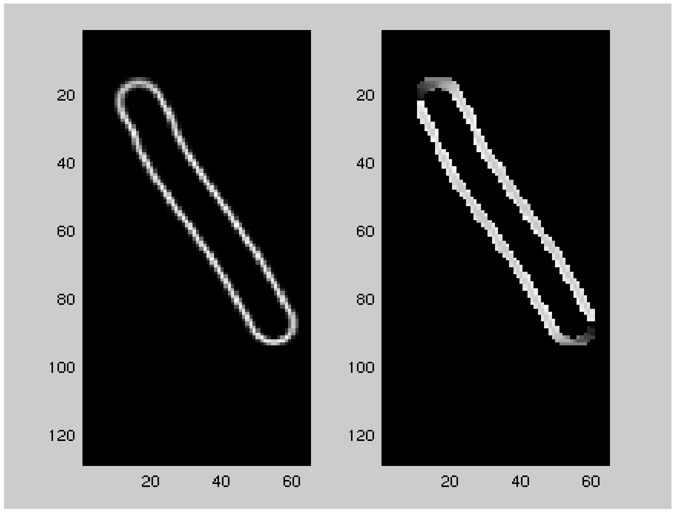
- Gradient Computation
  - 1-D mask [-1, 0, 1] => Img(x+1) – Img(x-1)
    - Run image through mask in x&y dimensions
  - Compute gradient magnitude and orientation =>
    - M(x,y) = $\sqrt{gx^2 + gy^2}$
    - O(x,y) = $tan^{-1}(\frac{gy}{gx})$
  - For RGB, do the above process for each channel.
    - Pick channel with largest magnitude as pixel's gradient vector

Implementation

- Normalize image values to between 0 and 1 for consistency – `Img = double(Img)/255`
- Use inbuilt conv2 function to run [-1 0 1] over image - `conv2(Img, [1,0,-1]) & conv2(Img, [1,0,-1]')`

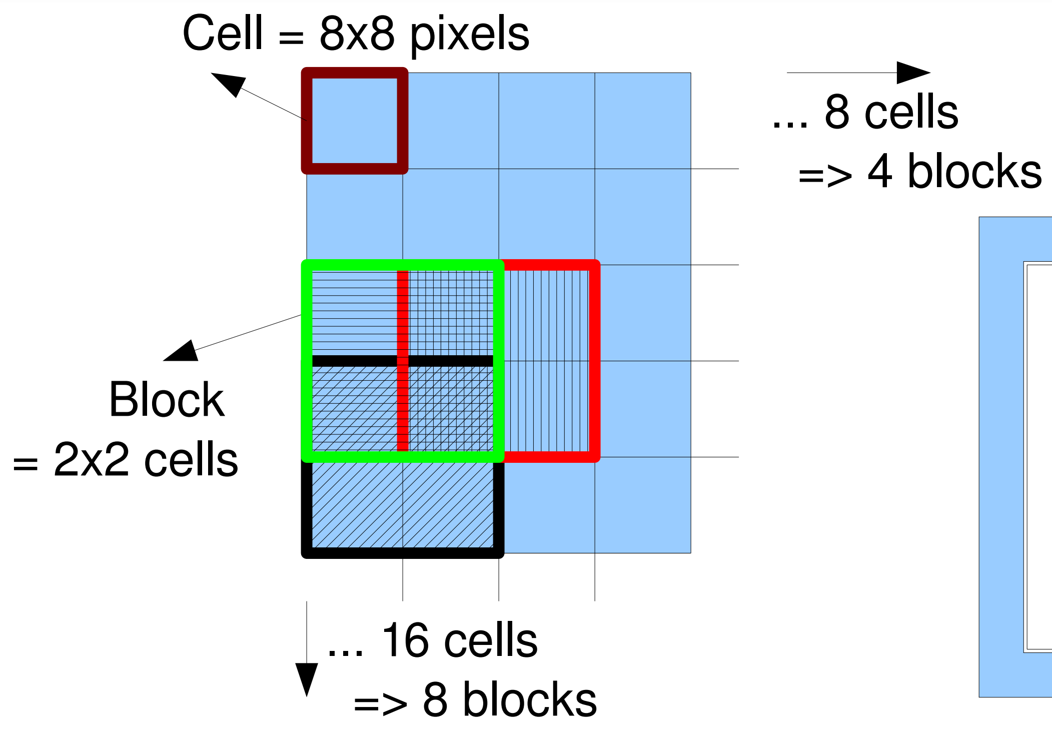
Test Image

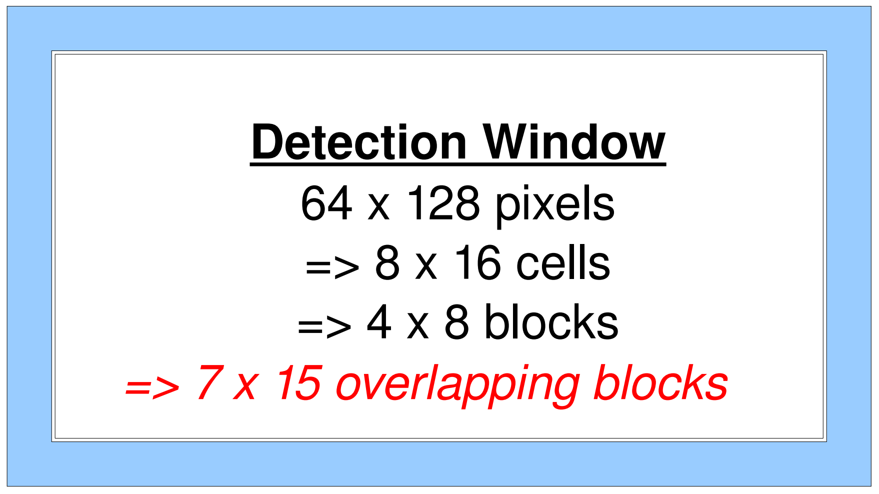
Gradient Magnitude and Orientation images



Dividing up Detection Window

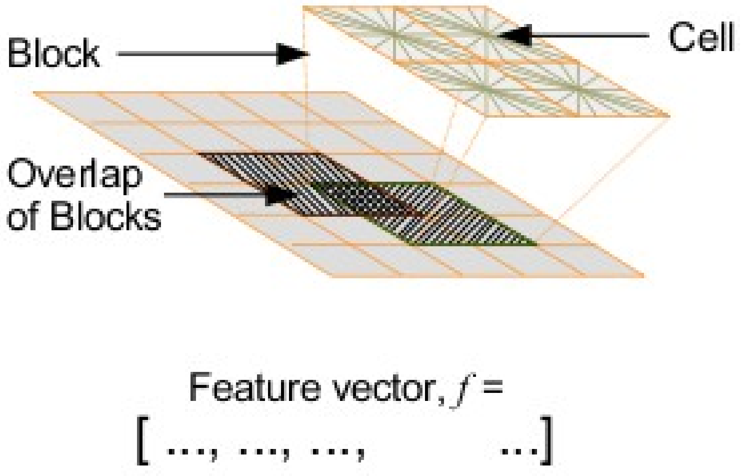
- Single Detection Window (64x128)
- Split into 7x15 overlapping blocks (stride length is 8 pixels)
- Each block is a 2x2 grid of cells that are 8x8 pixels in size => block size is 16x16 pixels





Dividing up Detection Window

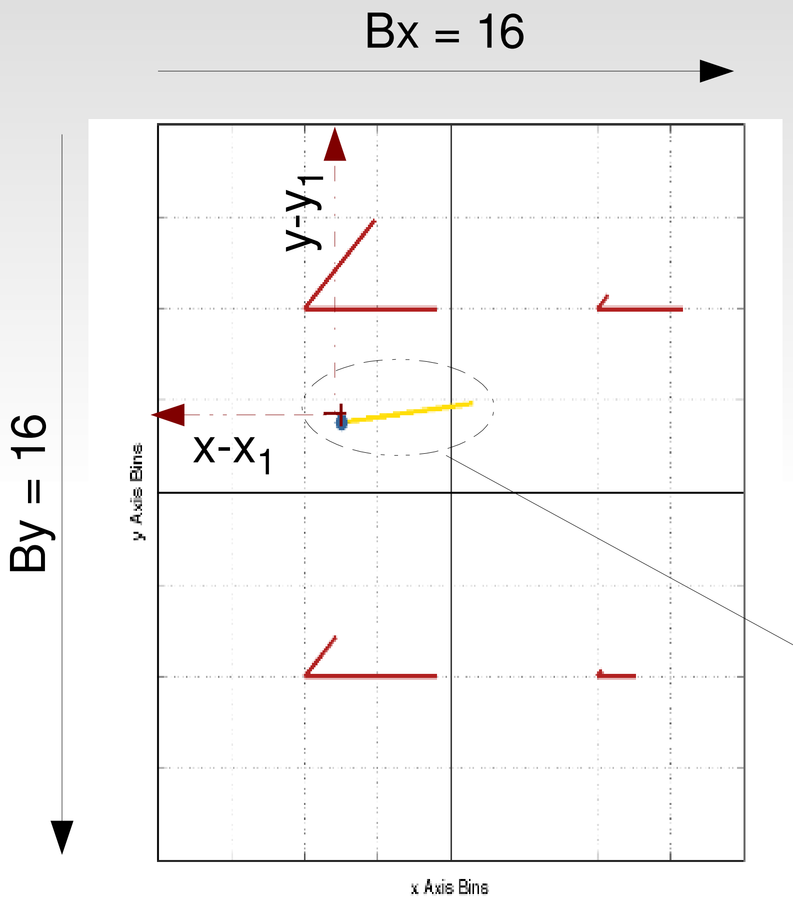
- Create a 9-bin histogram of the gradient orientations in each cell => each cell gives one 9-D feature vector
- To this concatenate the 9-D feature vectors of the other 3 cells of the block => *each block gives a 36-D feature vector*
- Each detection window thus gives a 3780-D vector (36x7x15)



Binning

- Spatial/Orientation Binning for each cell
  - Orientation bins centered at 10,30,... 170 degrees
    - Bin size is 20 deg
    - Using unsigned gradients
    - Wrap around at the edges
  - Tri-linear interpolation of pixel weight (gradient magnitude) to reduce aliasing and increase invariance to rotation
    - Bi-linear interpolation spatially in x and y to neighboring bin centers and linear interpolation in orientation dimension

