Intelligent Vehicles Symposium, June 2012

Topic: Pedestrian Protection

Real-time Pedestrian Detection with Deformable Part Models

Hyunggi Cho
ECE, CMU

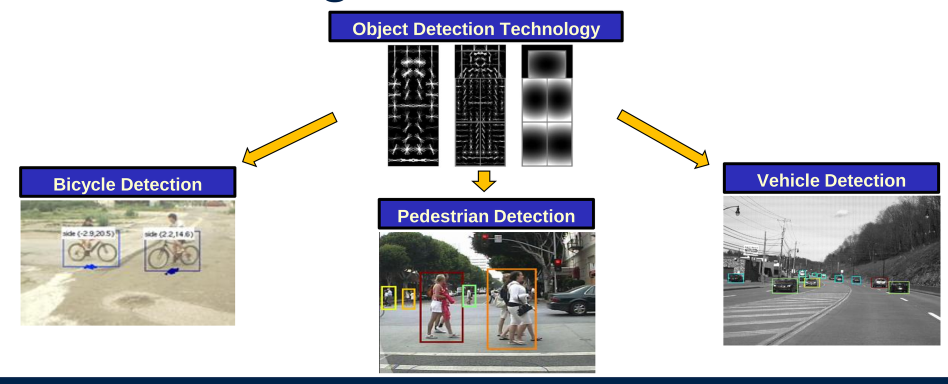
Paul E. Rybski
RI, CMU

Aharon Bar-Hillel
ATCI, GM

Wende Zhang
ECIL, GM

Introduction

- We engineered a well-known object detection method called deformable part models [10,11] to develop a real-time pedestrian detection system for use in automotive applications. Our system demonstrates superior detection performance when compared to many state-of-the-art detectors and is able to run at 14 fps on an Intel Core i7 computer when applied to 640x480 images. The contributions of this work are :
  - C implementation of baseline detectors [10, 11]
  - Simple image geometry analysis
  - Quantitative evaluation using the Caltech Pedestrian Benchmark [7]

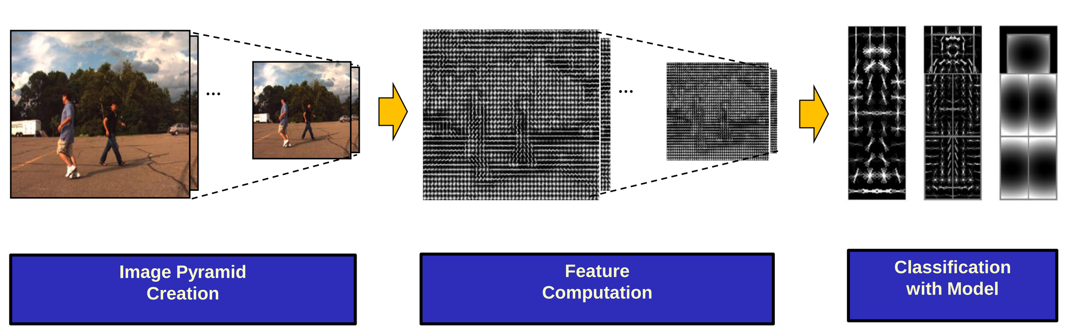


Implementation

Why Deformable Part-Based Model ?

- Elegant mechanism for handling a wide range of intra-class variability
  - Multiple sub-models for different view points
  - Dynamic configuration of parts for each view point
- Well-designed learning technique called 'latent SVM'
  - Does not require exact part labels
- Efficient detection method called star-cascade [10]
  - Suitable for real-time applications

