Real-time traffic sign detection

Hassan Shojania

Agenda

- Introduction
- Method [Escalera ’97]
- Color segmentation
- Mask Generation and corner detection
    - Angle dependent edge detection [Sandoval ’00]
    - Optimal corner detector [Rangarajan ’89]
- Shape recognition
- Results & observations
- Future Work
- References

Introduction

- Part of the bigger problem of *Autonomous vehicles*
  - Recognition of road and lane
  - Obstacle detection
  - Detection of passing vehicles
  - Following the course of own vehicle
  - Detection and interpretation of traffic signals
- Sample projects:
  - PROMETHEUS (Program for European Traffic with Highest Efficiency and Unprecedented Safety).
  - UC Berkeley's PATH (http://www-path.eecs.berkeley.edu or Computer Vision Group http://http.cs.berkeley.edu/projects/vision/)

Introduction

- PROMETHEUS collision avoidance project at Daimler-Benz (from [Heinze '97] "Trapper: Eliminating Performance Bottlenecks in a Parallel Embedded Application")
  - 18 cameras and 60 computing nodes (whole system)
  - Parallel system/application staged as a pipeline

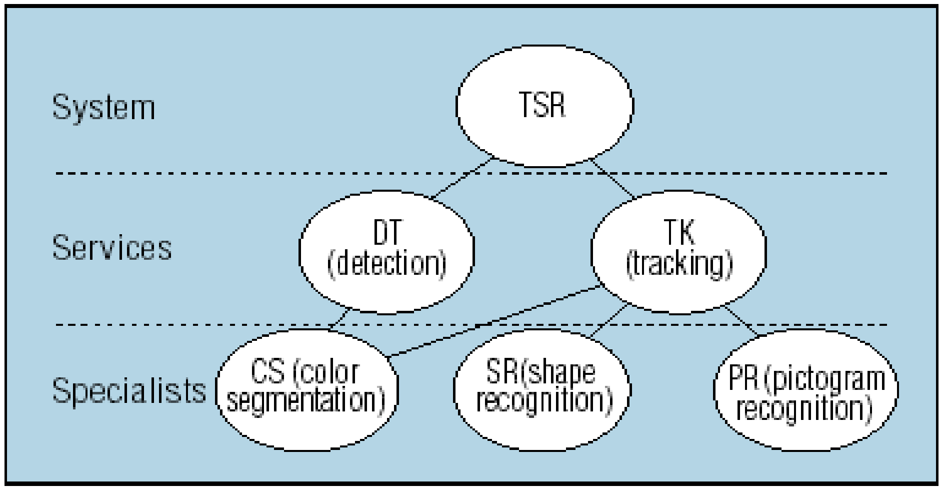
Mercedes-Benz test vehicle with image processing system, hazard assessment system, and automatically controlled brakes, accelerator and steering. *Figure taken from [Heinze97].*

TSR system. The image processing system selects elements based on their color characteristics.

Figure taken from [Heinze97].

Introduction

- Daimler's Traffic Sign Recognition (TSR) system
  - Initially based on Transputer processors, then moved to PowerPC601.
  - Detection Process (DT): Scans an image for possible sign candidates and forwards them to the TK.
    - Color segmentation specialist: Classifies regions of picture with probable traffic sign based on color of pixels in the region.
  - Tracking Process (TK): Classifies and identifies signs within an image. Also tracks each recognized sign in subsequent images.
    - Shape recognition specialist: Classifies candidates according to their contour.
    - Pictogram-recognition specialist: Classifies the pictograms inside a traffic sign by comparing against the library.



Hierarchical structure of the TSR system.

Figure taken from [Heinze97].

Introduction

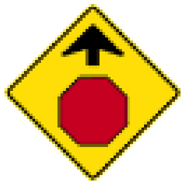
- Offline traffic-sign recognition is not very difficult problem in principle.
- Signs are 2D with discriminating shape and colors.
- Many papers in mid 90's using different methods from neural networks, fuzzy logics, … applied to different stages.
- Issues:
  - Variety of signs with different colors, shape and pictographic symbols
  - Complex and uncontrolled road environment (lighting, shadow, occlusion, orientation, distance, …)
  - *Real-time recognition!!*

Method

- Based on [Escalera '97] "Road Traffic Sign Detection and Classification"
- Main stages:
  - 1. Color segmentation
  - 2. Corner detection
  - 3. Shape recognition
  - 4. Sign classification (based on neural network and for triangular/ circular signs)
    - Training sets generated from 9 signs, 5 different rotation angle, 3 different noise levels, 4 different color threshold and 3 horizontal displacement level.
- Signs considered:
  - Signs with equilateral triangles one vertex upward
  - Circular signs with red border
  - Rectangles
  - *Yield and stop signs are excluded.*
  - *Relatively ideal case, not much tolerant of projection effect.*
  - *Doesn't mention shape verification method and issues will see later…*

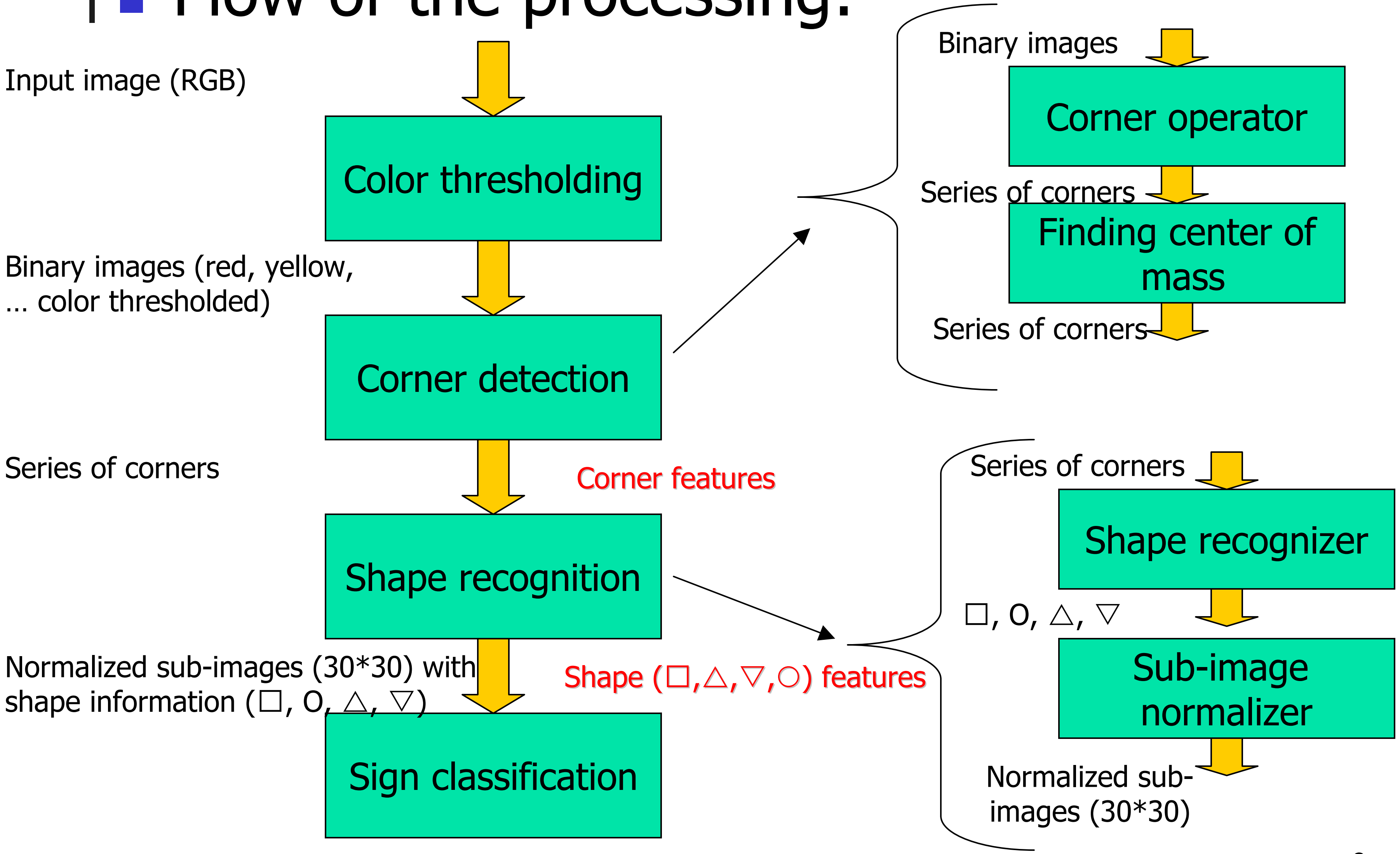
Method

- Different sign types

	European	North American
Warning		
Regulatory & obligation		
Informative		PHOENIX TUCSON

Method

- Flow of the processing:

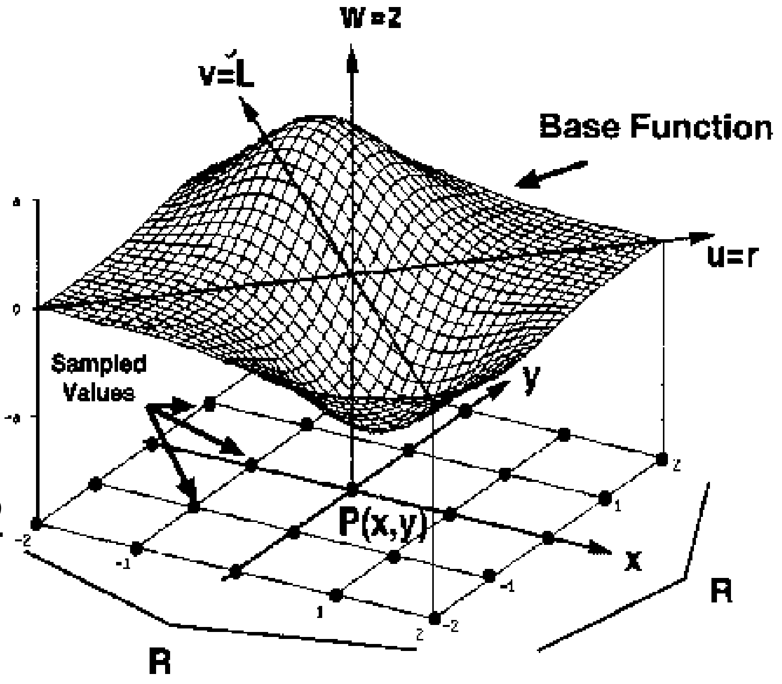


Method

- Our limitations/assumptions:
  - Considering only yield sign, stop sign and red bordered circular signs
  - No pictographic classification
  - Inherited from original method:
    - Pictures are not rotated. Just minor tilt due to camera position allowed. Basically same view as what a driver sees normally.
    - Pictures are not taken from a narrow angle, some degree of skew allowed but not very much.
    - Occlusion not considered

Angle dependent edge detection

- [Sandoval '00] "Angle-dependent Edge Detection for Traffic Sign Recognition"
- Generates convolution masks to detect circular and radial edges.
- Rotates the basis function *g(u, v)* around the center of image (where center of circle is assumed) to each individual point in the image.
- Creates position-dependent convolution mask.
  - Detector of Circular Edges (DCE) by aligning v at θ+π/2
  - Detector of Radial Edges (DRE) by aligning v at θ+π

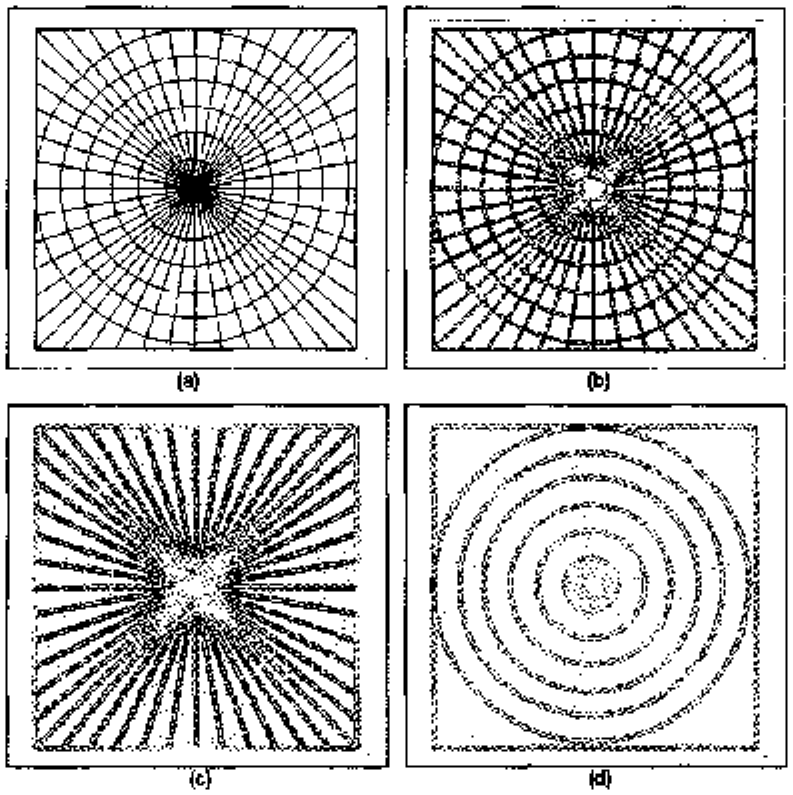


Base Function for a pixel rotated $\frac{\pi}{4}$.