Trilinear Interpolation



$$
\begin{aligned}
\mathbf{h}(x_1,y_1,z_1) &\leftarrow \mathbf{h}(x_1,y_1,z_1) + w\left(1-\frac{x-x_1}{b_x}\right)\left(1-\frac{y-y_1}{b_y}\right)\left(1-\frac{z-z_1}{b_z}\right)\\
\mathbf{h}(x_1,y_1,z_2) &\leftarrow \mathbf{h}(x_1,y_1,z_2) + w\left(1-\frac{x-x_1}{b_x}\right)\left(1-\frac{y-y_1}{b_y}\right)\left(\frac{z-z_1}{b_z}\right)\\
\mathbf{h}(x_1,y_2,z_1) &\leftarrow \mathbf{h}(x_1,y_2,z_1) + w\left(1-\frac{x-x_1}{b_x}\right)\left(\frac{y-y_1}{b_y}\right)\left(1-\frac{z-z_1}{b_z}\right)\\
\mathbf{h}(x_2,y_1,z_1) &\leftarrow \mathbf{h}(x_2,y_1,z_1) + w\left(\frac{x-x_1}{b_x}\right)\left(1-\frac{y-y_1}{b_y}\right)\left(1-\frac{z-z_1}{b_z}\right)\\
\mathbf{h}(x_1,y_2,z_2) &\leftarrow \mathbf{h}(x_1,y_2,z_2) + w\left(1-\frac{x-x_1}{b_x}\right)\left(\frac{y-y_1}{b_y}\right)\left(\frac{z-z_1}{b_z}\right)\\
\mathbf{h}(x_2,y_1,z_2) &\leftarrow \mathbf{h}(x_2,y_1,z_2) + w\left(\frac{x-x_1}{b_x}\right)\left(1-\frac{y-y_1}{b_y}\right)\left(\frac{z-z_1}{b_z}\right)\\
\mathbf{h}(x_2,y_2,z_1) &\leftarrow \mathbf{h}(x_2,y_2,z_1) + w\left(\frac{x-x_1}{b_x}\right)\left(\frac{y-y_1}{b_y}\right)\left(1-\frac{z-z_1}{b_z}\right)\\
\mathbf{h}(x_2,y_2,z_2) &\leftarrow \mathbf{h}(x_2,y_2,z_2) + w\left(\frac{x-x_1}{b_x}\right)\left(\frac{y-y_1}{b_y}\right)\left(\frac{z-z_1}{b_z}\right)
\end{aligned}
$$

Represents magnitude and angle of pixel at x and y

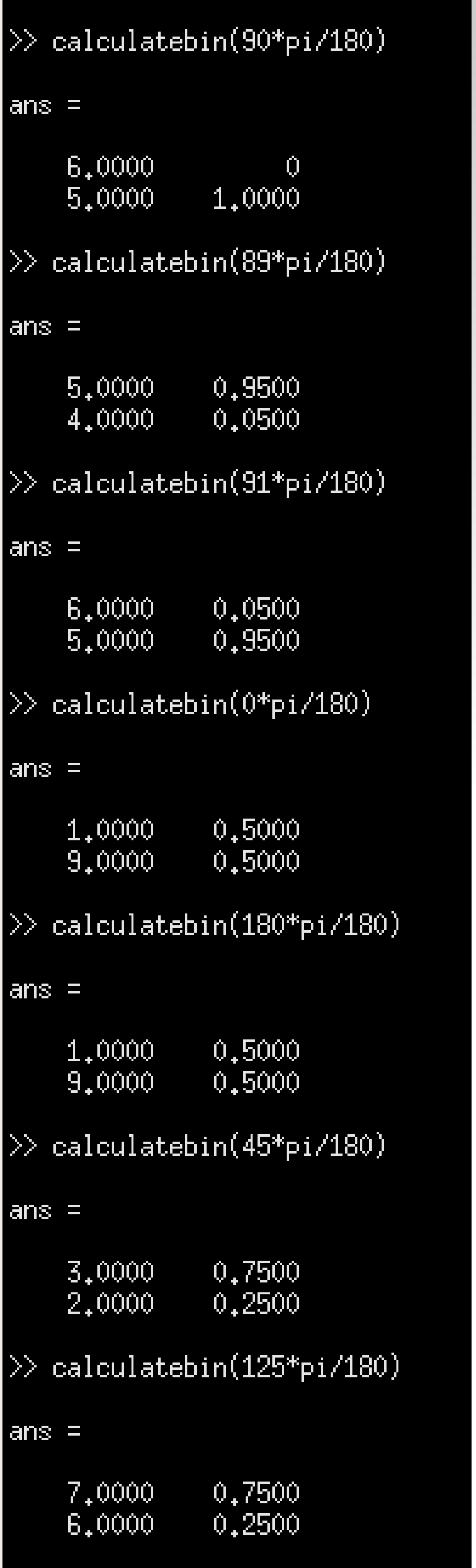
Angle z is *linearly* interpolated between neighboring *orientation bins* (z1 and z2)

<u>In essence, pixel magnitude is spread into bins z1 and z2 of each of the 4 cells in a block</u>

Weighting *bilinearly* in x and y into neighboring *spatial bins*

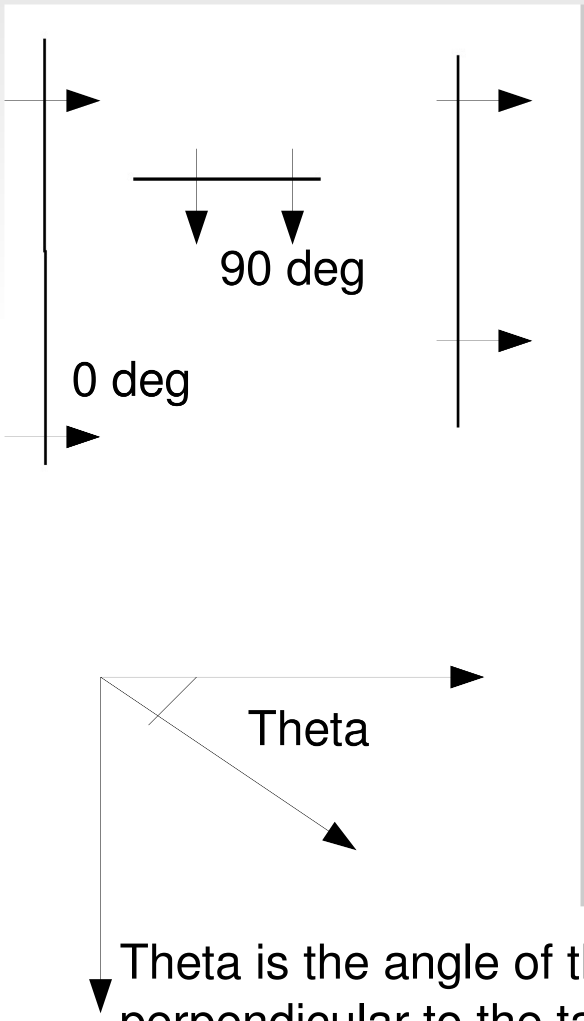
*bins are wrapped around in the orientation dimension

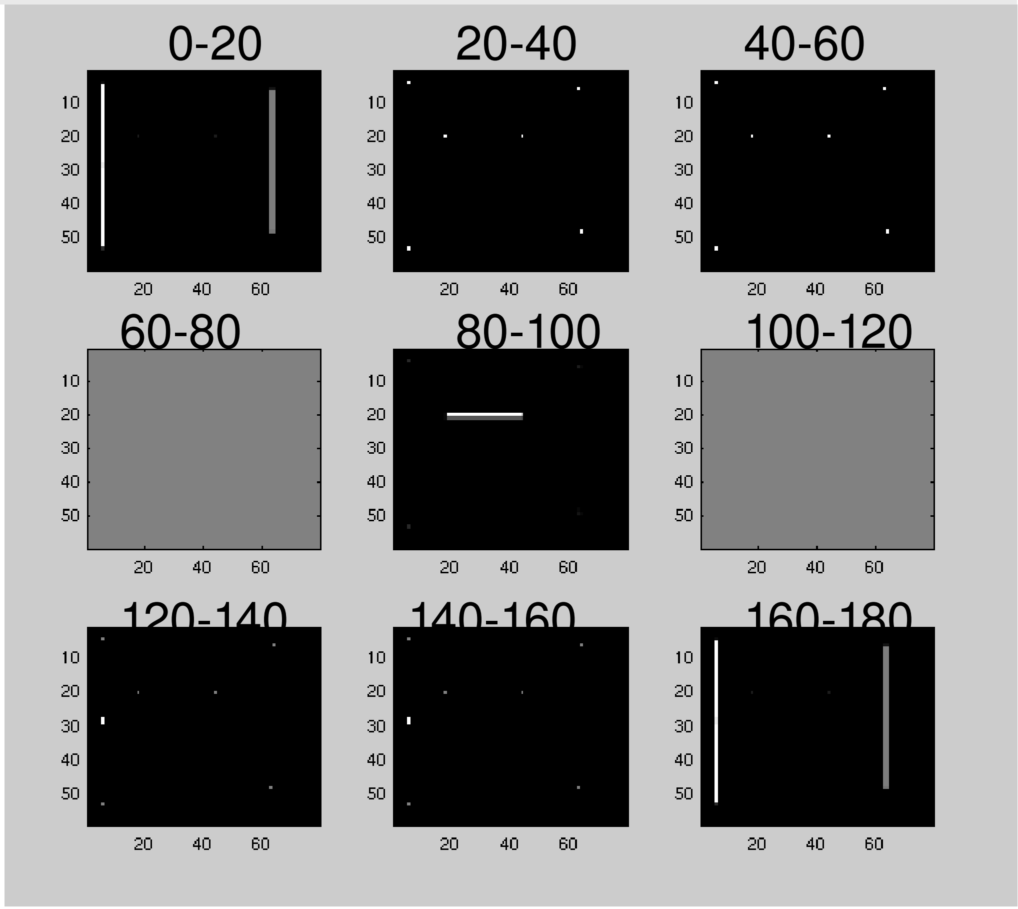
Binning Implementation



```
>> calculatebin(90*pi/180)

ans =

    6.0000         0
    5.0000    1.0000

>> calculatebin(89*pi/180)

ans =

    5.0000    0.9500
    4.0000    0.0500

>> calculatebin(91*pi/180)

ans =

    6.0000    0.0500
    5.0000    0.9500

>> calculatebin(0*pi/180)

ans =

    1.0000    0.5000
    9.0000    0.5000

>> calculatebin(180*pi/180)

ans =

    1.0000    0.5000
    9.0000    0.5000

>> calculatebin(45*pi/180)

ans =

    3.0000    0.7500
    2.0000    0.2500

>> calculatebin(125*pi/180)

ans =

    7.0000    0.7500
    6.0000    0.2500
```