Procedure for a multi-scale detection

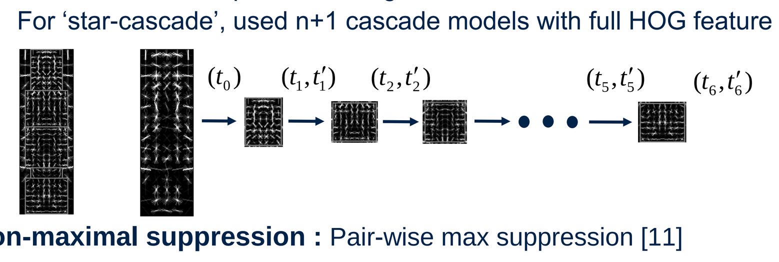


Profiling Results for All Implementations

Resolution	Module	Lenovo T400 (Intel® Core2 Duo P8800 @ 2.66GHz 2.67GHz)		Lenovo W520 (Intel® Core i7 – 2920XM @ 2.50GHz)	
		MATLAB _star_cascade [10]	C_star_cascade (Ours)	MATLAB _star_cascade [10]	C_star_cascade (Ours)
320x240 (QVGA)	Feature Comp.	305 ms	80 ms	165 ms	20 ms
	Detection	155 ms	60 ms	105 ms	25 ms
	FPS	2.2 fps	7.1 fps	3.7 fps	22.2 fps
640x480 (SVGA)	Feature Comp.	1145 ms	300 ms	840 ms	80 ms
	Detection	584 ms	330 ms	324 ms	105 ms
	FPS	0.6 fps	1.5 fps	0.9 fps	5.4 fps
1024x768 (HD)	Feature Comp.	3550 ms	750 ms	1770 ms	250 ms
	Detection	1810 ms	660 ms	758 ms	250 ms
	FPS	0.2 fps	0.7 fps	0.4 fps	2 fps

Implementation Details

- **Feature computation :**
  - Parallelize the original HOG feature computation using *pthread* library
  - 10X speed up for this operation
- **Sliding-window classification :**
  - For 'voc-release3', ported the original method's MEX function
  - For 'star-cascade', used n+1 cascade models with full HOG feature



- **Non-maximal suppression :** Pair-wise max suppression [11]

Quantitative Evaluation

Experimental Goal & Setup

- **Experimental Goal**
  - To find the key design parameters for the deformable part-based model
    - Number of training samples
    - Number of parts
    - Number of pyramid levels
  - Quantitative evaluation using the Caltech Pedestrian Dataset
- **Experimental Setup**
  - Data set : Caltech Pedestrian Dataset (4 hours video , 2.5 hours annotated)
  - Training set : S0~S5, Testing set : S6~S10
  - 347,000 total instances of pedestrians
  - 2,300 unique pedestrians
  - Image condition : Size : SVGA (640 x 480) , Frame rate : 30 fps , VFOV : 27˚

System Design Parameters

- **Number of Training Samples**
  - To obtain a statistically valid set of training images.
  - Training models with from the full data set at a different sampling frequency.

TABLE II
DIFFERENT SAMPLING SCHEMES FOR MODEL TRAINING

Sampling Scheme	No. of Pos. Samples	LAMR (%)
All frames	65570	56
Every 10th	6557	56
Every 20th	3261	55
Every 30th	2198	54
Every 40th	1629	55
Every 50th	1280	56
Every 60th	1088	58

- **Number of Parts**
  - To identify the optimal number of parts required for the pedestrian models
  - The optimal number of parts depends on the variability of an object class

TABLE III
DIFFERENT NUMBER OF PARTS

No. of Parts	LAMR (%)
2	58
3	55
4	55
5	56
6	54
7	56
8	56

- **Number of Scales Per Octave**
  - To find the best trade-off between detection rate and detection time.
  - Modern pedestrian detectors use two or three octaves and sample 8-14 scales per octave.