Picture taken from [Sandoval '00]

Angle dependent edge detection

- Advantages
  - Custom-made masks for detection of circles of particular size or radial edges in any direction.
- Drawbacks?
  - Many masks for each point on the circle contour, and for every size.
  - Center of circle must be known!
- 5*5 masks



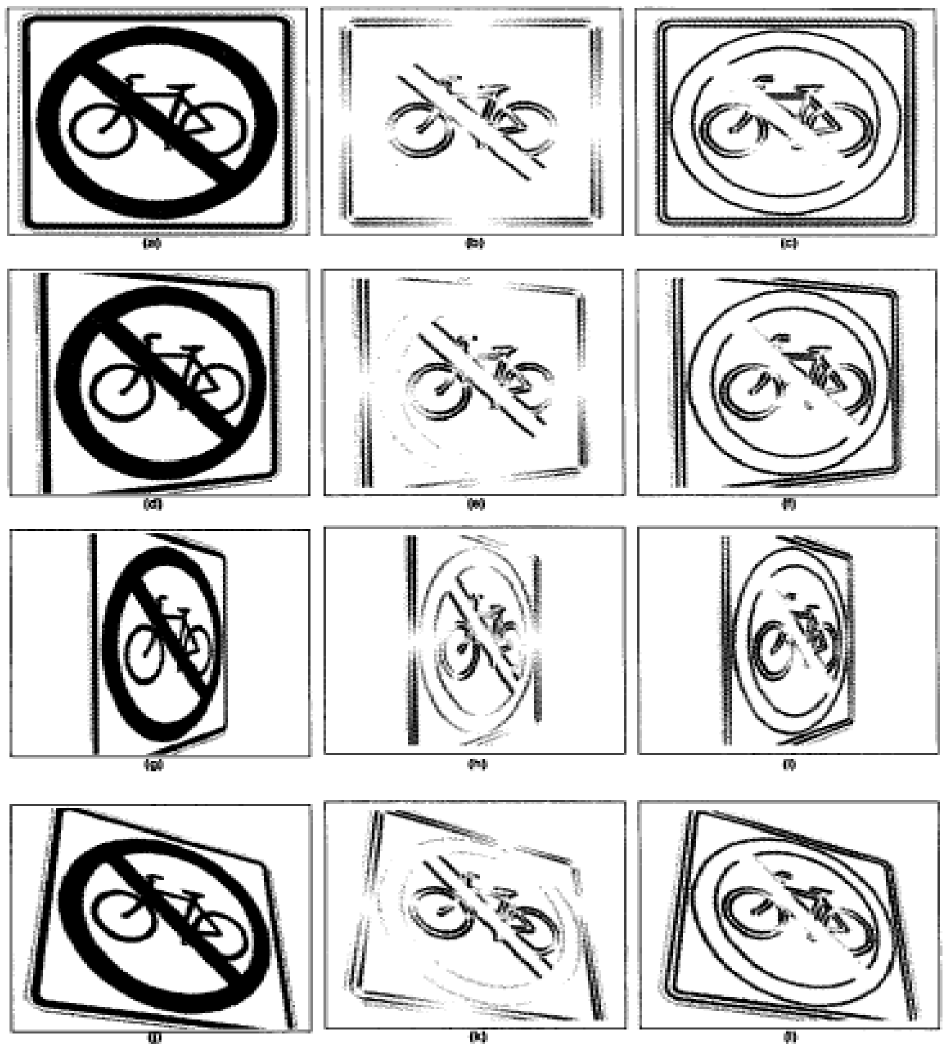
(a) Test Image (b) Sobel response

(c) DRE response (d) DCE response

Picture taken from [Sandoval '00]

Angle dependent edge detection

- Masks applied here are for *filtering* the circles/radials.
- Decomposes the original image.



Picture taken from [Sandoval '00]

Response using an non-real image of a traffic sign. (a) Original front image. (b) **DRE** response. (c) **DCE** response. (d) Original image rotated $\frac{\pi}{6}$. (e) **DRE** response. (f) **DCE** response. (g) Original image rotated $\frac{\pi}{3}$. (h) **DRE** response. (i) **DCE** response. (j) Original image rotated $\frac{\pi}{6}$ in plane YZ and $\frac{\pi}{6}$ in plane XZ. (k) **DRE** response. (l) **DCE** response.

Optimal corner detector

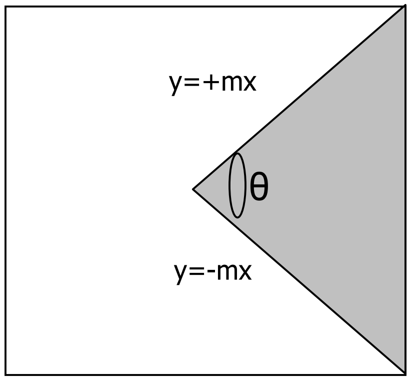
- [Rangarajan '89] "Optimal corner detector" (no electronic version available; go to library).
- Current corner detectors involve many stages:
  - Use edge information, or
  - Computing the gradient directions/rate of change
- Considers gray level characterization around a small neighborhood of a *corner with particular angle and orientation* to find a corner mask ➔ classified corners
- From the infinite number of possible masks (infinite number of corner angles and orientations), they argue that only 12 masks are good approximate of the whole set!
- Similar to edge detectors, called a "corner operator".

Optimal corner detector

- Qualitative objectives:
  - (*)Should not be sensitive to noise.
  - (*)Should not delocalize the corner.
  - Detected corner should be an edge point too.
  - The corner point should have at least two neighbors with different gradient than the corner itself.

  - Converts the first two objectives into quantitative functions.
  - Using variational calculus to solve the optimization problem.

- Canny edge operator
  - Good Detection
  - Good localization
  - Single response to an edge



Optimal corner detector

- Follows very closely Canny's approach.

Corner operator

$$I(x,y)=\begin{cases} A & \text{if } x>0 \text{ and } -mx<y<mx \\ 0 \end{cases} \qquad (1)$$

$$F(x,y)=I(x,y)+n(x,y) \qquad (2)$$

$$O(x,y)=F(x,y)*g(x,y) \qquad (3)$$

$$SNR=\Xi=\frac{A\int_{0}^{+\infty}\int_{-mx}^{mx} g(x,y)\,dydx}{n_0\sqrt{\int_{-\infty}^{+\infty}\int_{-\infty}^{\infty} g^2(x,y)\,dydx}} \qquad (4)$$

$$Delocailza\ tion=E[x_0^2+y_0^2]=\Lambda=$$

$$\frac{n_0^2\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} g^2(x,y)\,dydx}{\left(A\int_{0}^{+\infty}\int_{-mx}^{mx} g_{xx}(x,y)\,dydx\right)^2}+\frac{n_0^2\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} g^2(x,y)\,dydx}{\left(A\int_{0}^{+\infty}\int_{-mx}^{mx} g_{yy}(x,y)\,dydx\right)^2} \qquad (5)$$

$$\text{Maximize}\quad (SNR\ /\ Delocaliza\ tion\ )$$

By minimizing $\int_{-\infty}^{+\infty}\int_{-\infty}^{\infty} g^2(x,y)dydx$
with all other integrals held constant as constraints.

Canny operator

$G(x)$: Step function

$$SNR=\frac{\left|\int_{-W}^{+W} G(-x)\,f(x)\,dx\right|}{n_0\sqrt{\int_{-W}^{+W} f^2(x)\,dx}}$$

$$Delocailza\ tion=E[x_0^2]=$$

$$\frac{n_0^2\int_{-W}^{+W} f'^2(x)\,dx}{\left(\int_{-W}^{+W} G'(-x)\,f'(x)\,dx\right)^2}=\delta x_0^2$$

$$Localizati\ on=\frac{1}{\delta x_0}=\frac{\left|\int_{-W}^{+W} G'(-x)\,f'(x)\,dx\right|}{n_0\int_{-W}^{+W} f'^2(x)\,dx}$$

$$\text{Maximize}\quad (SNR\ \cdot Localizati\ on\ )$$

Optimal corner detector

- After change of variable, using LaGrange multiplier, and solving the partial differential equation:

$$g(x,y)=\begin{cases} c_1\cdot\sin\dfrac{m\pi x}{W}\cdot\left[-\left(e^{zW}+e^{-zW}\right)+e^{zy}+e^{-zy}\right] & \text{for corner portion} \quad (6) \\ c_2\cdot\sin\dfrac{n_1\pi x}{W}\cdot\sin\dfrac{n_2\pi y}{W} & \text{for non-corner portion} \quad (7) \end{cases}$$

- Since 0<=y<=W, the exponential portion of (6) is negative, so choosing m=-1 to end up with positives for corner portion.
- Choosing n_1=1 and n_2 =-1 to make (7) negative
- Couldn't find rational for selecting z. Higher z means better noise suppression. Same with higher W.
- A 9*9 mask for θ=60° and choosing z=0.2 :

-3	-5	-5	-3	0	-3	-5	-5	-3
-5	-9	-9	-5	0	-5	-9	-9	-5
-5	-9	-9	-5	0	-5	-9	8	5
-3	-5	-5	-3	0	6	9	9	6
0	0	0	0	0	6	10	10	6
-3	-5	-5	-3	0	6	9	9	6
-5	-9	-9	-5	0	-5	-9	8	5
-5	-9	-9	-5	0	-5	-9	-9	-5
-3	-5	-5	-3	0	-3	-5	-5	-3

Optimal corner detector

- What's the drawback?
- Need to enumerate all possible corners we're interested and generate a mask for each one.
- ➔Have to apply several masks to every point of image.
- Solution?
  - Approximate a class of corners to one and use one mask for all the class.
  - They claim the following 12 class are enough.

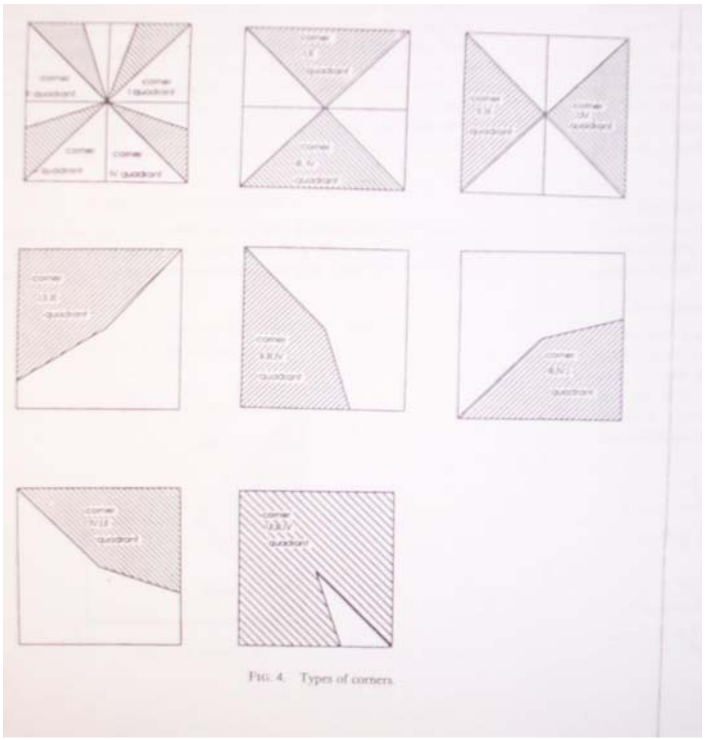


Figure taken from [Rangarajan' 89].

Optimal corner detector

- Introducing basic masks:

-6	-11	-11	-6	0	4	7	7	4
-11	-18	-18	-11	0	8	13	13	8
-11	-18	-18	-11	0	10	17	17	10
-3	-5	-5	-3	0	12	19	19	12
0	0	0	0	0	12	20	20	12
-6	-11	-11	-6	0	-6	-11	-11	-6
-11	-18	-18	-11	0	-11	-18	-18	-11
-11	-18	-18	-11	0	-11	-18	-18	-11
-6	-11	-11	-6	0	-6	-11	-11	-6

90° corner detector. Responds well to all corners in quarter 1 (with different response).

4	7	7	4	0
8	13	13	8	0
10	17	17	10	0
12	19	19	12	0
0	0	0	0	0

12	19	19	12	0
10	17	17	10	0
8	13	13	8	0
4	7	7	4	0
0	0	0	0	0

-6	-11	-11	-6	0
-11	-18	-18	-11	0
-11	-18	-18	-11	0
-3	-5	-5	-3	0
0	0	0	0	0

12	20	20	12	0

12
20
20
12
0

Anything Missing?

Why zeros?

Basic masks: C1, C2, NC, CX

C1	C2
NC	CX

Optimal corner detector

- Convolving with the mask can be calculated using smaller base masks; for example:
  - Mask size = 2n + 1
  - d= (n + 1)/2

$$I(x,y)*g(x,y)=I_{C1}(x+d,y+d-1)+I_{CX}(x+d,y)+I_{NC}(x+d,y-d)+$$
$$I_{NC}(x-d+1,y+d-1)+I_{NC}(x-d+1,y-d)$$

- Can build the other masks in the same way.
- Even can split each sub-masks C1, C2 and CNC into two one-dimensional masks using separation:

$$I(x,y)*C1(x,y)=\left(\sum_j\left(\sum_i I(x+i,y+j)\cdot XC1(i)\right)\cdot YC1(j)\right)$$

$$\begin{array}{|c|c|c|c|}\hline 2 & 4 & 4 & 2 \\ \hline\end{array} \qquad \begin{array}{|c|}\hline 2 \\ \hline 3 \\ \hline 4 \\ \hline 4 \\ \hline\end{array}$$