Test Image



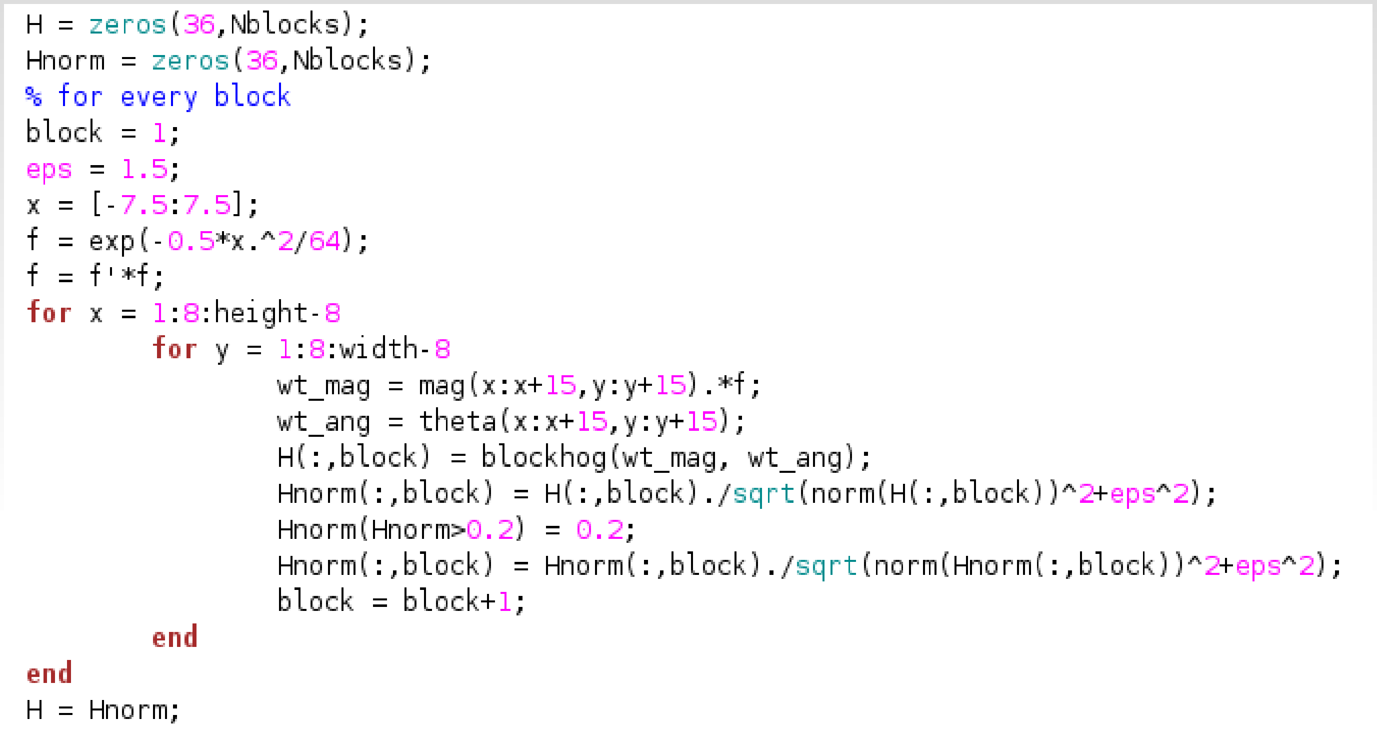


Theta is the angle of the line drawn perpendicular to the tangent

Block Level

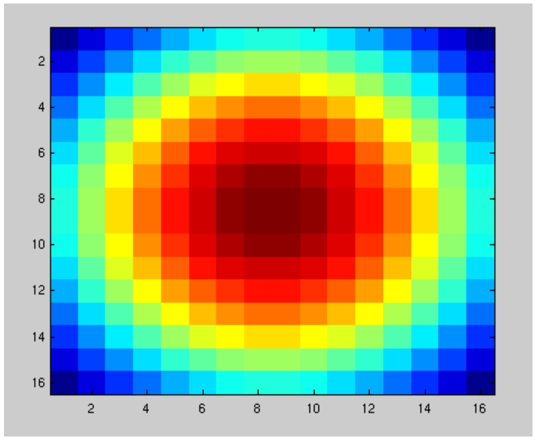
- Down-weight pixels near the edges of the block before accumulating votes in the cells
  - Gaussian Spatial Window with sigma = 8
  - Reduce impact of shifting on the output histogram
- Block Normalization – *L2-Hys* scheme:
  - To increase invariance to illumination, contrast
  - *L2-norm* followed by clipping of v to 0.2, and then re-normalizing again
    - *L2-norm*: $v \rightarrow v / \sqrt{\|v\|_2^2 + \varepsilon^2}$
    - v: unnormalized descriptor vector

Block Level - Matlab



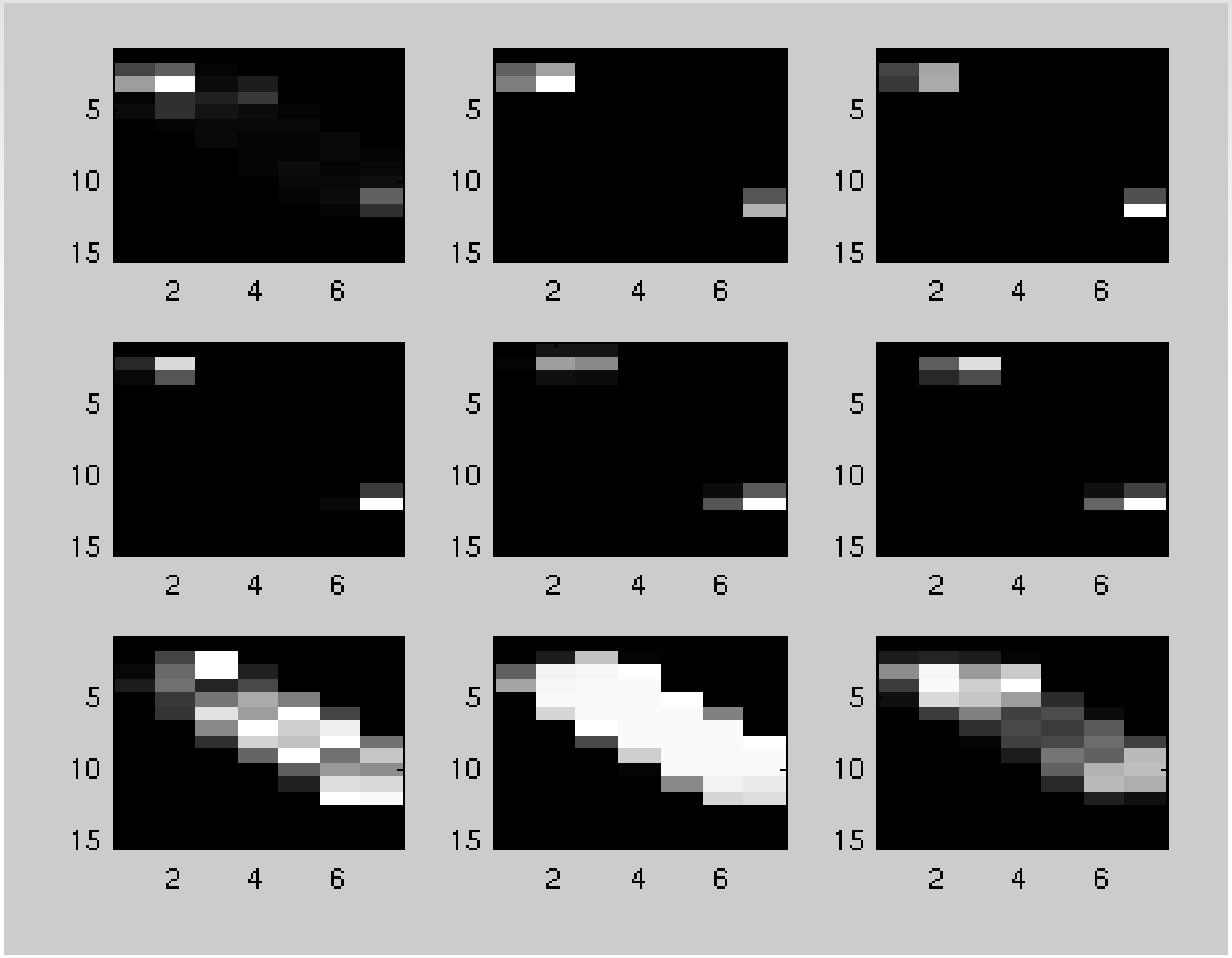
```
H = zeros(36,Nblocks);
Hnorm = zeros(36,Nblocks);
% for every block
block = 1;
eps = 1.5;
x = [-7.5:7.5];
f = exp(-0.5*x.^2/64);
f = f'*f;
for x = 1:8:height-8
        for y = 1:8:width-8
                wt_mag = mag(x:x+15,y:y+15).*f;
                wt_ang = theta(x:x+15,y:y+15);
                H(:,block) = blockhog(wt_mag, wt_ang);
                Hnorm(:,block) = H(:,block)./sqrt(norm(H(:,block))^2+eps^2);
                Hnorm(Hnorm>0.2) = 0.2;
                Hnorm(:,block) = Hnorm(:,block)./sqrt(norm(Hnorm(:,block))^2+eps^2);
                block = block+1;
        end
end
H = Hnorm;
```

f =

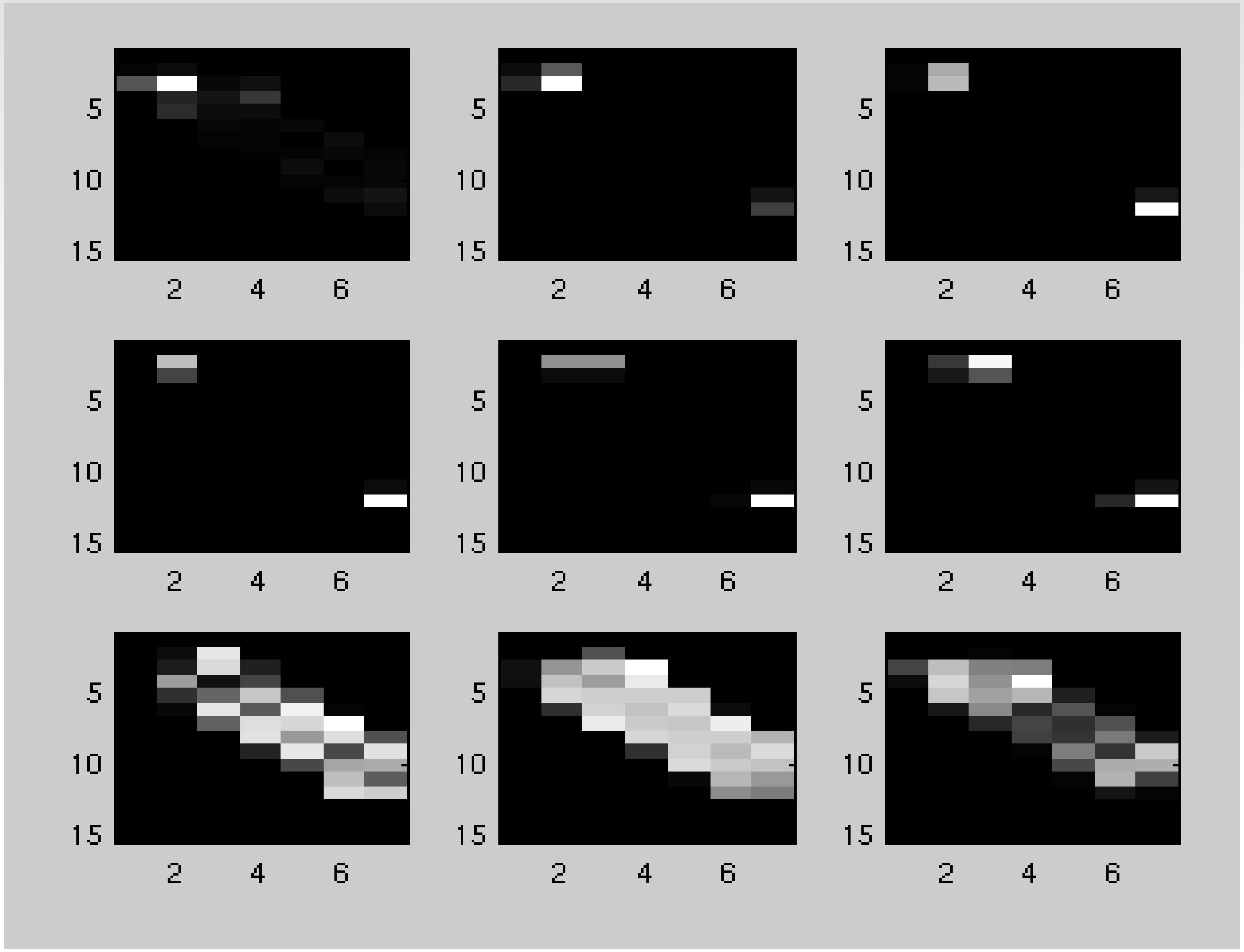


Gaussian Mask
with sigma = 8

Edge Images



Dalal's Source Code



GentleBoost Classifier

- Training -

Dataset of Images		Labels	Weights
Img 1	3780 features	1	1/N
Img 2	3780 features	-1	1/N
...	...		
Img N	3780 features	-1	1/N

Initial Weights ... These will be updated every time a weak classifier is trained

Person OR No Person

Every weak-learner is trained on these set of images -> *a series of weak models are created*

*Weak-learner used is an 8-node decision tree

After a particular weak learner has been trained, it is tested on the same dataset it was trained on. This results in a set of predictions from which an error function can be defined. This function is used to update the weights that will be used for training next weak classifier.