TABLE IV
DIFFERENT NUMBER OF SCALES PER OCTAVE

Scales / Octave	No. of Levels	LAMR (%)
2	10	61
4	19	56
6	28	56
8	37	55
10	46	54
12	56	54
14	65	53

Evaluation with Caltech Benchmark

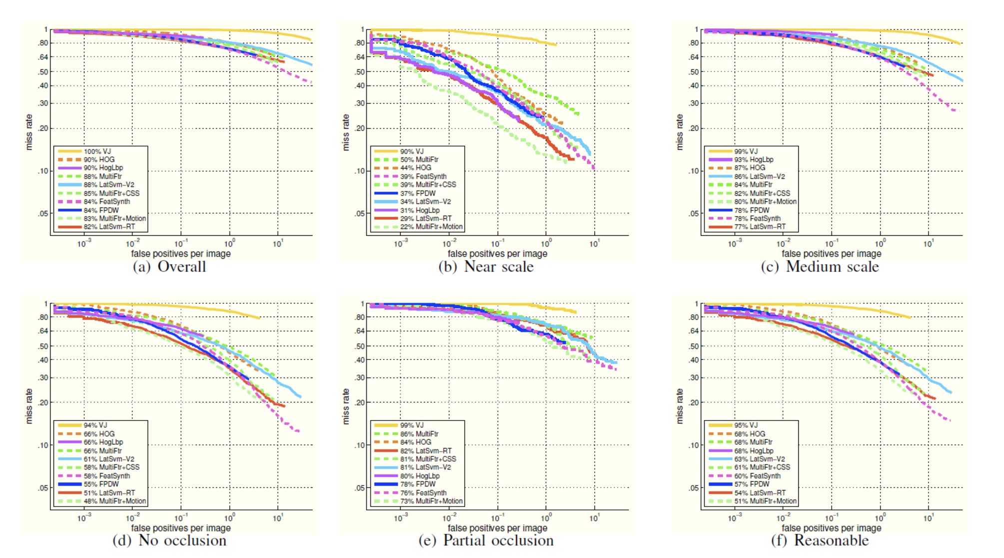


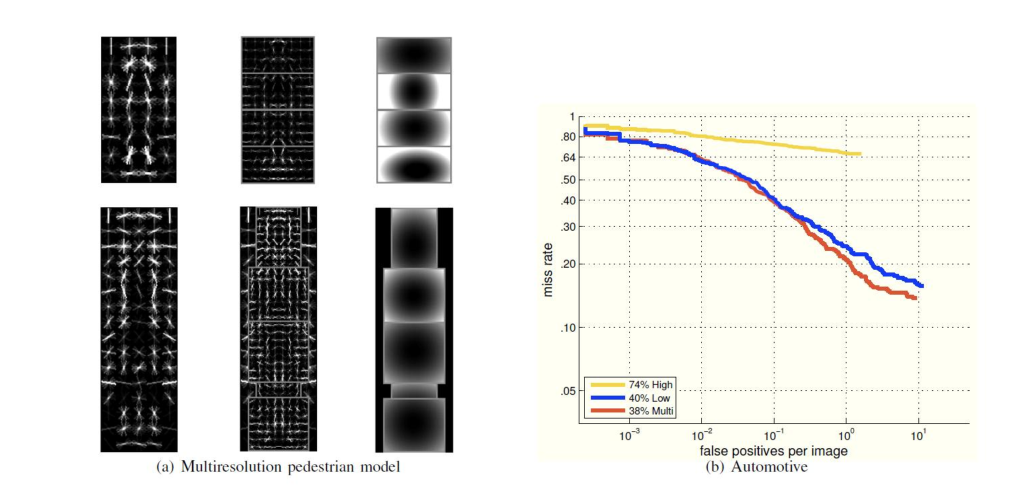
Fig. 4. Evaluation results using the same criterions in [7]. (a) Overall performance on all annotated pedestrian images. (b) Performance on unoccluded pedestrians over 80 pixels in height. (c) Performance on unoccluded pedestrians between 30-80 pixels in height. (d) Peformance on unoccluded pedestrians over 50 pixels in height. (e) Same as (d) but with partial occlusion. (f) Performance on pedestrians at least 50 pixels in height under no or partial occlusion.