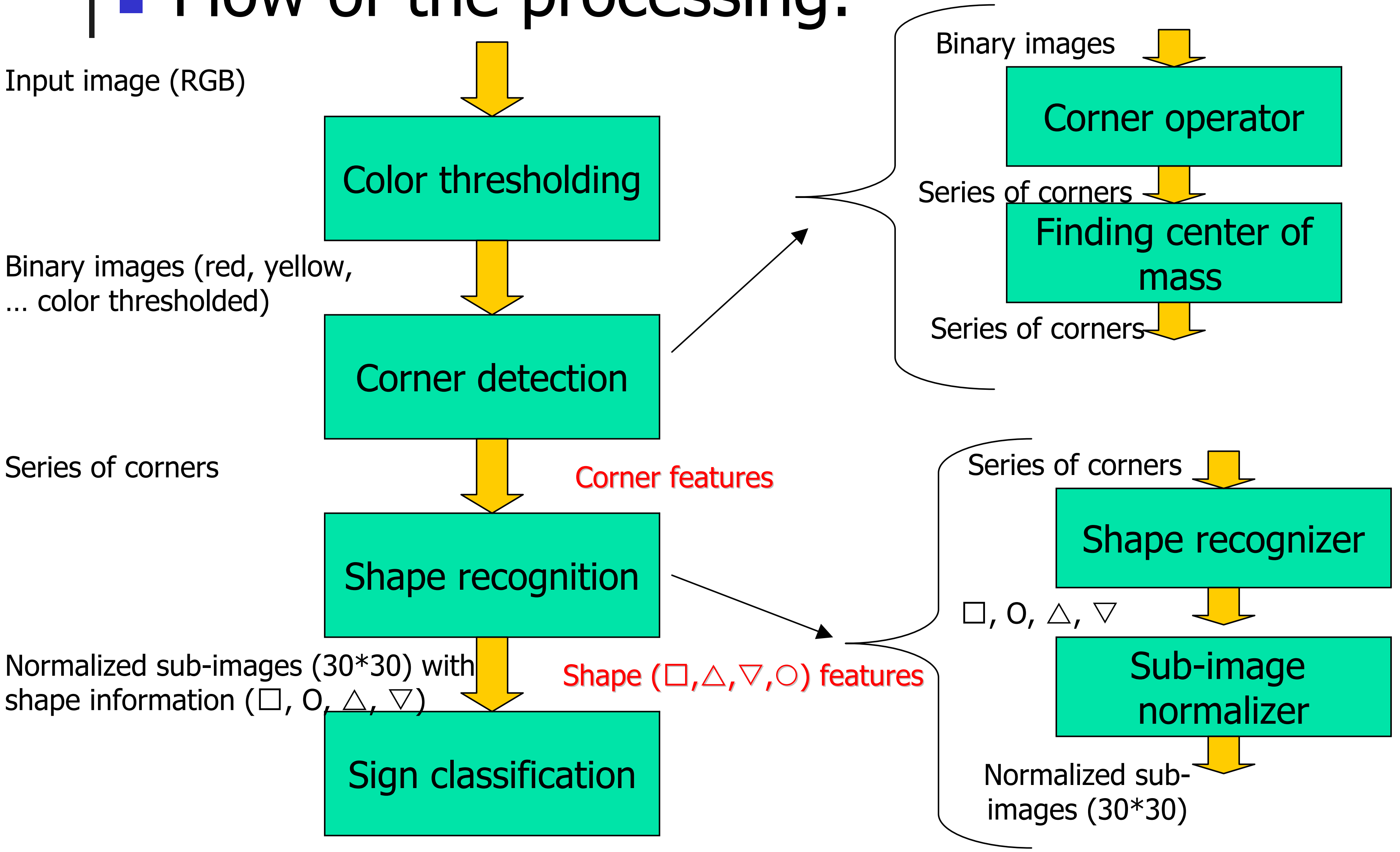
XC1 and YC1

Optimal corner detector

- Algorithm:
  - Compute cornerness at all pixels by applying the corner masks to the image and apply a threshold.
  - Remove candidate corner pixels which are not edge pixels using a detector like Canny.
  - Discard corners which have at least two neighbors in a 3 by 3 neighborhood with a similar gradient angle.
- We’re not going to use 2nd and 3rd items.

Method (revisiting)

- Flow of the processing:



Color segmentation

- Traffic signs have very discriminating background/border colors.
- Color segmentation can be seen as a classification task. The target is different class of colors for the set of signs.
- Different approaches:
  - Using a discovery neural networks to separate clusters (i.e. classes) of colors [Kehtarnavaz ’95].
  - Split and merge concept. Picks an initial set of class centers. Distribute pixel colors among them. Split or merge classes based on statistical data of class members, distance from each other [Kang ’94].
  - Color thresholding. Using a different threshold for each class of color.

$$g(x,y) = \begin{cases} k_1 & \text{if } \begin{cases} R_l <= R(x,y) <= R_h \\ G_l <= G(x,y) <= G_h \\ B_l <= B(x,y) <= B_h \end{cases} \\ k_2 & \text{otherwise} \end{cases}$$

Color segmentation

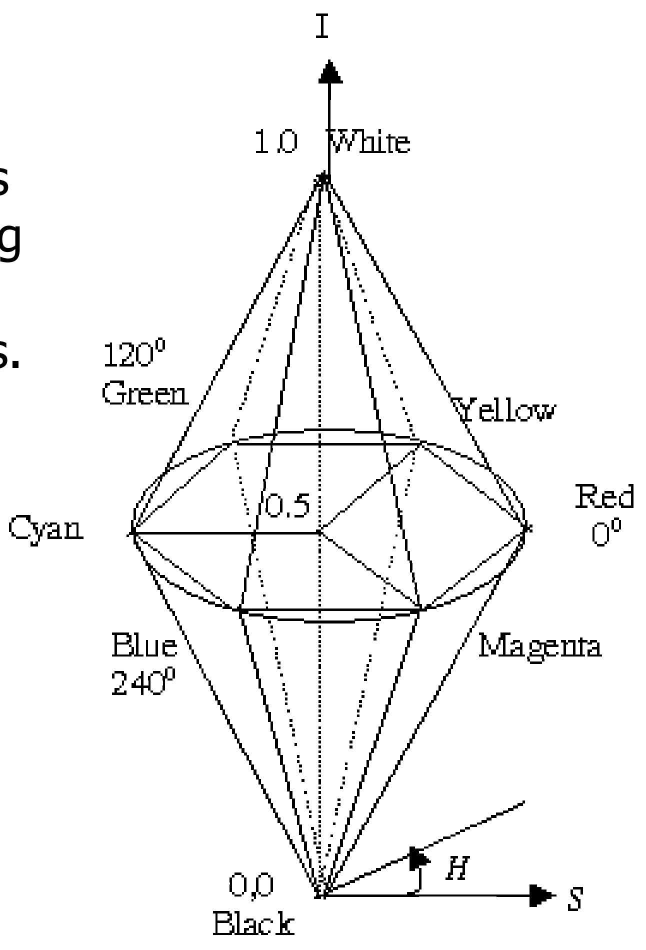
- Big problem: RGB color space is very sensitive to lighting; e.g. due to sun angle, weather, clouds, …
- HSI color space is the answer. It decouples the intensity component from color-carrying information (hue:purity, saturation:dilution by white light). See 6.2 of Gonzalez-Woods.

$$H = \begin{cases} \theta & \text{if } B <= G \\ 360 - \theta & \text{if } B > G \end{cases}$$

$$\theta = \cos^{-1}\left[\frac{\frac{1}{2}[(R-G)+(R-B)]}{\left[(R-G)^2 + (R-B)(G-B)\right]^{1/2}} \right]$$

$$S = 1 - \frac{3}{(R+G+B)}\left[\min(R,G,B)\right]$$

$$I = \frac{1}{3}(R+G+B)$$



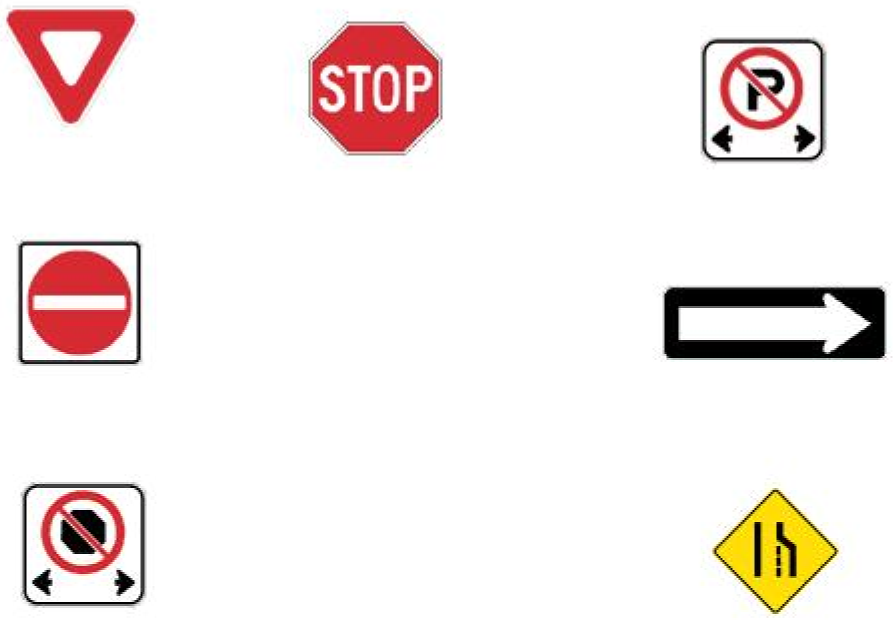
Color segmentation

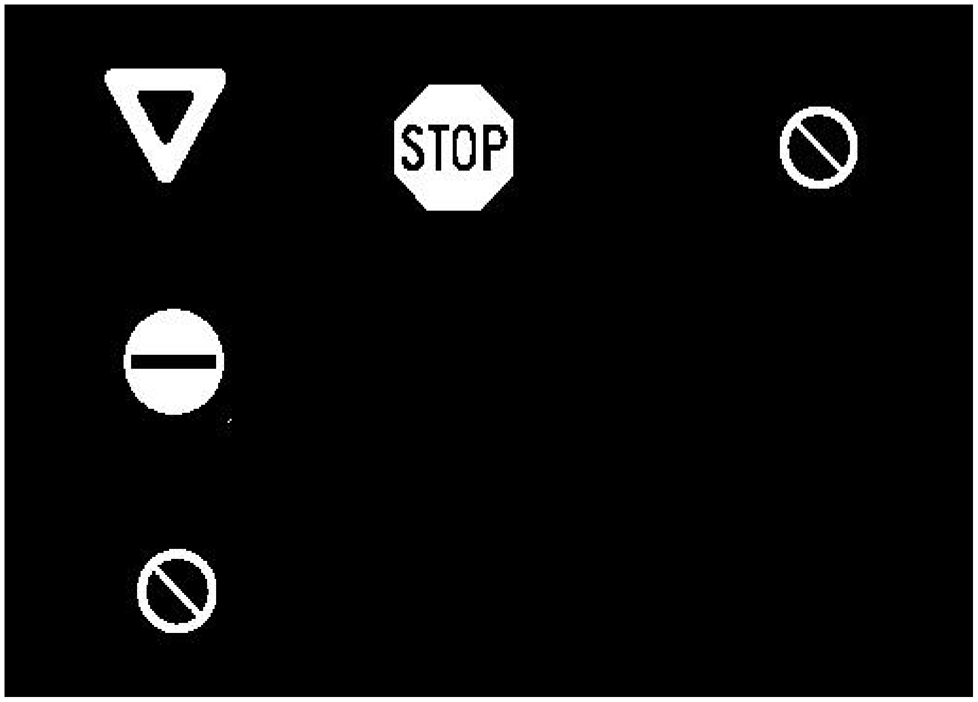
- Converting to HIS has heavy computational cost!
- Instead, comparing the color/intensity ratio to do the threshold.
- We’re interested in red color because of the scope of our signs.
- A variation is:

$$g(x,y) = \begin{cases} k_1 & \text{if } \begin{cases} R_l <= R(x,y) <= R_h \\ G'_l <= \dfrac{G(x,y)}{R(x,y)} <= G'_h \\ B'_l <= \dfrac{B(x,y)}{R(x,y)} <= B'_h \end{cases} \\ k_2 & \text{otherwise} \end{cases}$$

Color segmentation

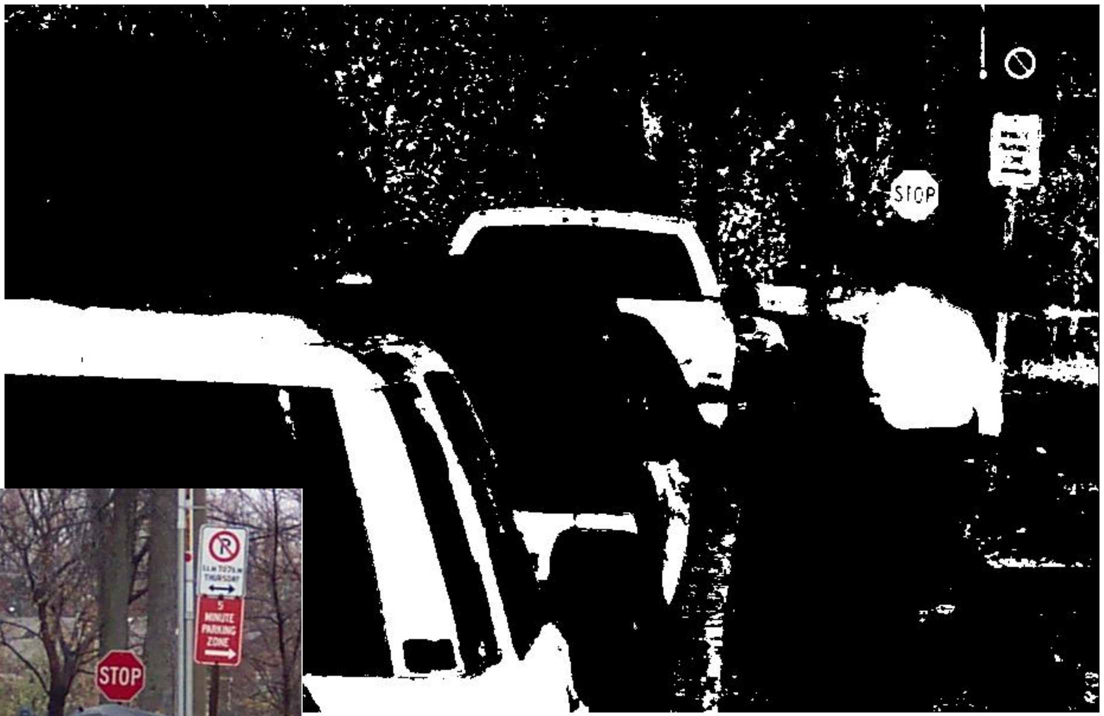
- Sample1:

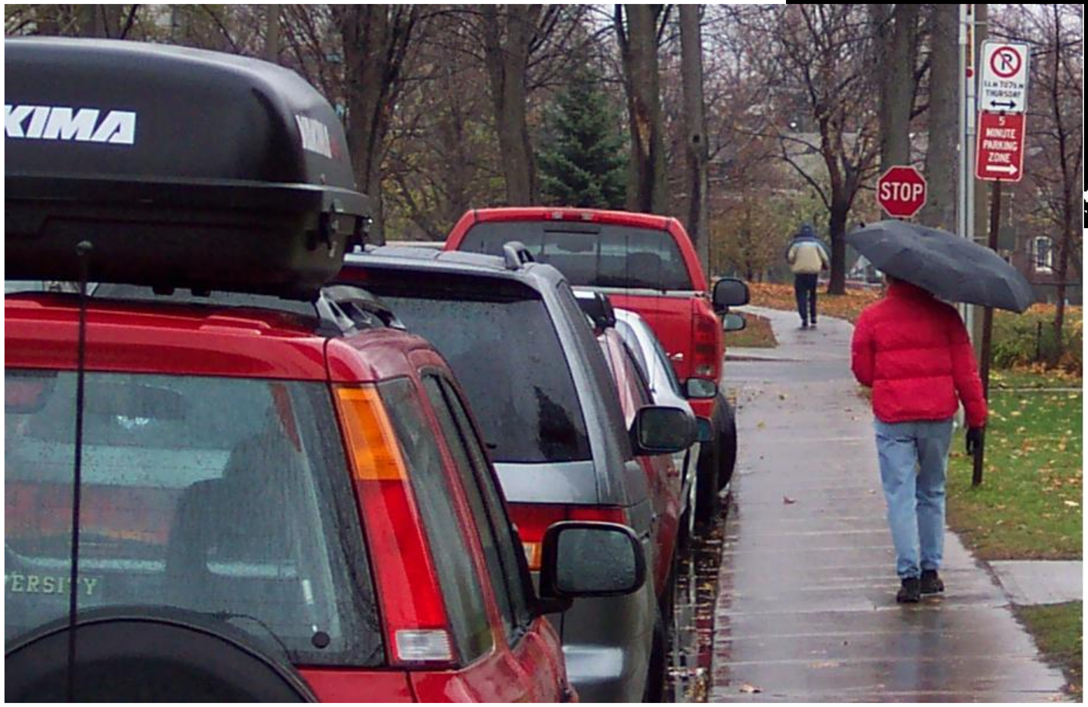




Color segmentation

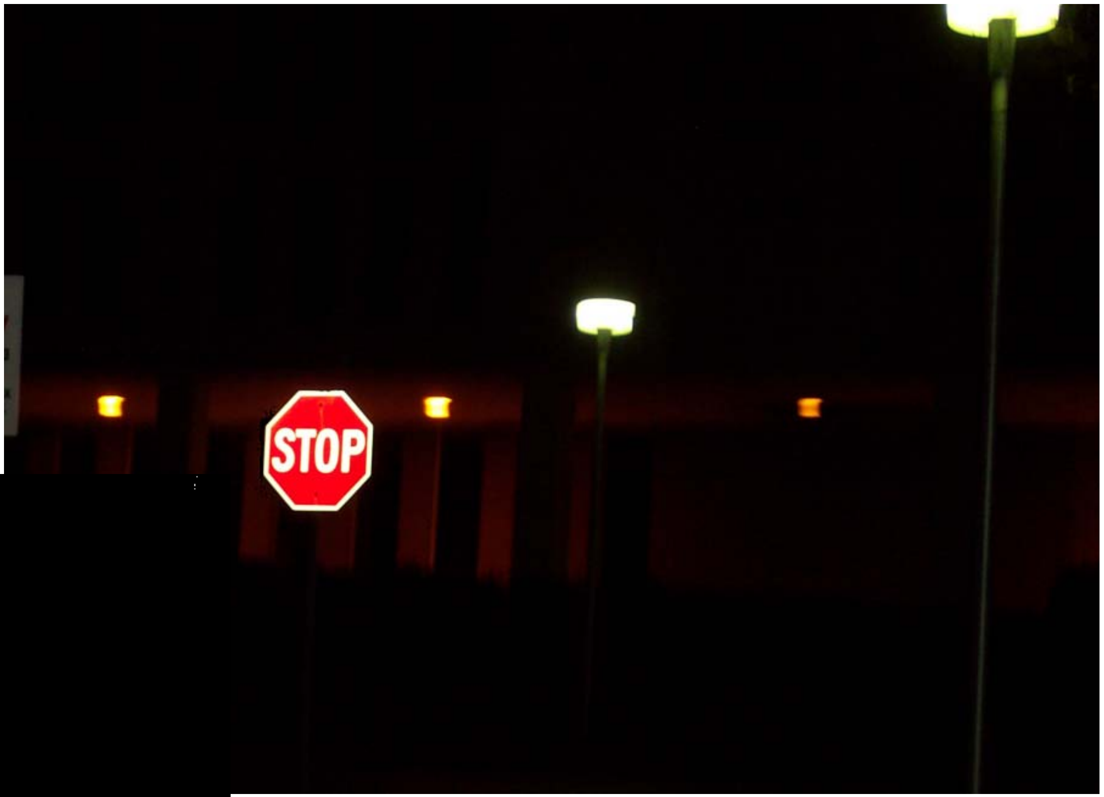
- Sample2:

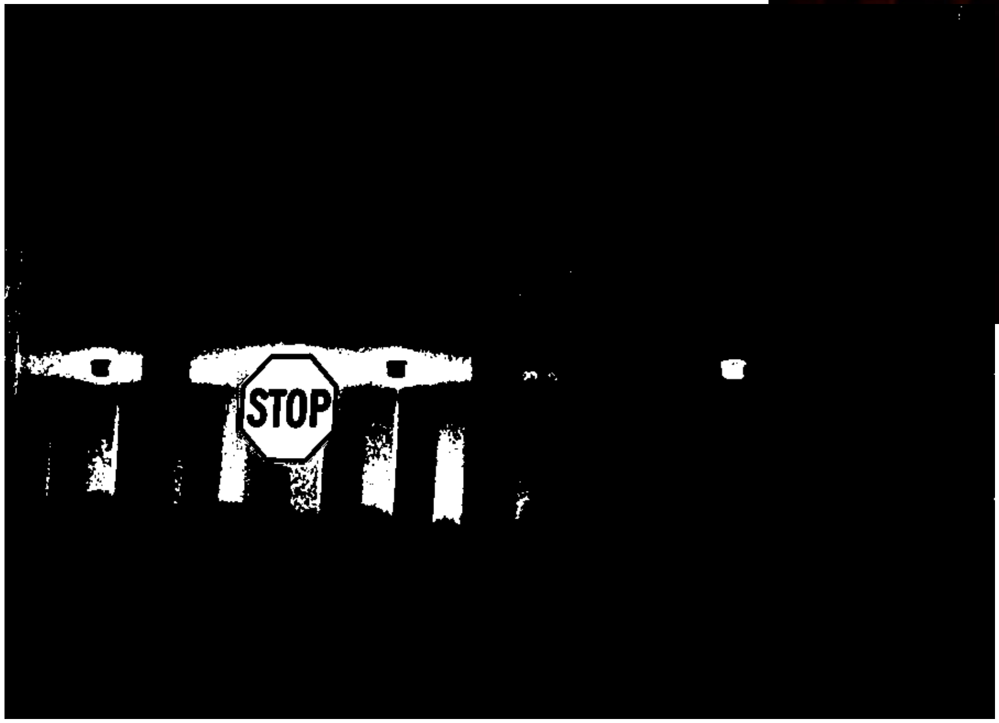




Color segmentation

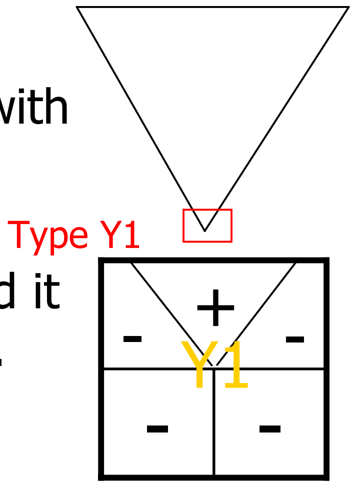
- Sample3:





Corner detection

- Using optimal corner detection [Rangarajan '89].
- *_Less expensive_. Input image is a binary image. No multiplication necessary. Can be done very fast with SSE2 instructions.
- 5 different 9*9 masks:
  - Y1: 1 for central lower corner of yield sign (60° corner).
  - C1, C2, C3, C4: 4 masks for 4 kinds of 90° corner.
- Steps for each mask:
  - For every pixel:
    - Convolve mask with every pixel and threshold result.
    - If corner, append it to list of corners.
  - Refine the list of corners based on center of gravities.

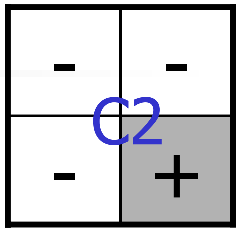


0	12	20	20	12	20	20	12	0
-6	-11	10	12	12	12	10	-11	-6
-11	-18	17	19	20	19	17	-18	-11
-11	-18	-18	19	20	19	-18	-18	-11
-6	-11	-11	12	12	12	-11	-11	-6
-6	-11	-11	-6	0	-6	-11	-11	-6
-11	-18	-18	-11	0	-11	-18	-18	-11
-11	-18	-18	-11	0	-11	-18	-18	-11
-6	-11	-11	-6	0	-6	-11	-11	-6

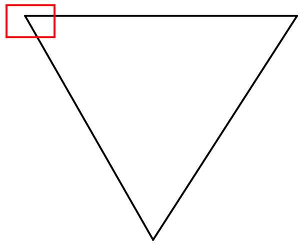
Central lower corner (60° corner)

Corner detection

- Masks for yield sign:



Type Y2=C2



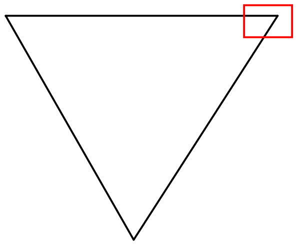
-6	-11	-11	-6	0	-6	-11	-11	-6
-11	-18	-18	-11	0	-11	-18	-18	-11
-11	-18	-18	-11	0	-11	-18	-18	-11
-6	-11	-11	-6	0	-6	-11	-11	-6
0	0	0	0	0	12	20	20	12
-6	-11	-11	-6	0	12	19	19	12
-11	-18	-18	-11	0	10	17	17	10
-11	-18	-18	-11	0	-11	13	13	8
-6	-11	-11	-6	0	-6	7	7	4

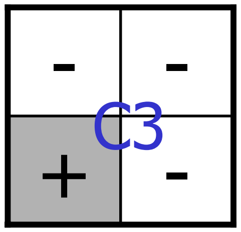
Mask for upper left corner of yield sign

-6	-11	-11	-6	0	-6	-11	-11	-6
-11	-18	-18	-11	0	-11	-18	-18	-11
-11	-18	-18	-11	0	-11	-18	-18	-11
-6	-11	-11	-6	0	-6	-11	-11	-6
0	0	0	0	0	12	20	20	12
-6	-11	-11	-6	0	12	19	19	12
-11	-18	-18	-11	0	10	17	17	10
-11	-18	-18	-11	0	8	13	13	8
-6	-11	-11	-6	0	4	7	7	4