GentleBoost

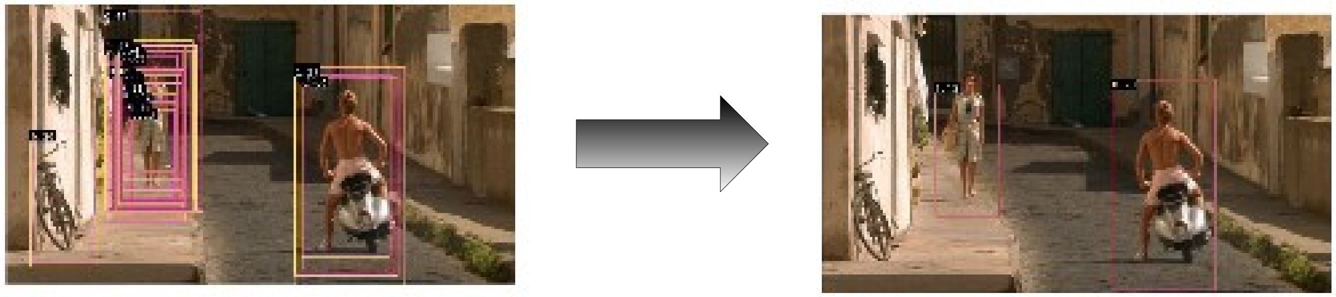
- Key idea: difficult images => higher weights
- Subsequent weak classifiers are tweaked to focus on these images – this is *boosting*
- Strong classifier is a combination of the predictions (and the confidence with which these prediction were made) as obtained from a series of weak classifiers
- GentleBoost is resistant to noisy data (because of how it updates weights)

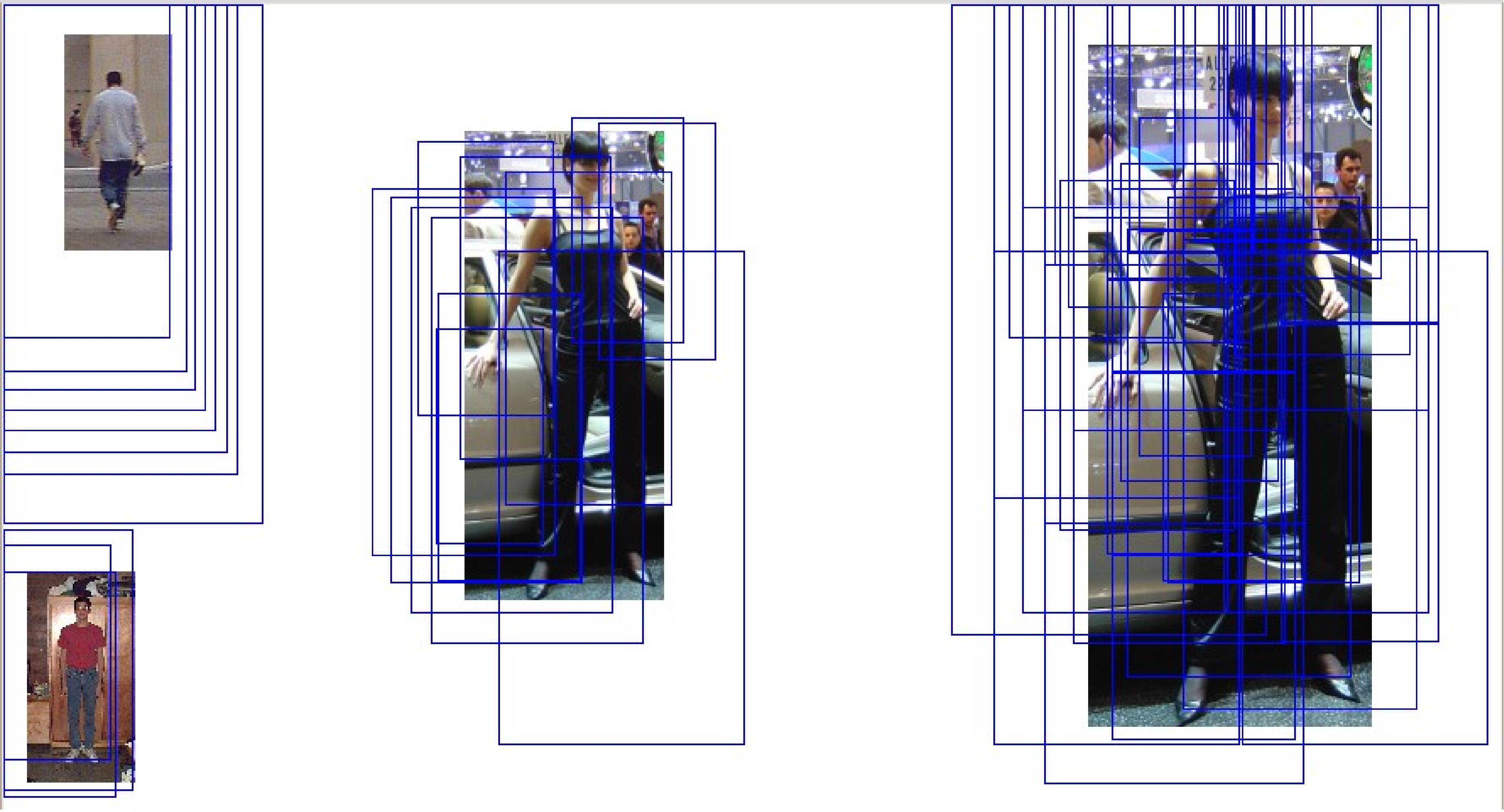
Internal Motion Histogram

- Possible additional step...
- Internal Motion Histograms are basically Oriented Histograms of Differential Optical Flow (motion-based detectors)
  - Uses two consecutive images to compute optical flow
    - Differential optical flow cancels out effects of camera motion
  - **It is combined with HoG (appearance descriptors)**