Qualitative Detection Results

Fig. 6. Qualitative detection results on the Caltech testset. The first and second row shows correct pedestrian detections in various scenarios. The third row shows typical false positives.

Real-Time Evaluation

- **Motivation :**
  - To evaluate the system's real-time performance under a realistic setting
  - Thus, develop a new test scenario called *'automotive'*
  - Integrate the system into a real vehicle
- ***'Automotive'* scenario :**
  - Unlike *'Reasonable'* scenario in Caltech Benchmark, it only includes unoccluded pedestrians within 25m from a vehicle (corresponds to 70 pixels in height) in the ground truth.
  - Does not require upscaled input images.
  - Thus, trained a multiresolution pedestrian model to detect pedestrians up to 25m reliably.



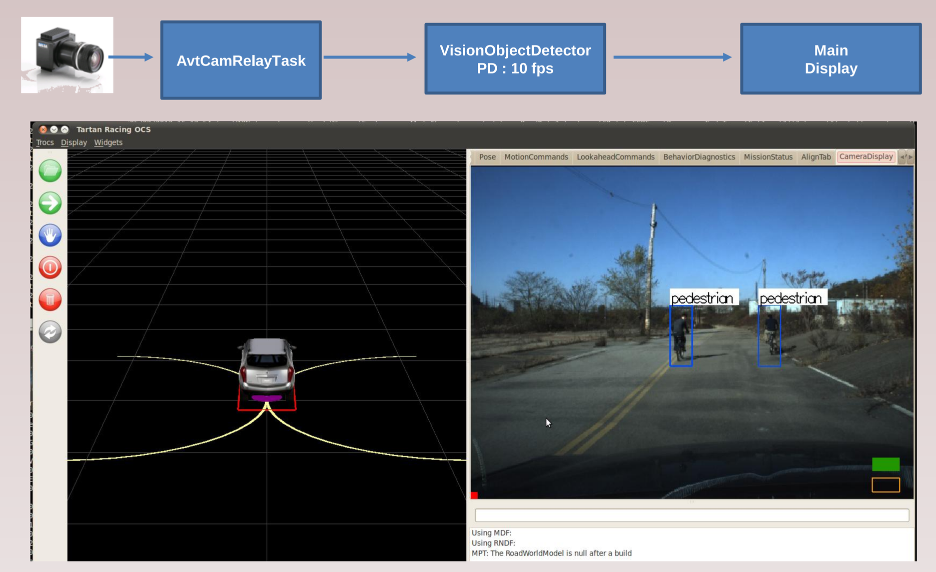
(a) Multiresolution pedestrian model (b) Automotive

Fig. 5. Evaluation results using the *automotive* criterion. (a) Multiresolution pedestrian model: high-resolution (128 × 40) with 475 positive samples and row-resolution (64 × 32) with 2430 positive samples. (b) Performance on unoccluded pedestirans over 70 pixels tall.

- **Performance evaluation with a real vehicle (Not included in the paper) :**
  - Integrate our pedestrian detection system into a CMU's experimental vehicle to evaluate its real-time performance
  - Computing hardware specification on the experimental vehicle :

Processor	Core 2 Extreme QX9300, 2.53 Ghz, 12MB Cache 4 cores / 4 threads
Memory	8GB DDR3 PC3-8500
Storage	40GB SSD
CANbus	PEAK Dual Channel miniPCI
GPU	GT 430 Fermi (low profile) 96 cores, 700 Mhz Graphics clock, 1400 Mhz CPU clock, 1GB DDR3 memory
Ethernet	2 Gigabit ports
Camera	Allied Vsion Technology, Prosilica GC1380C