Approximated mask

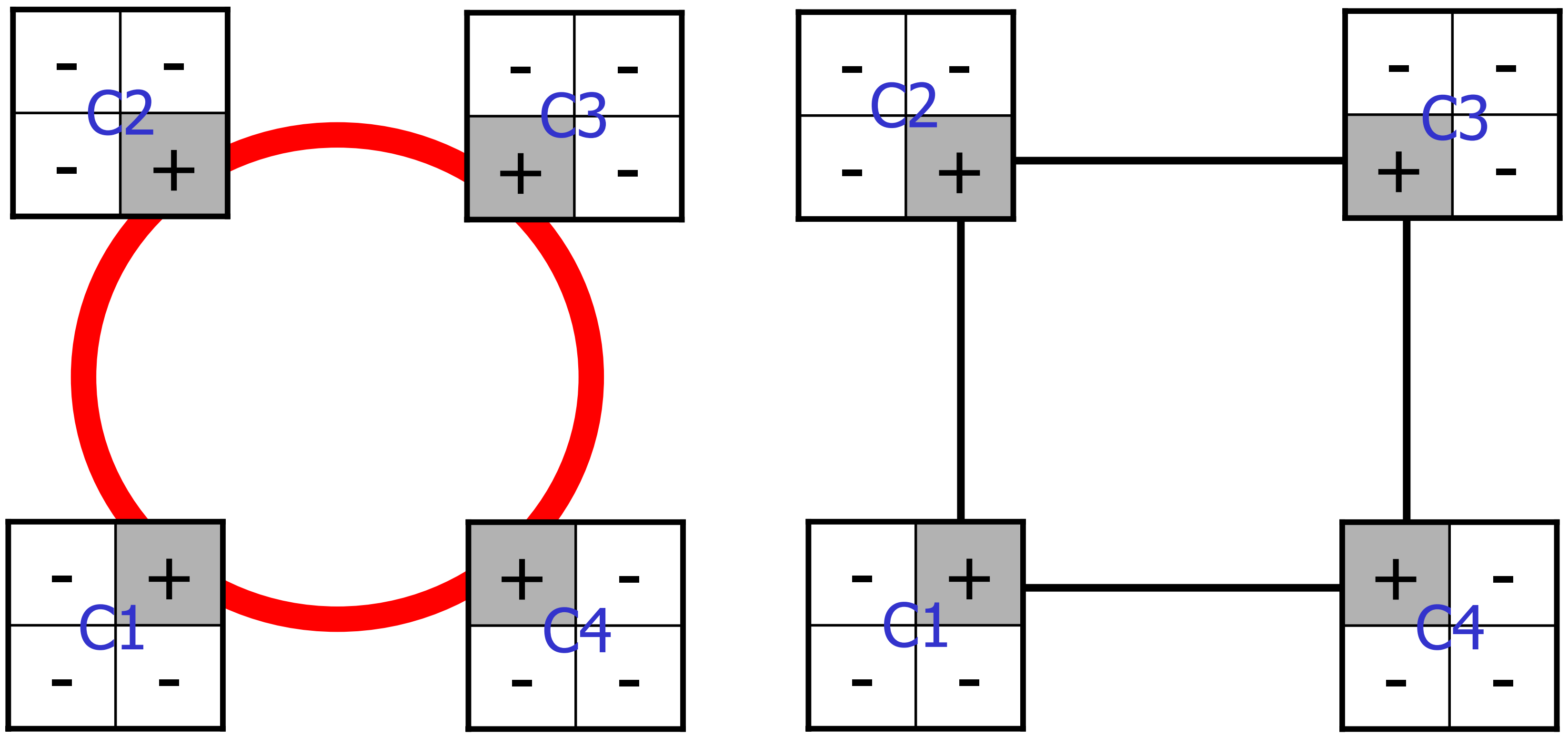
Type Y3=C3





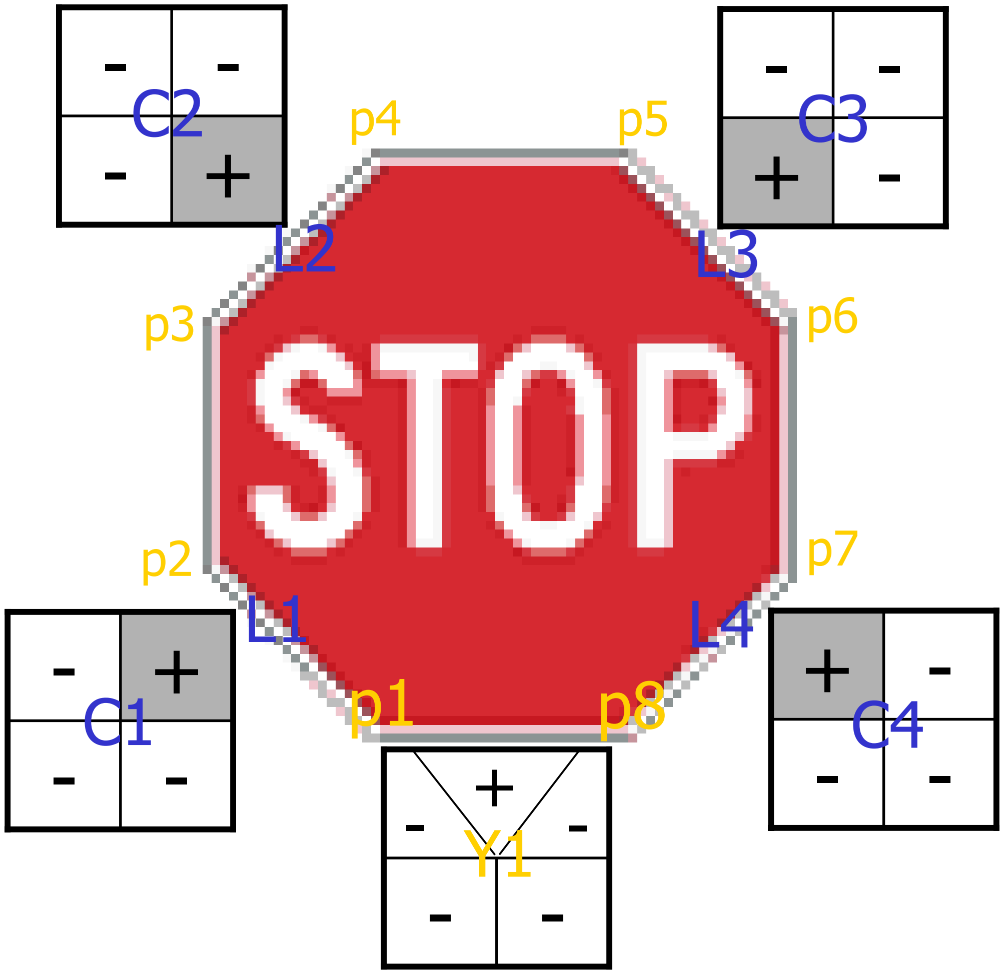
Corner detection

- Masks for circular/rectangular signs:



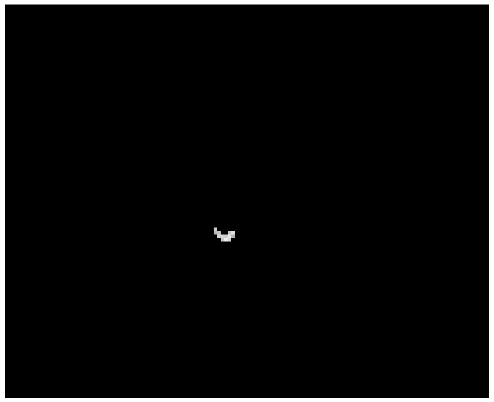
Corner detection

- Masks for stop sign:
  - Point p1 is detected by Y1
  - Point p8 is detected by Y1
  - Segment L1 detected by C1
  - Segment L2 detected by C2
  - Segment L3 detected by C3
  - Segment L4 detected by C4



Corner detection

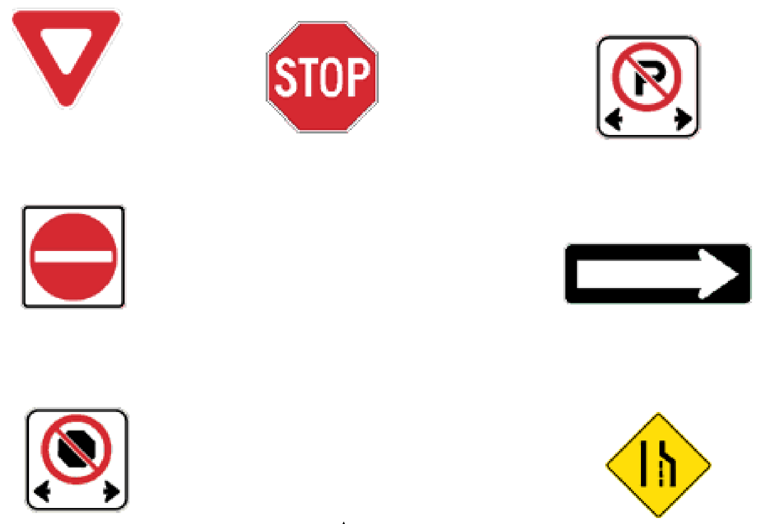
- Center of gravity:
  - The process/mask is not perfect. Multiple corners detected in a neighborhood of each real corner.

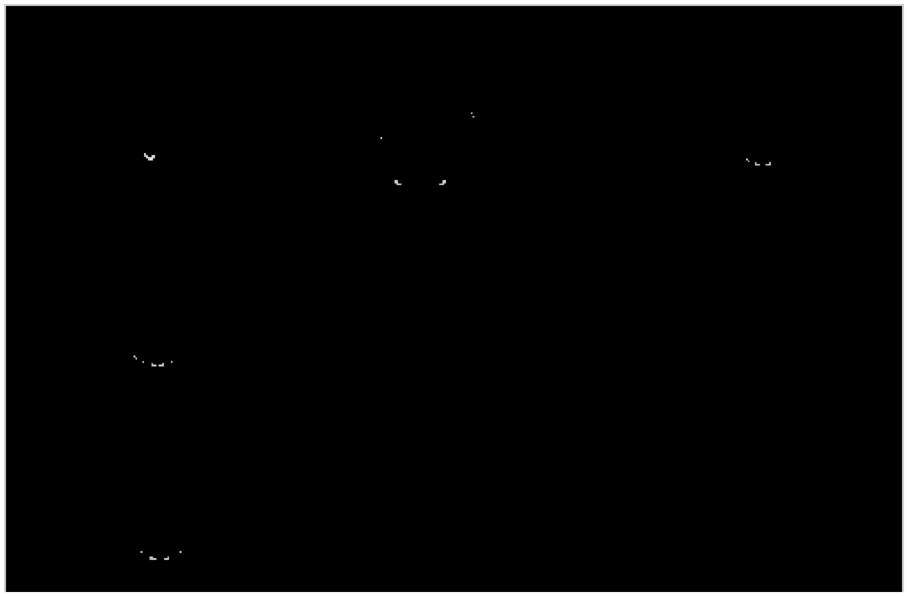

Response of the bottom corner of yield sign.

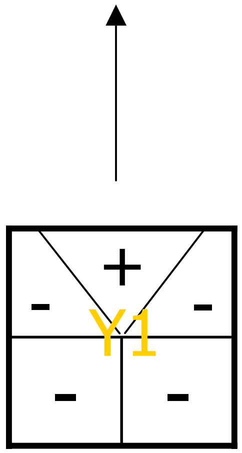
  - Solution?
    - Replace the corners in a neighborhood with the center of gravity.
    - Corners are weighted with the response to convolution at each point.
    - Neighborhood of 7*7 used.

Corner detection

- Response to Y1 mask:

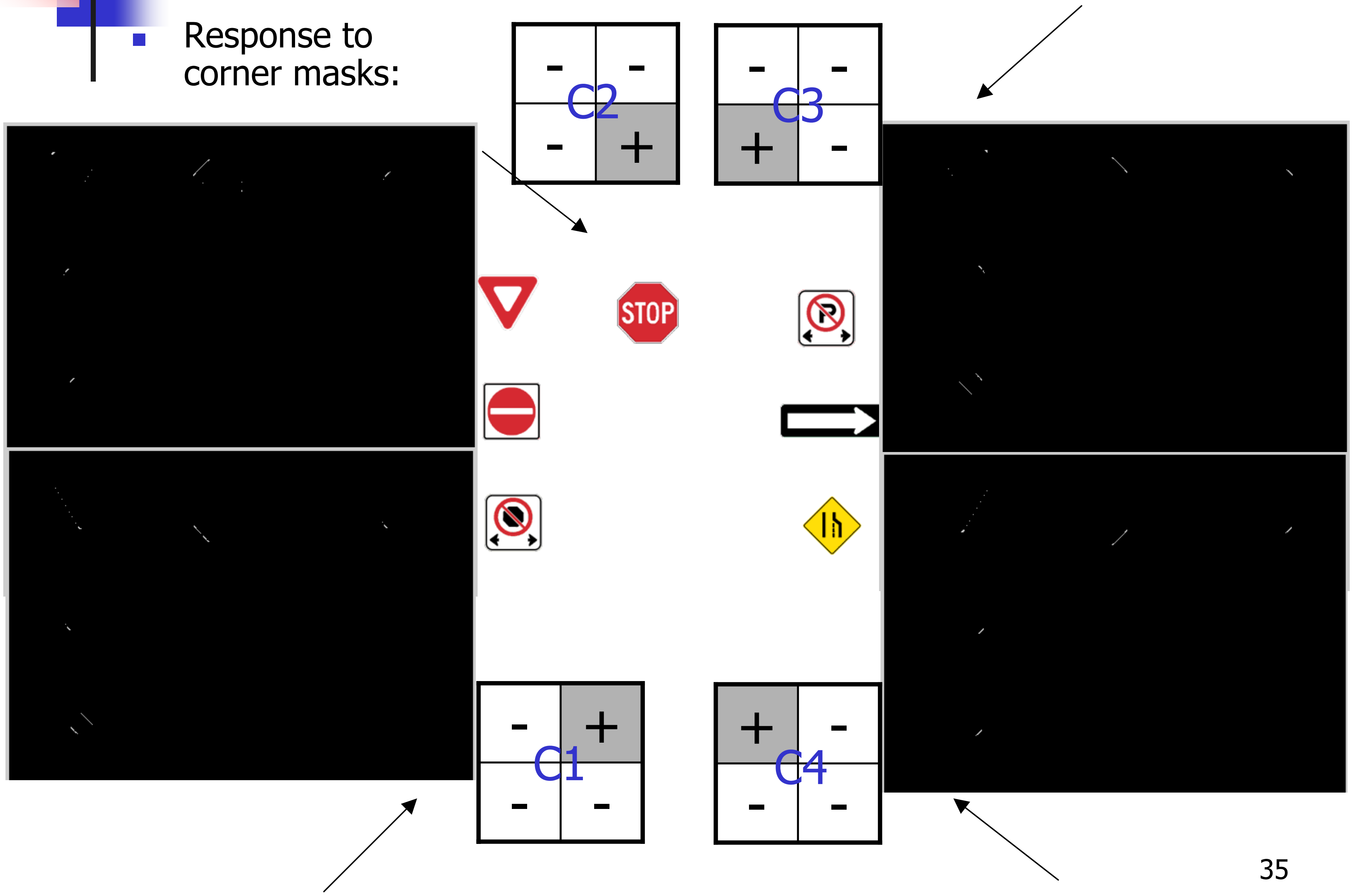






Corner detection

- Response to corner masks:



Corner detection

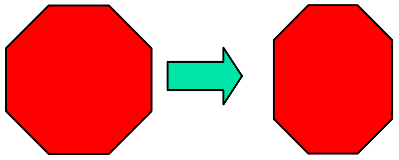
- Number of corners of a different class for a sample image:

Corner set	# of raw corners	# of corners (after center of mass)
Y1	806	218
C1	303	96
C2	279	88
C3	298	97
C4	243	82