Results

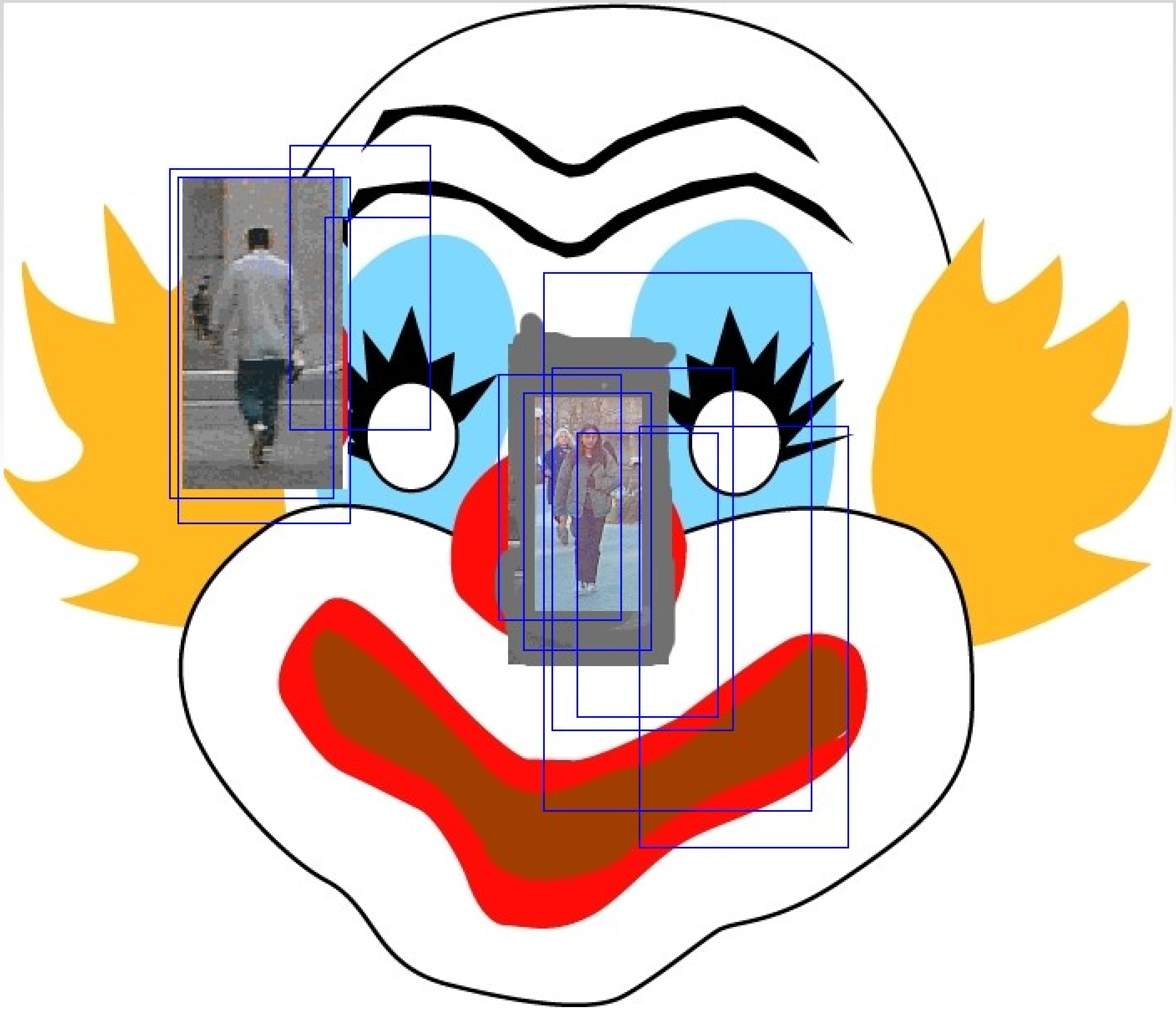
Note: Without non-maxima suppression





*Detection Window Stride = 64 Pixels

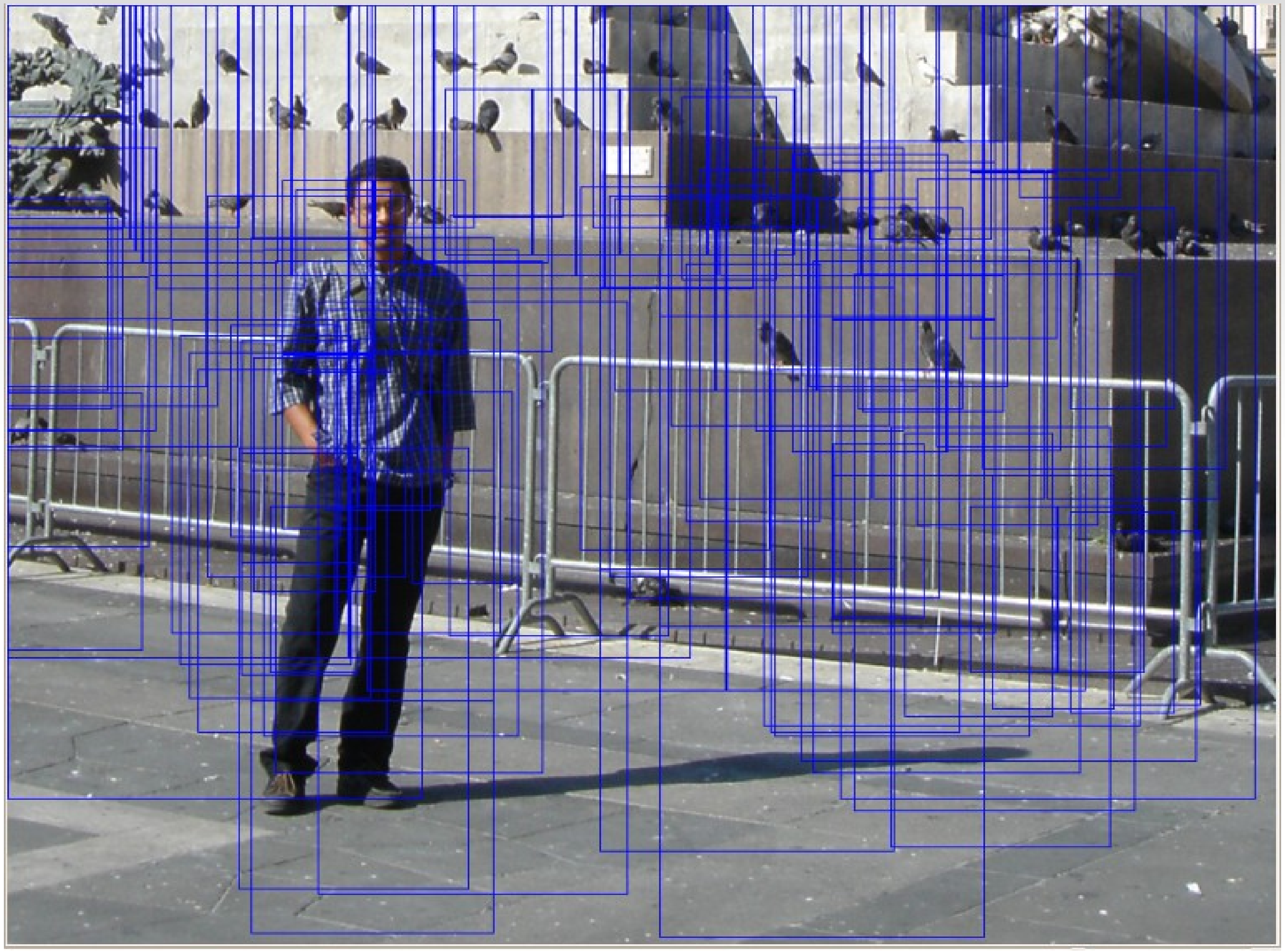
*Detection Window running at all possible scales and locations of the original image



*Detection Window Stride = 64 Pixels

*Detection Window running at all possible scales and locations of the original image

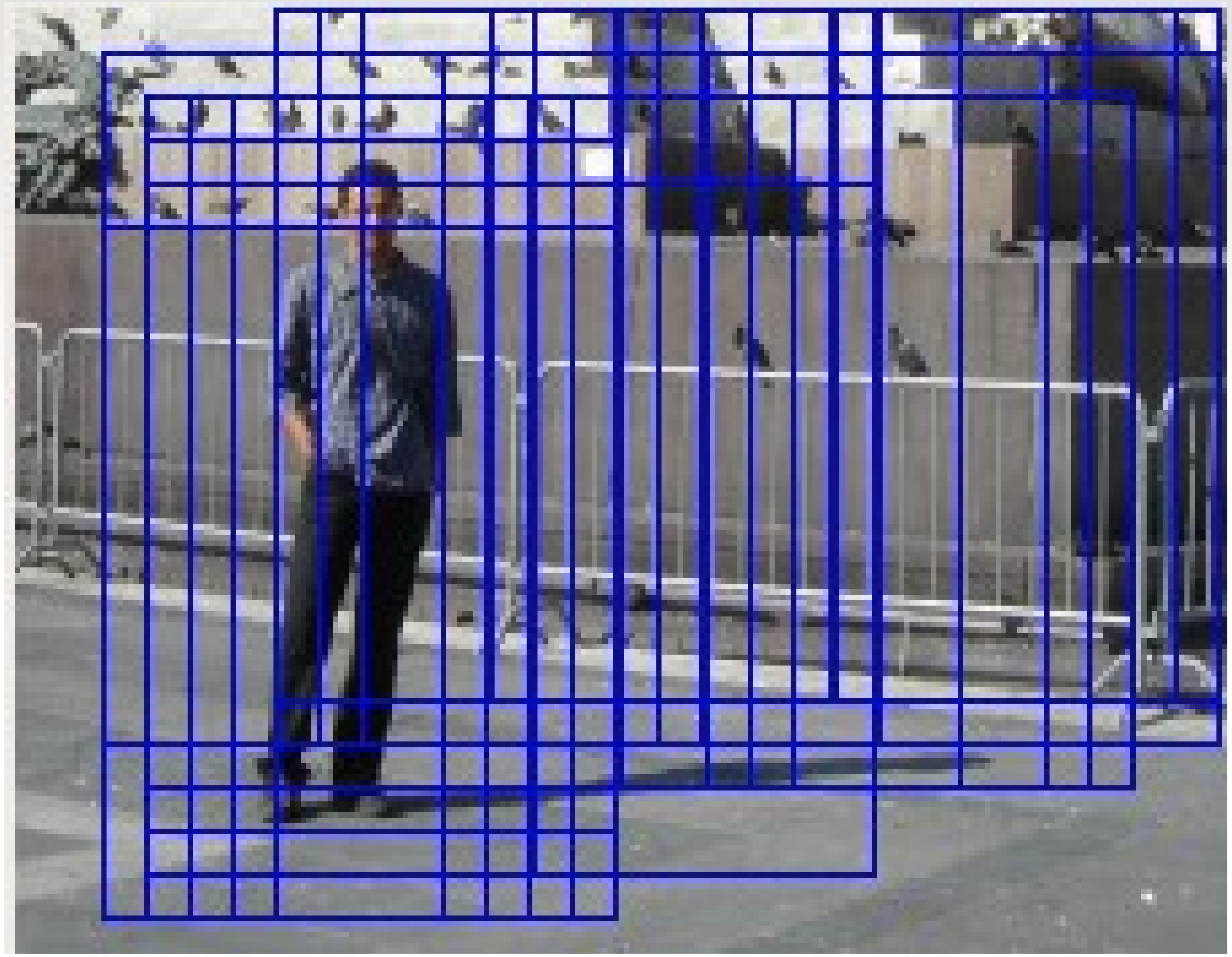
Image was scaled 32 times



*Detection Window Stride = 64 Pixels
*Detection Window running at all possible scales and locations of the original image

Scaled three times

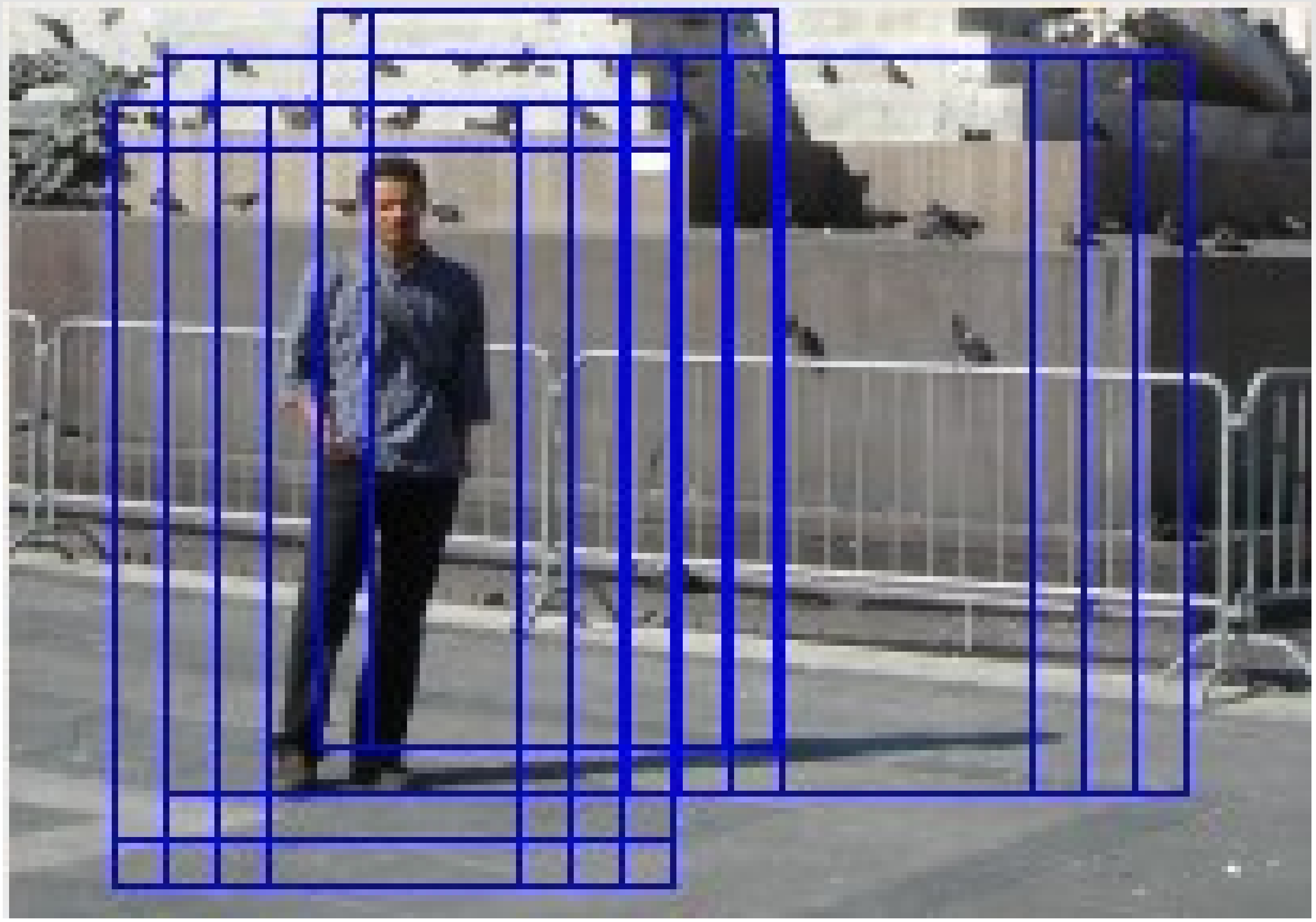
*Detection Window Stride = 64 Pixels



*Detection window was run on only 3 scales of the image
That is why rectangles are of only 3 sizes

No Scaling

*Detection Window Stride = 64 Pixels



*Detection window was run on only one scales of the image
That is why rectangles are all of one size