- **Pedestrian Detection in Action (Not included in the paper)**



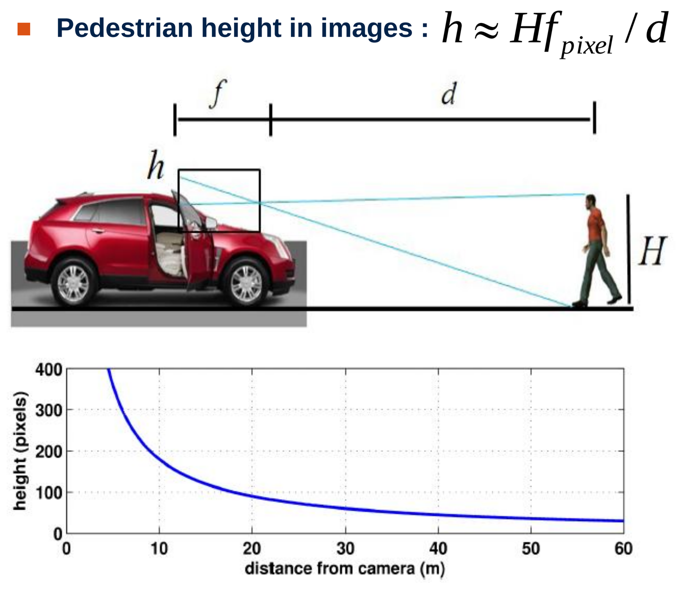
Geometry Analysis

Two primary benefits

- **Efficiency :** By searching only geometrically valid regions in the image space
- **Accuracy :** By suppressing a number of potential false positives from the irrelevant image space

Analyzing geometric relationships

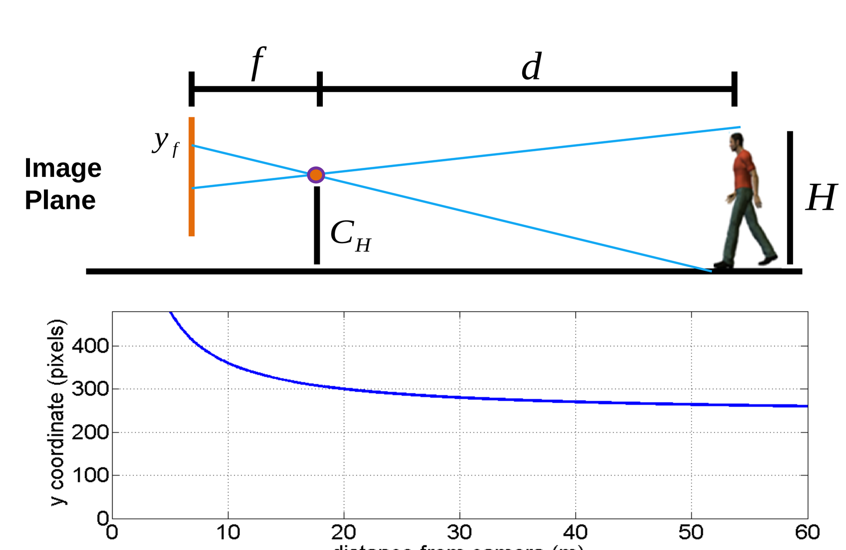
- **Pedestrian height in images :** $h \approx H f_{pixel} / d$



Symbol	Note
h	Observed pixel height
H	True height (e.g.,1.8m)
f_{pixel}	Focal length in pixels
C_H	Camera height (e.g., 1.2m)
y_f	y coordinate of foot postion

Fig. 3. Geometric constraints analysis. (a) Scene geometry. (b) Pixel height h as a function of distance d.

- **Depth computation :** $y_f = C_H f_{pixel} / d$



Conclusions & Future Work

- **Real-time pedestrian detection system using deformable part models**
  - C implementation of a baseline [11] and a star-cascade method [10]
- **Simple scene geometry analysis for an efficient feature pyramid search**
  - Pedestrian height in images / Depth computation (based on several assumptions)
- **Quantitative evaluation of our PD system on Caltech Pedestrian Benchmark**
  - Optimal design parameter for DPM HOG detector
  - 80% detection rate with 1 FPPI at 14fps@640x480 under our scenario called 'Automotive'
- **Partial occlusion handling algorithm as future work**