C2 corner set

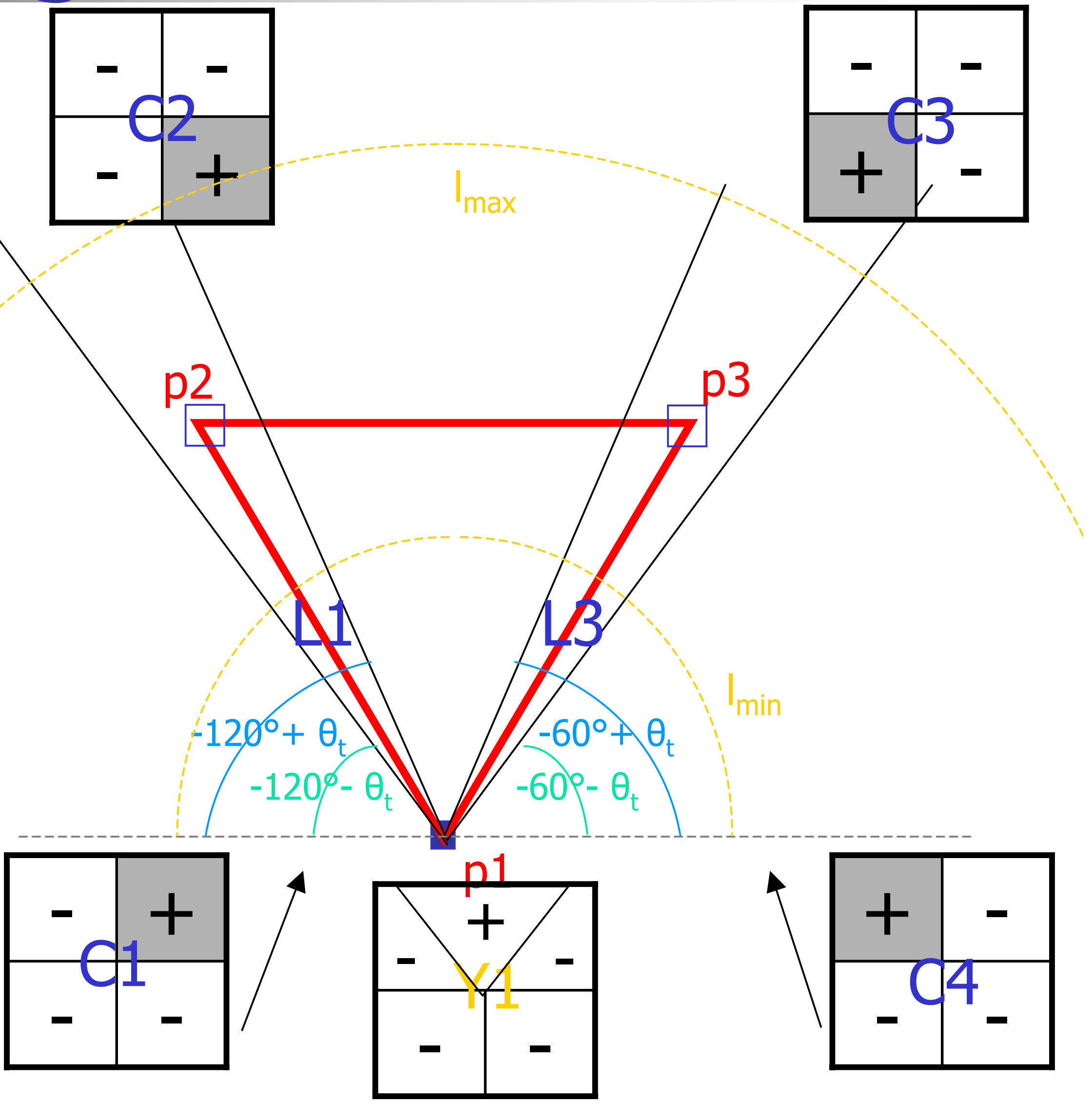
Shape recognition

- Approaches:
  - Geometric hashing:
    - Traffic signs are 2D objects. So is it a mapping in 2D?!! Nop! Projection effect!
    - Can approximate it with parallel projection rather than perspective; considering ratio of sign dimension to the usually long distance.
    - Affine transform of a plane to another plane. Need 3 points to specify the plane.
    - A unique affine transformation exists mapping any 3 non-collinear points in a plane to another triplet in another plane.
    - See [Lamdan '88-1] and [Lamdan '88-2].
    - Can support occlusion; any degree of rotation.
  - Ours is like Interpretation Tree:
    - Geometric constraints limit the number of searches, eliminates sub-trees.
    - Classified corners; e.g. Y1, C1, C2, …
    - Corners are our low-level features. Their distance/angle really used to match features and categorize shapes.
    - The object model is hard-coded into the code with our geometric constraints compared to a database in geometric hashing.



Shape recognition

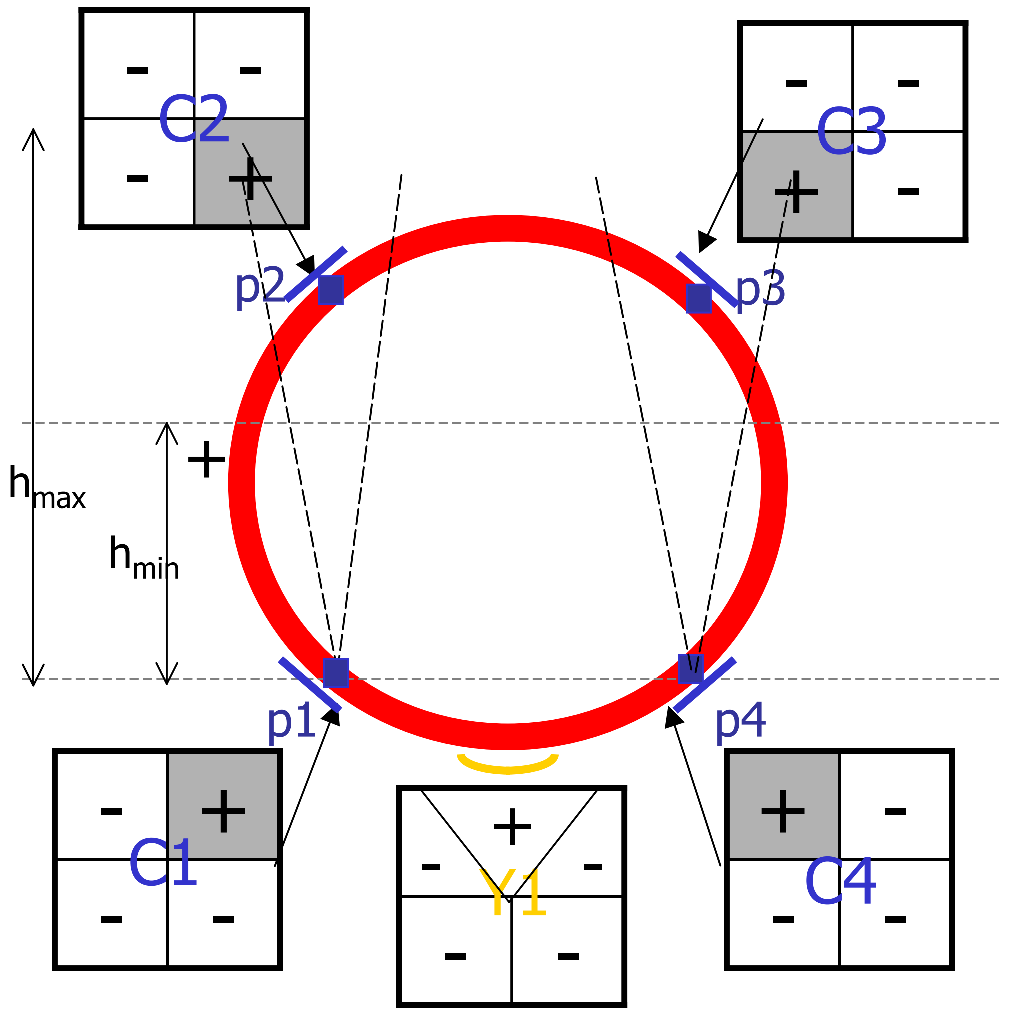
- Triangle detection procedure:
  - Pick a point p1 from Y1 corners set.
  - Find p2 from C2 corner set and p3 from C3 corner set (within angle range) where they can roughly make an equilateral triangle.
  - * Verify the validity of triangle by getting vote from points along L1 in C1 corner set and along L3 from C4 corner set.
  - Check for a non-red region in center of the sign to throw out the case of complete red triangle.



Shape recognition

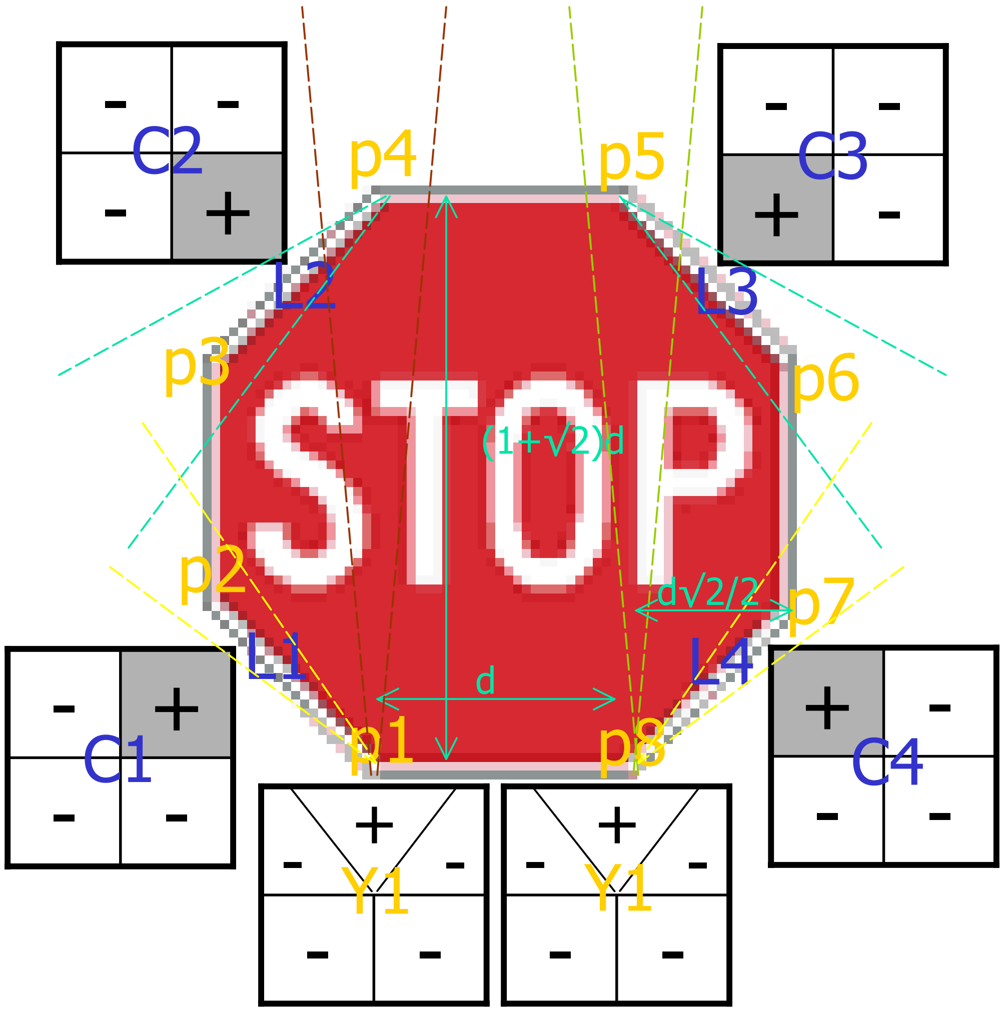
- Circle detection procedure:
  - Pick a point p1 from C1 corners set and p4 from C4 set.
  - Find p2 from C2 corner set within the angle range. Then find p3 from C3 corner set.
  - * Verify the validity of circle (***not done yet***).

- Similar method for rectangle detection. Need different constraints and verification.



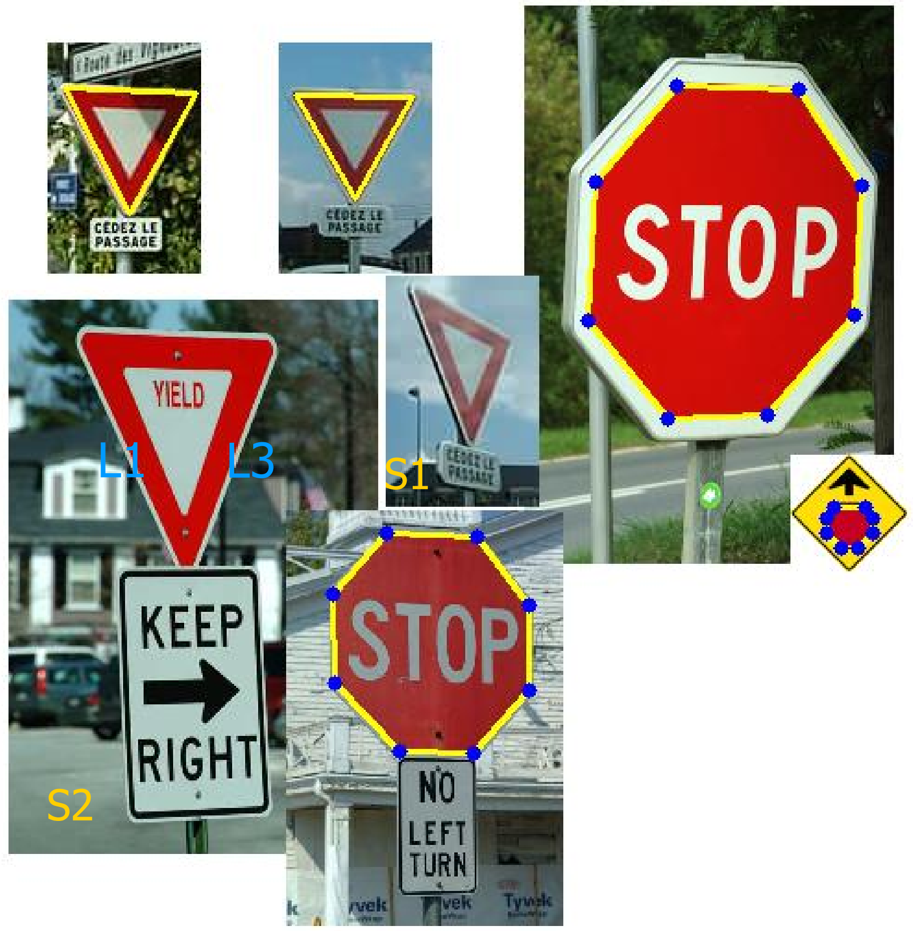
Shape recognition

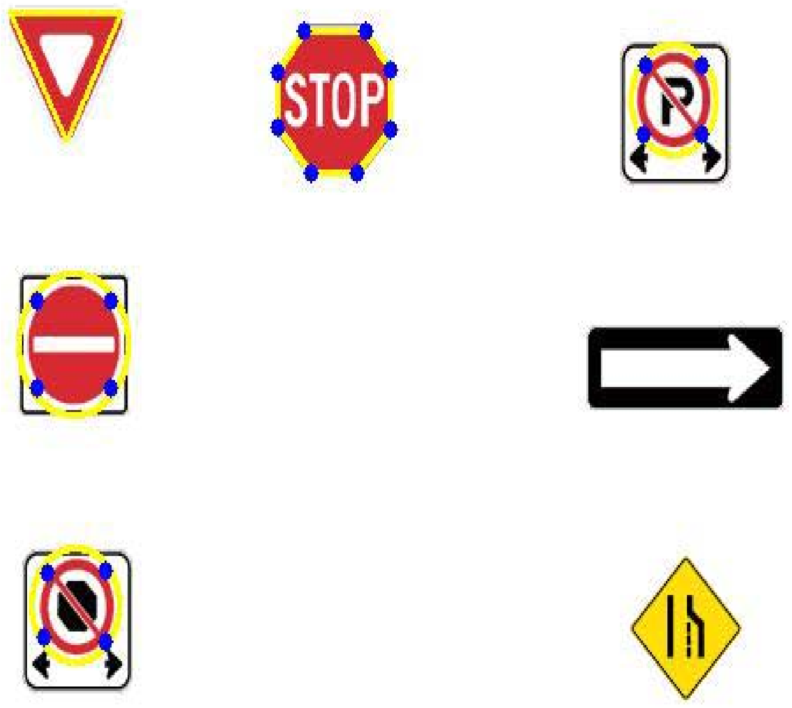
- Stop sign detection procedure:
  - Pick pair of p1 and p8 from Y1 corner set (within the angle tolerance).
  - Find p4/p5 from C2/C3 corner set within the angle tolerance range. They have to satisfy the d distance and their own angle requirement too.
  - Find p6/p7 from C3/C4 corner sets using p5/p8.
  - Find p3/p2 from C2/C1 corner sets using p4/p1.
  - Verify verticality of p3p2 and p6p7 segments.