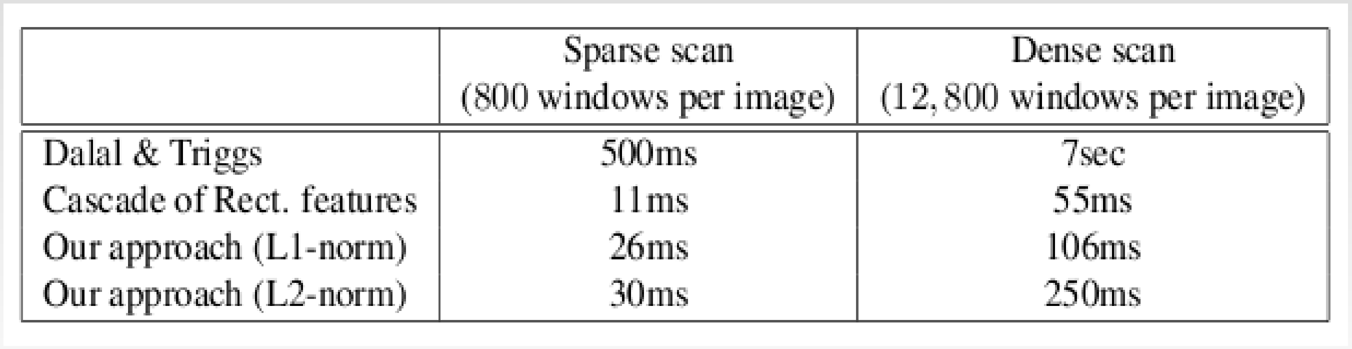
Roadmap

- ✓ Literature Survey
- ✓ HoG + Gentleboost System
  - ✓ HoG
  - ✓ Gentleboost
  - ✓ Results
- Cascade of Rejectors
  - Integral HoG
- Future Directions

Motivation

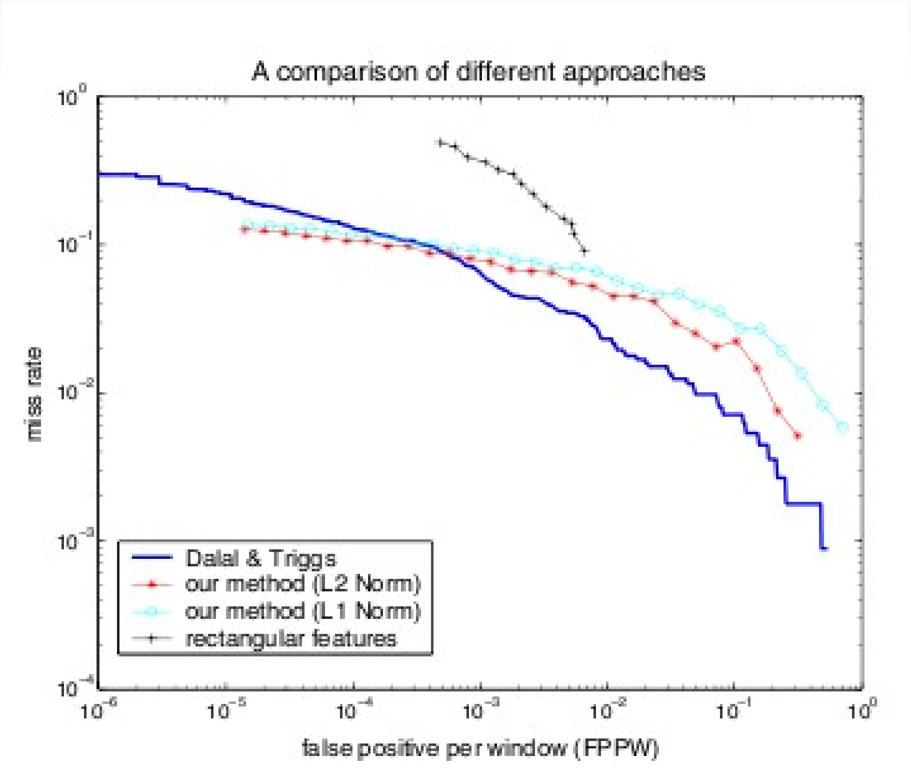
- Dalal's method can only process 320x240 images at 1 FPS (roughly 800 detection windows per image)
- Using the cascade of rejectors approach results in near *real-time human detection* (Zhu et al.) [9]
  - Similar to the one proposed by Viola and Jones for face detection
  - Rejects detection windows by evaluating 4.7 blocks on average
- For faster computation we use “integral images” of the histograms

Comparison



	Sparse scan (800 windows per image)	Dense scan (12, 800 windows per image)
Dalal & Triggs	500ms	7sec
Cascade of Rect. features	11ms	55ms
Our approach (L1-norm)	26ms	106ms
Our approach (L2-norm)	30ms	250ms

240x320 images



Gaussian down-weighting and trilinear interpolation don't fit in the integral image approach. Despite this they get comparable results to Dalal and outperform them in speed.

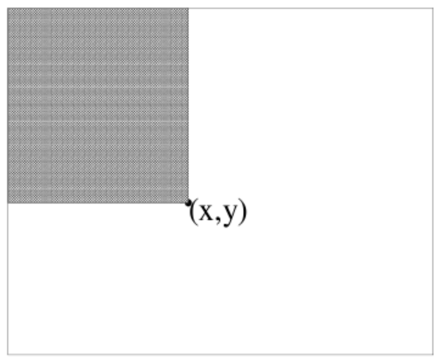
Variable Blocks

- Dalal used fixed blocks of 16x16 pixels
  - Small size not informative enough
- Zhu uses blocks of different sizes, location and aspect ration
  - Size range from 12x12 to 64x128
  - Aspect ratio – 1:1, 2:1, 1:2
  - Step size – {4,6,8} depending on block size
  - Results in 5031 blocks compared to Dalal's 105

Integral Images

- To efficiently compute the HoG of blocks chosen by the Adaboost-based feature selection algorithm
- Compute integral image using [10]:

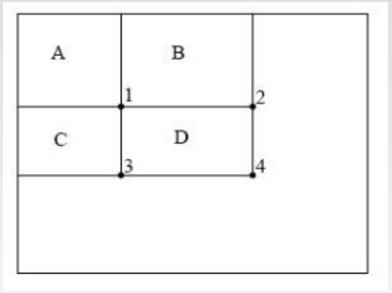
$$\begin{aligned} s(x,y) &= s(x,y-1) + i(x,y) \\ ii(x,y) &= ii(x-1,y) + s(x,y) \end{aligned}$$



  - where s(x,y) is the cumulative row sum and ii(x,y) is the integral image

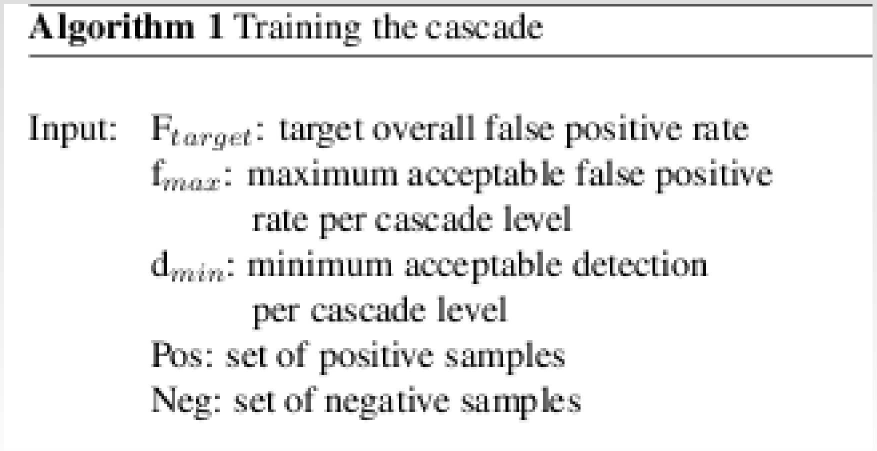
Integral Images

- Sum within D is 4+1-(2+3)
  - 4 image access operation



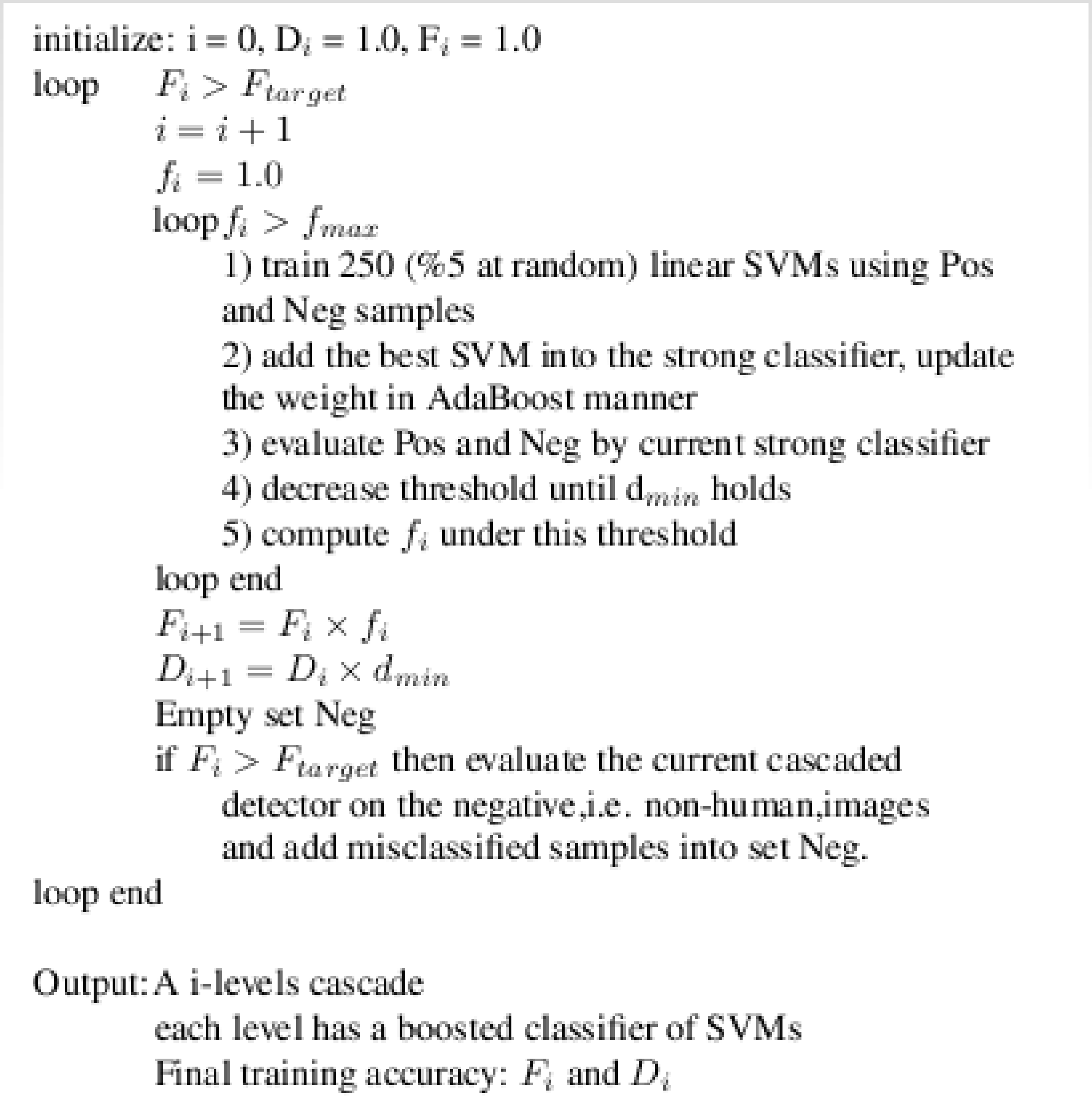
- Create integral images for each of the bins of the histogram => results in 9 integral images
  - 9x4 image access operations
- Now, to calculate the HoG of block D, it needs to be divided into four sub-regions, each of which gives a 9-D vector that are concatenated to result in a 36-D HoG for block D

Training the Cascade



Algorithm 1 Training the cascade

```
Input:  $F_{target}$: target overall false positive rate
        $f_{max}$: maximum acceptable false positive
                rate per cascade level
        $d_{min}$: minimum acceptable detection
                per cascade level
        Pos: set of positive samples
        Neg: set of negative samples
```

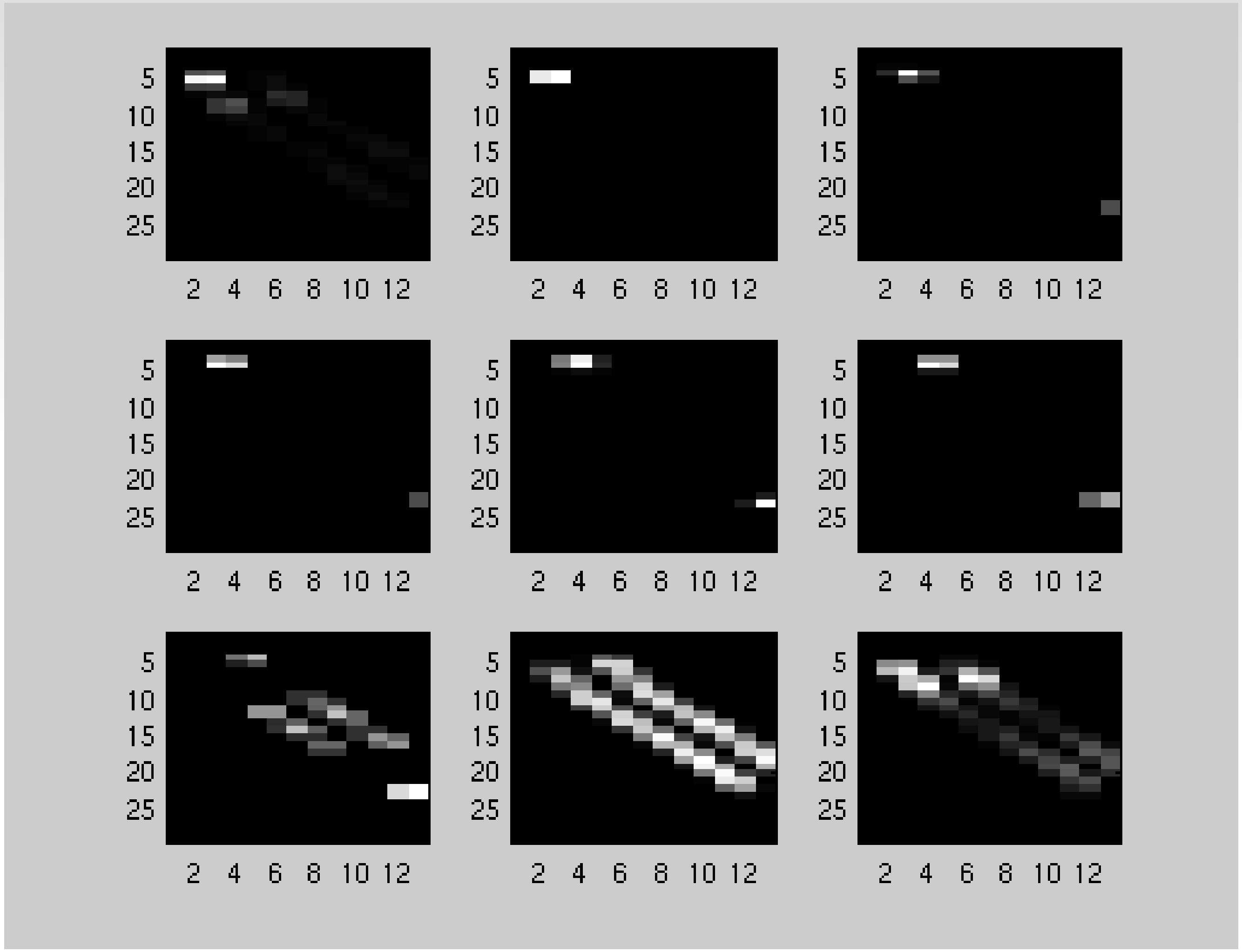


```
initialize: i = 0, $D_i$ = 1.0, $F_i$ = 1.0
loop    $F_i > F_{target}$
        $i = i + 1$
        $f_i = 1.0$
        loop $f_i > f_{max}$
            1) train 250 (%5 at random) linear SVMs using Pos
            and Neg samples
            2) add the best SVM into the strong classifier, update
            the weight in AdaBoost manner
            3) evaluate Pos and Neg by current strong classifier
            4) decrease threshold until $d_{min}$ holds
            5) compute $f_i$ under this threshold
        loop end
        $F_{i+1} = F_i \times f_i$
        $D_{i+1} = D_i \times d_{min}$
        Empty set Neg
        if $F_i > F_{target}$ then evaluate the current cascaded
            detector on the negative,i.e. non-human,images
            and add misclassified samples into set Neg.
loop end

Output: A i-levels cascade
        each level has a boosted classifier of SVMs
        Final training accuracy: $F_i$ and $D_i$
```

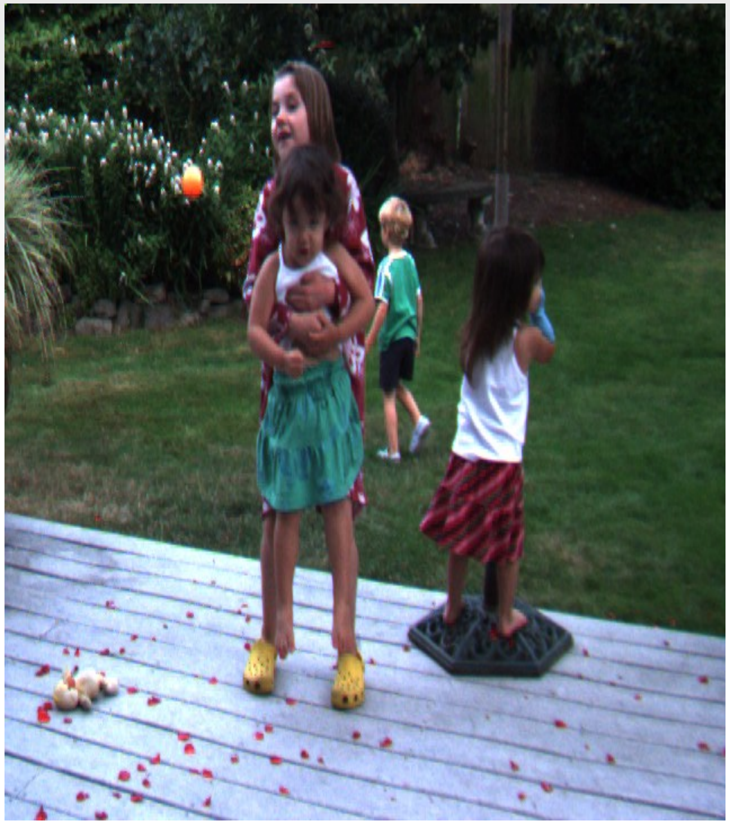
250 blocks are chosen at random at each level of the cascade and are trained on 250 linear SVMs

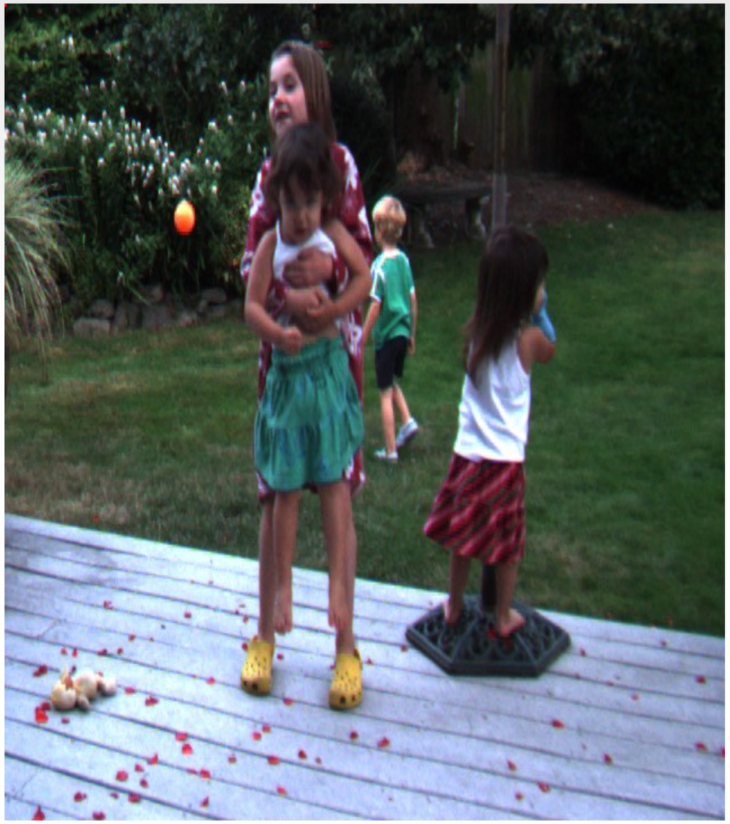
Each block is a feature. Therefore, in essence, through the use of boosted classifiers, at each stage feature selection is performed. Initially, bigger blocks are evaluated, and then smaller blocks in the later stages of the cascade. On average, 4.7 blocks are evaluated at each stage.