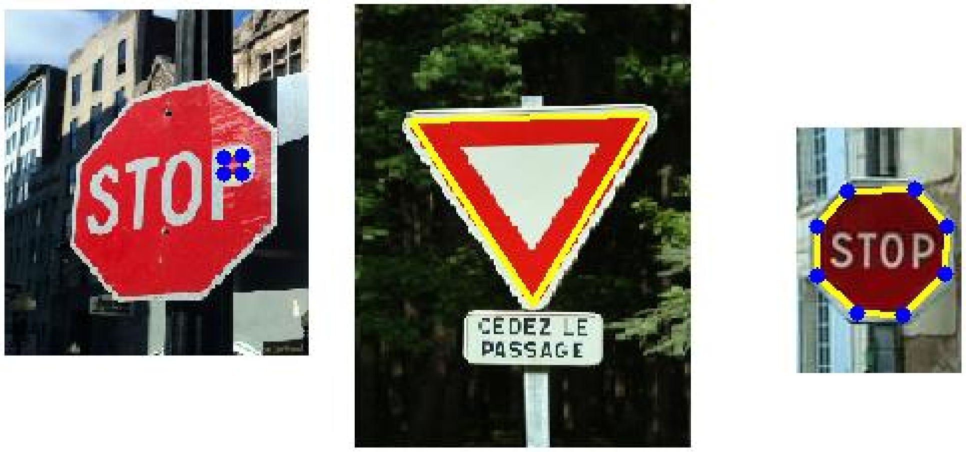
Results

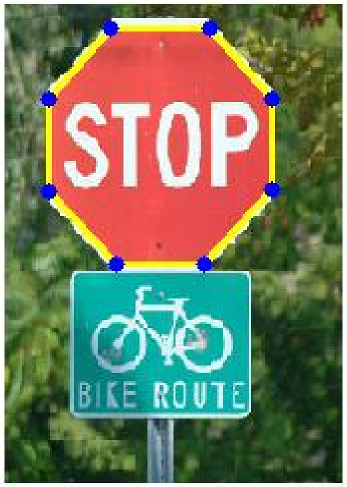
- S1 not detected at all because of it's narrow angle.
- S2 fails verification because L1 and L3 are not detected.



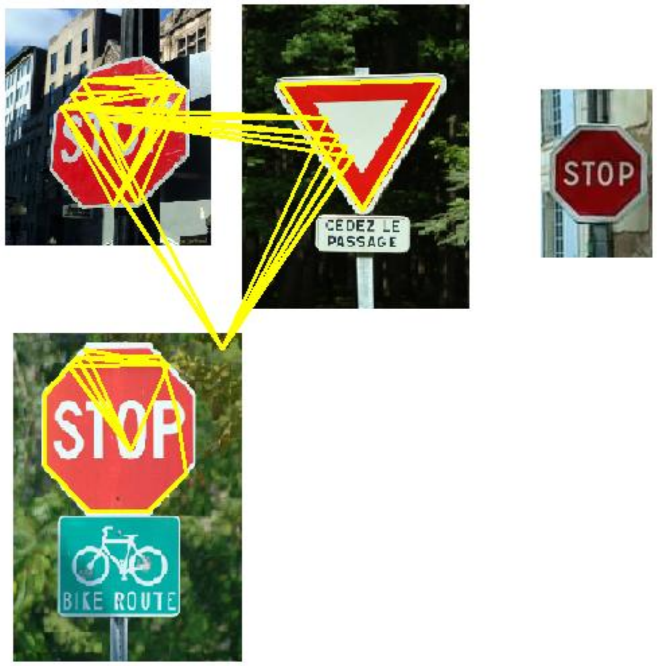


Results



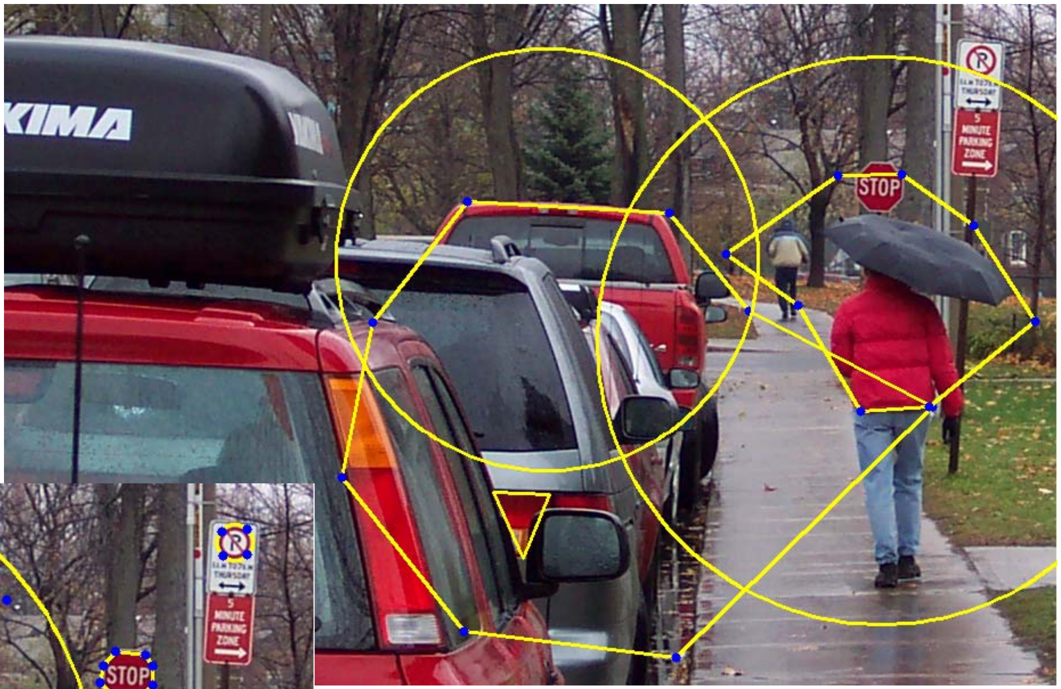


After improving recognition algorithms and implementing triangle verification.

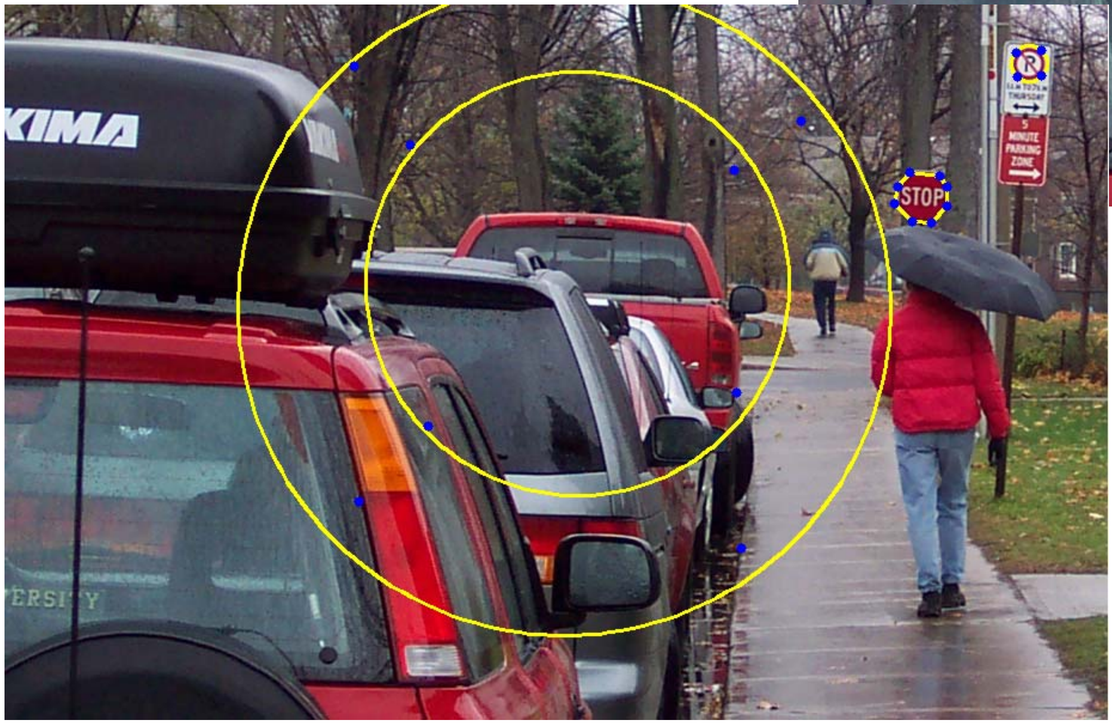


An early result with many outliers.

Results



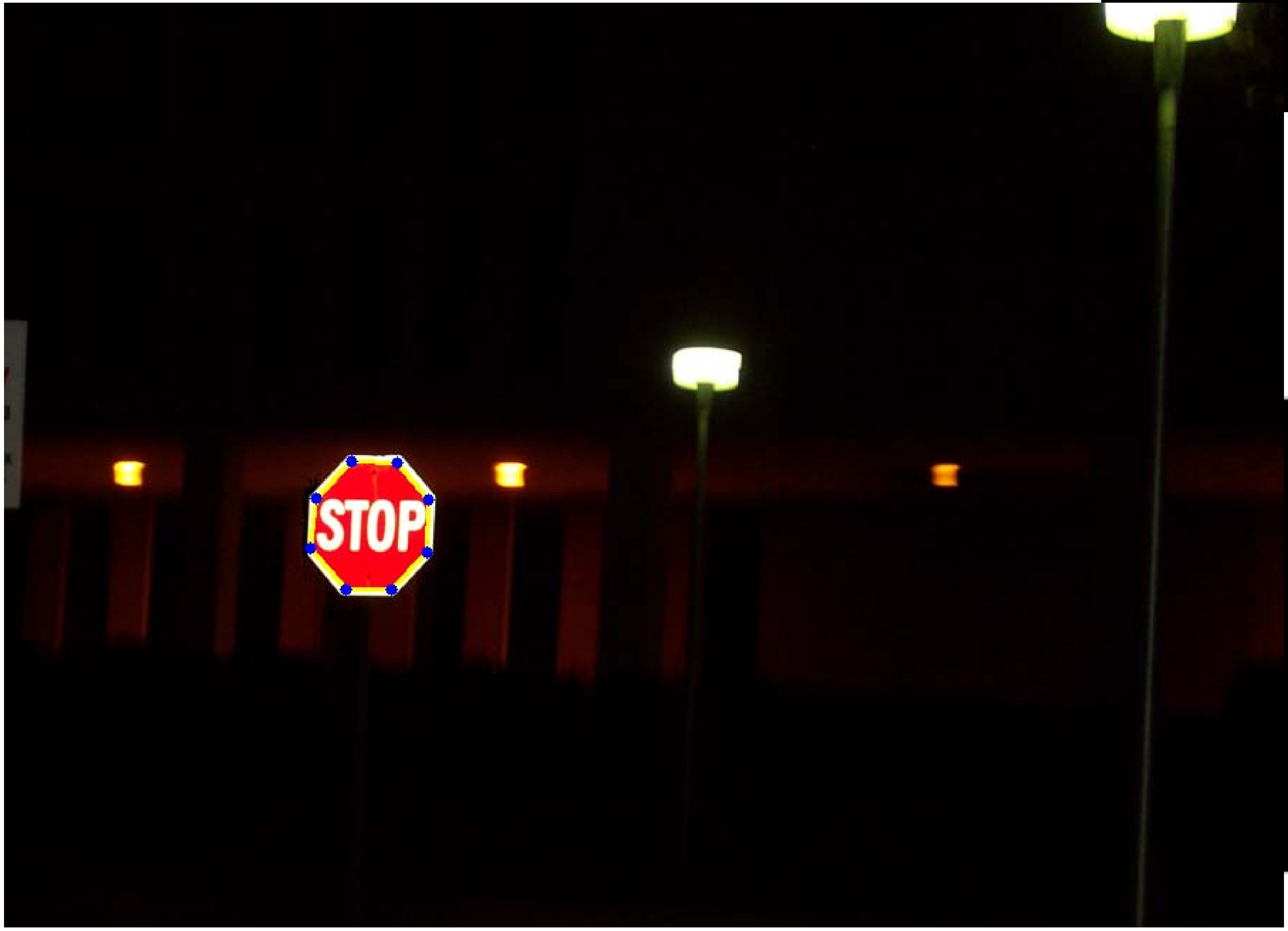
An early result with many outliers.