Edge Images of Integral HoG

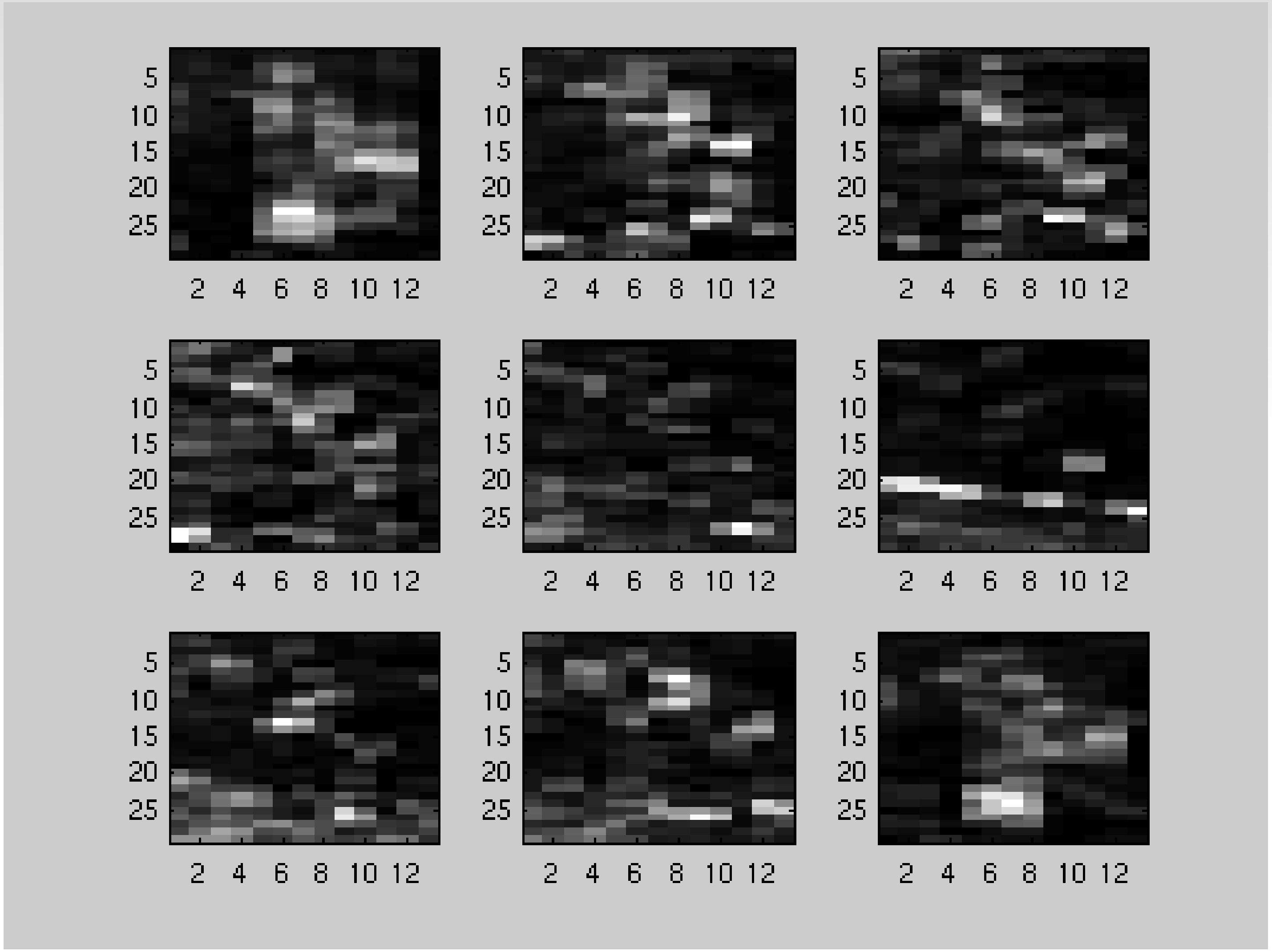


Two Consecutive Images

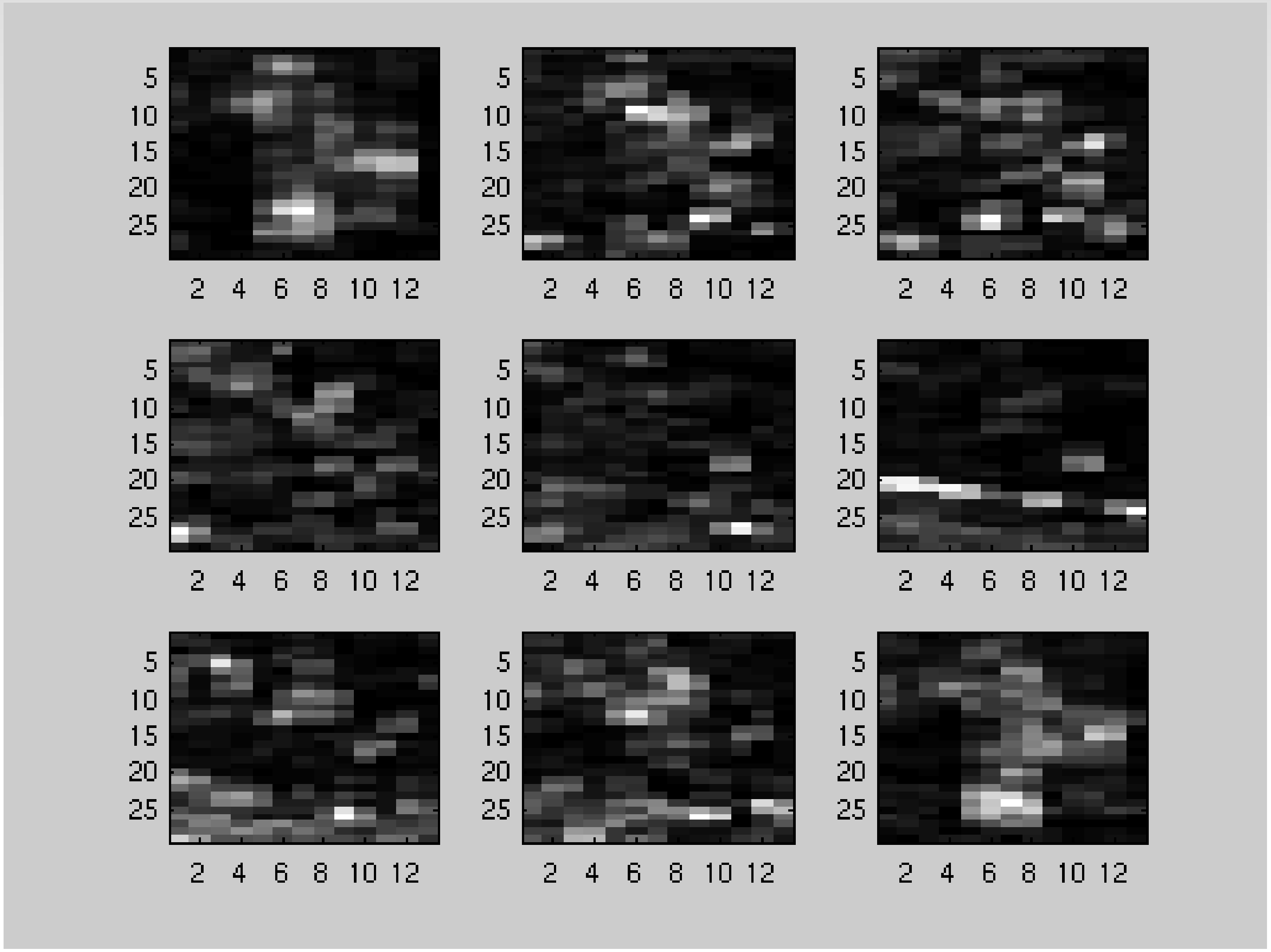




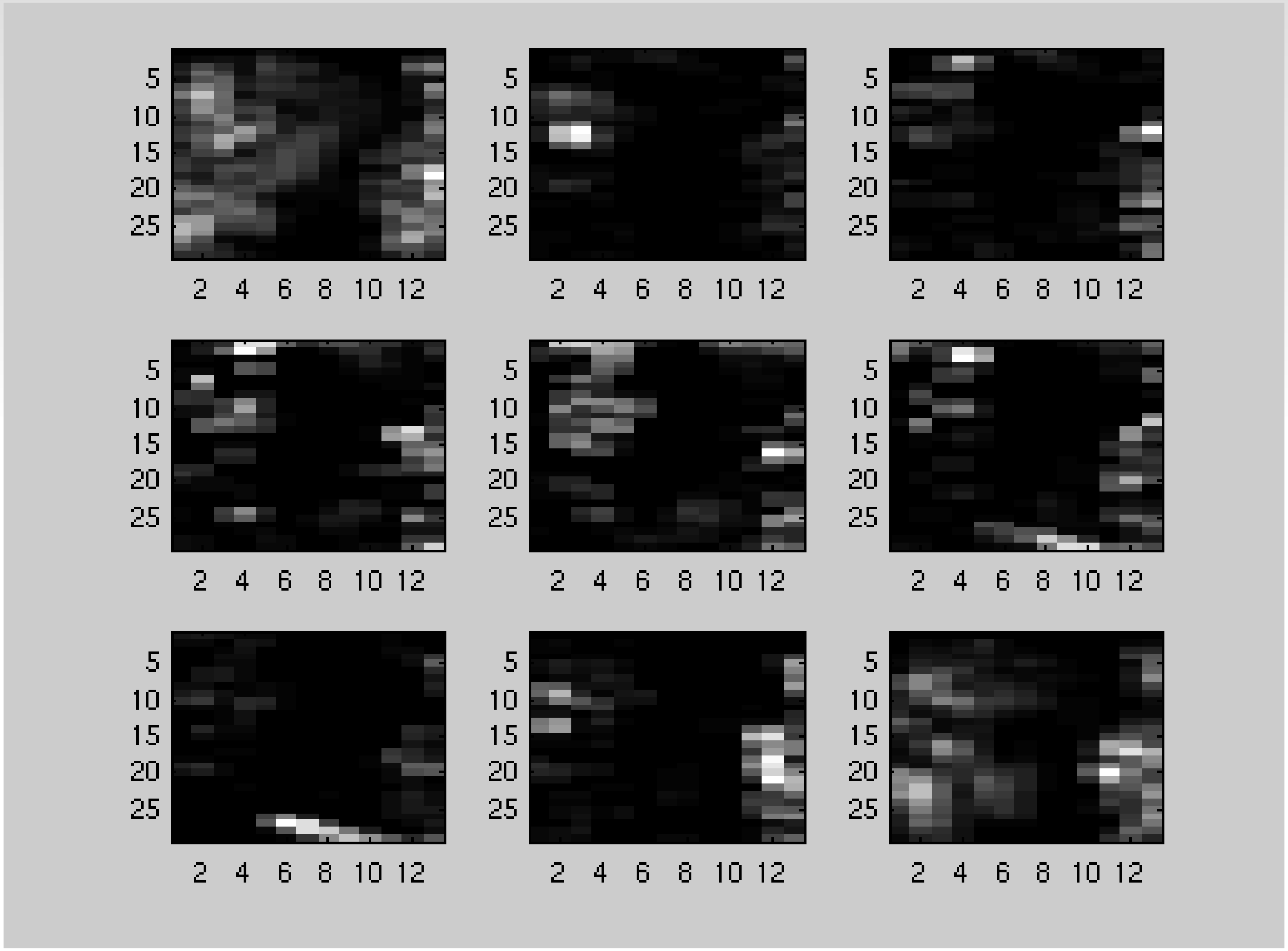
1st Edge Image



2nd Edge Image



A Different Image



Roadmap

- ✓ Literature Survey
- ✓ HoG + Gentleboost System
  - ✓ HoG
  - ✓ Gentleboost
  - ✓ Results
- ✓ Cascade of Rejectors
  - ✓ Integral HoG
- Future Directions

Conclusions/ Next Steps

- HoG + GentleBoost works reasonably well
- Need it to be faster for real-time detection
  - Implement cascade of rejectors using integral HoG
- Need to bring down the false positives
  - Use stereo vision (slide 14)
  - Integrate concurrently running tracking system into the detection system (slide 20)

Thank you!

Questions?

References

[1] Piotr Dollar, et al., “Pedestrian Detection: A Benchmark”

[2] Bernt Schiele, et al., “Visual People Detection – Different Models, Comparison and Discussion,” Proceedings of the IEEE ICRA 2009

[3] Pedro Felzenszwalb, et al., “A Discriminatively Trained, Multiscale, Deformable Part Model”

[4] Wael Abd-Almageed, et al., “Real-Time Human Detection and Tracking from Mobile Vehicles,” Proceedings of the 2007 IEEE, Sept. 30 – Oct. 3, 2007

[5] Ayato Toya, et al., “Pedestrian Recognition using Stereo Vision and Histogram of Oriented Gradients,” Proceedings of the 2008 IEEE ICVES, Sept. 22-24, 2008

[6] Feng Han, et al., “A Two-Stage Approach to People and Vehicle Detection with HOG-Based SVM”

[7] Christian Wojek, et al., “Multi-Cue Onboard Pedestrian Detection”

[8] Christian Wojek, et al., ”Sliding-Windows for Rapid Object Class Localization: a Parallel Technique”

[9] Qiang Zhu, et al., “Fast Human Detection Using a Cascade of Histograms of Oriented Gradients”

[10] Paul Viola, et al., “Robust Real-Time Object Detection,” Conference on Computer Vision and Pattern Recognition (CVPR), 2001

[12] Dalal, “Finding People in Images and Videos,” Thesis, 2006