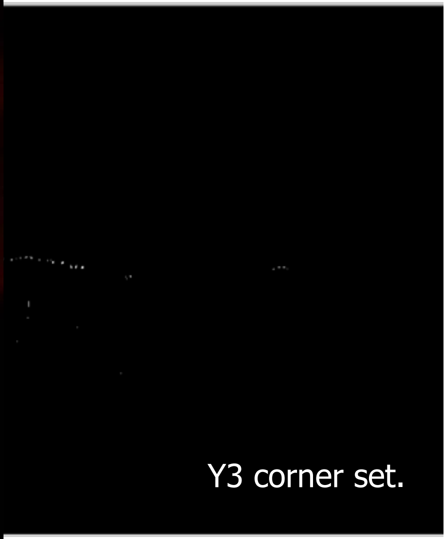
After improving recognition algorithms and implementing triangle/stop sign verification. Still no circle verification.

Results

- See high number of corners in the Y3 corner set.

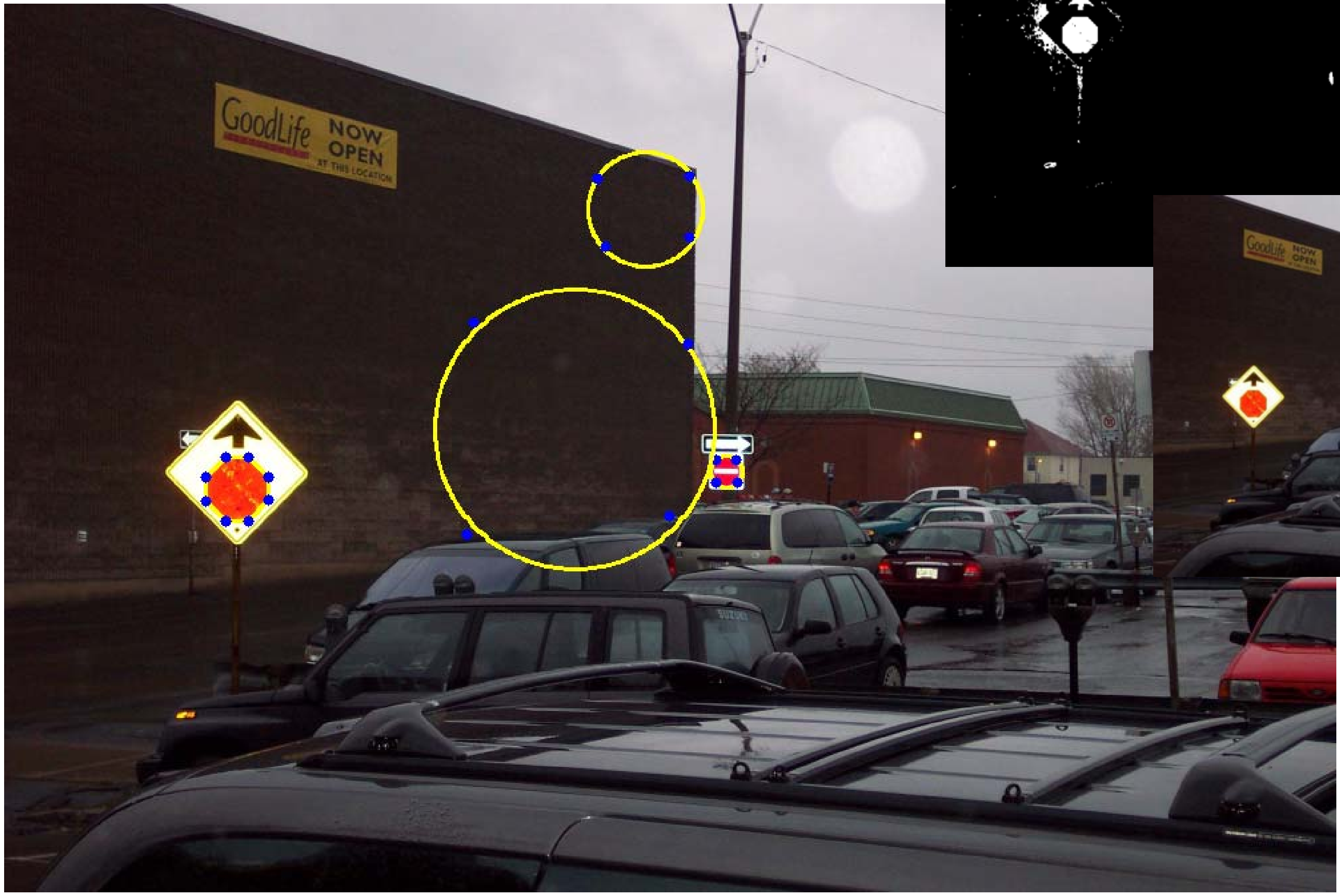


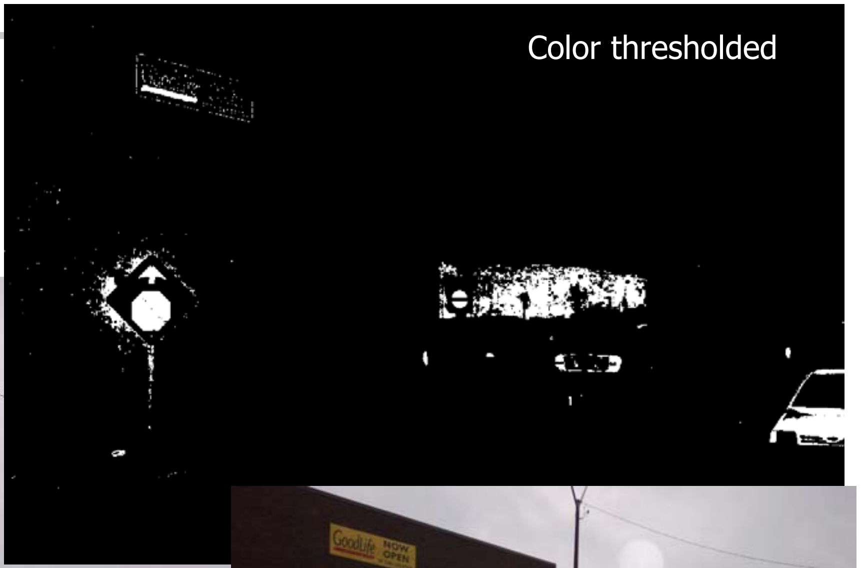
Color thresholded



Y3 corner set.

Results

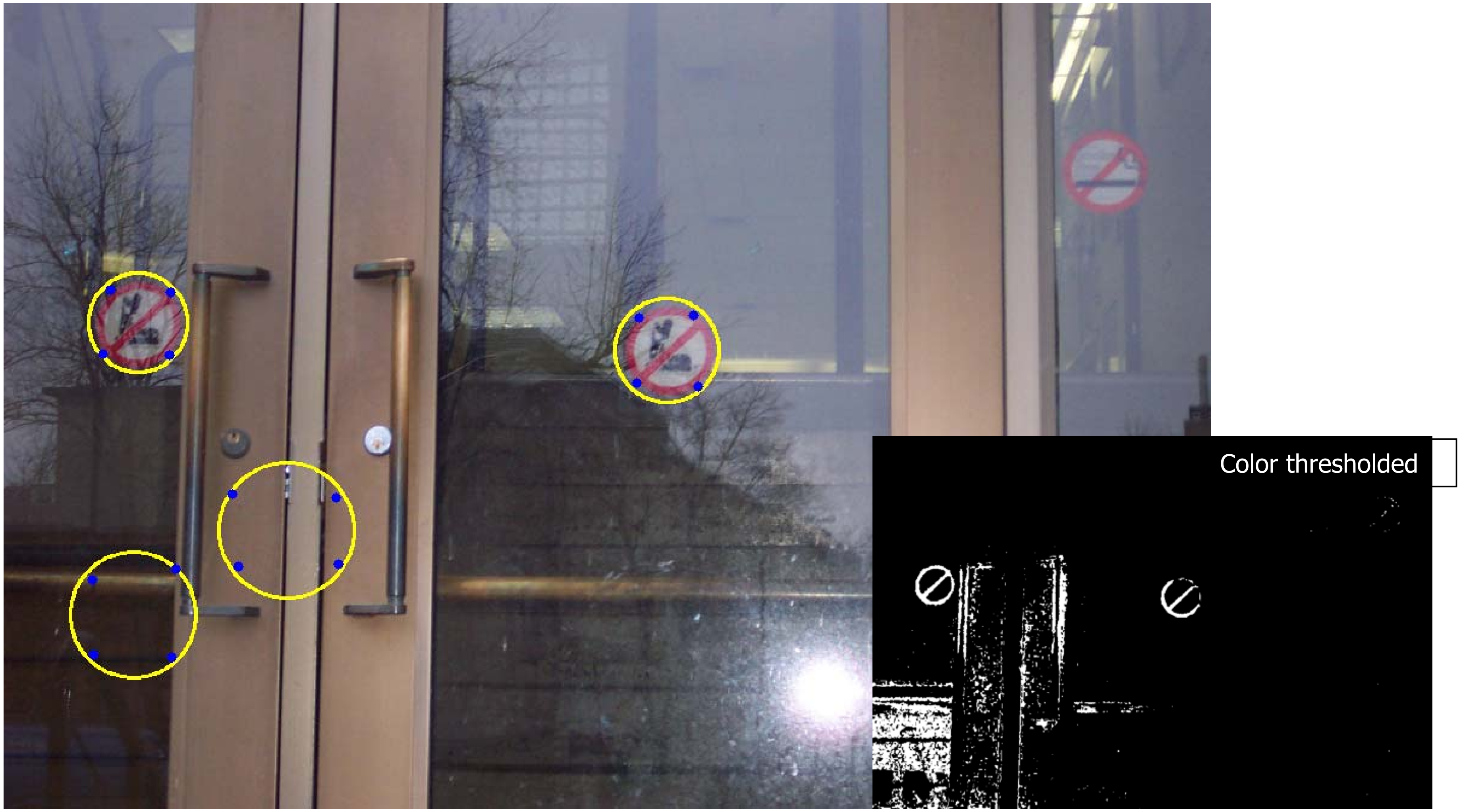



Color thresholded

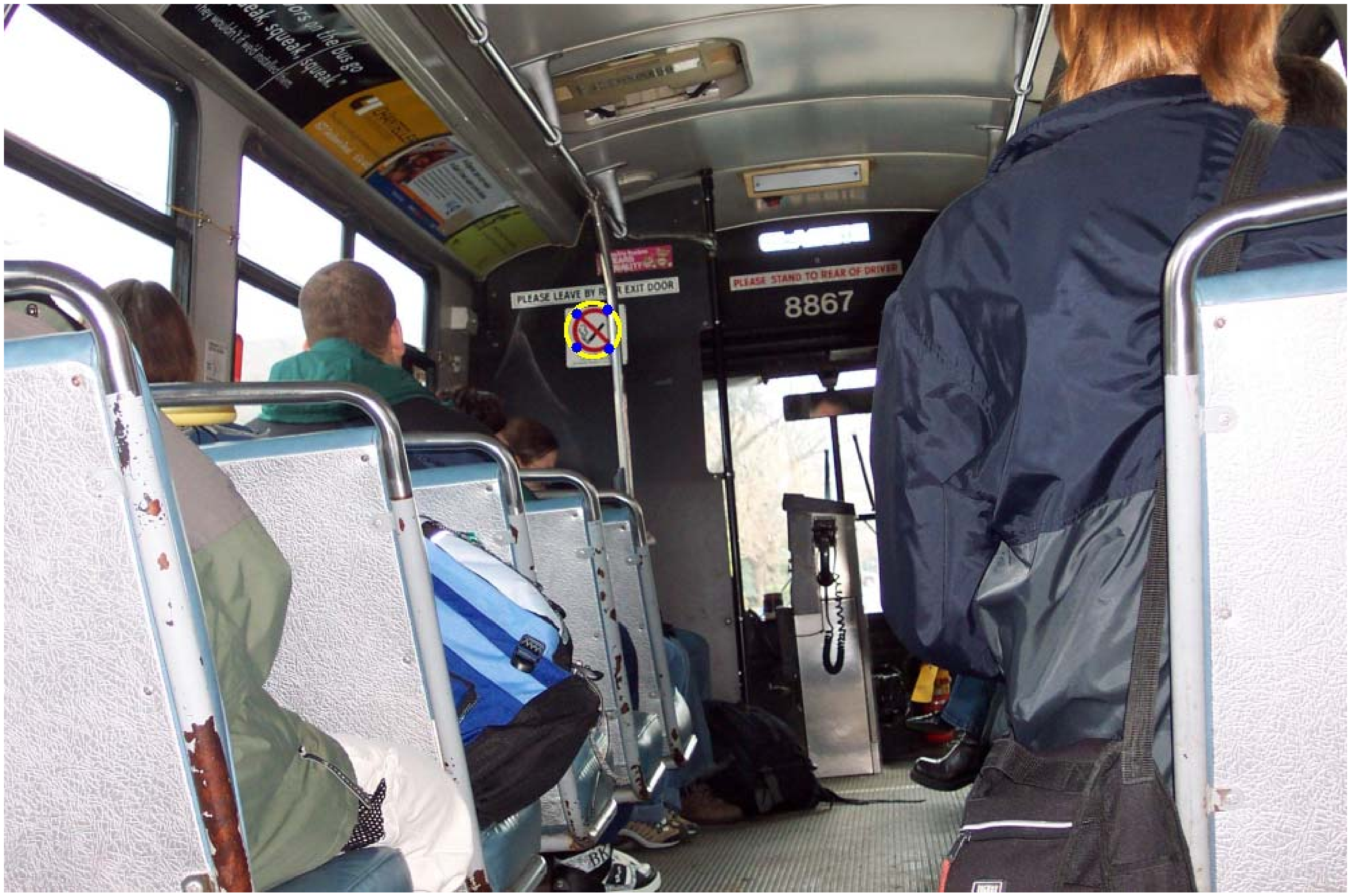
Original image

Results

- The “no smoking” sign is not detected because of bad color thresholding.

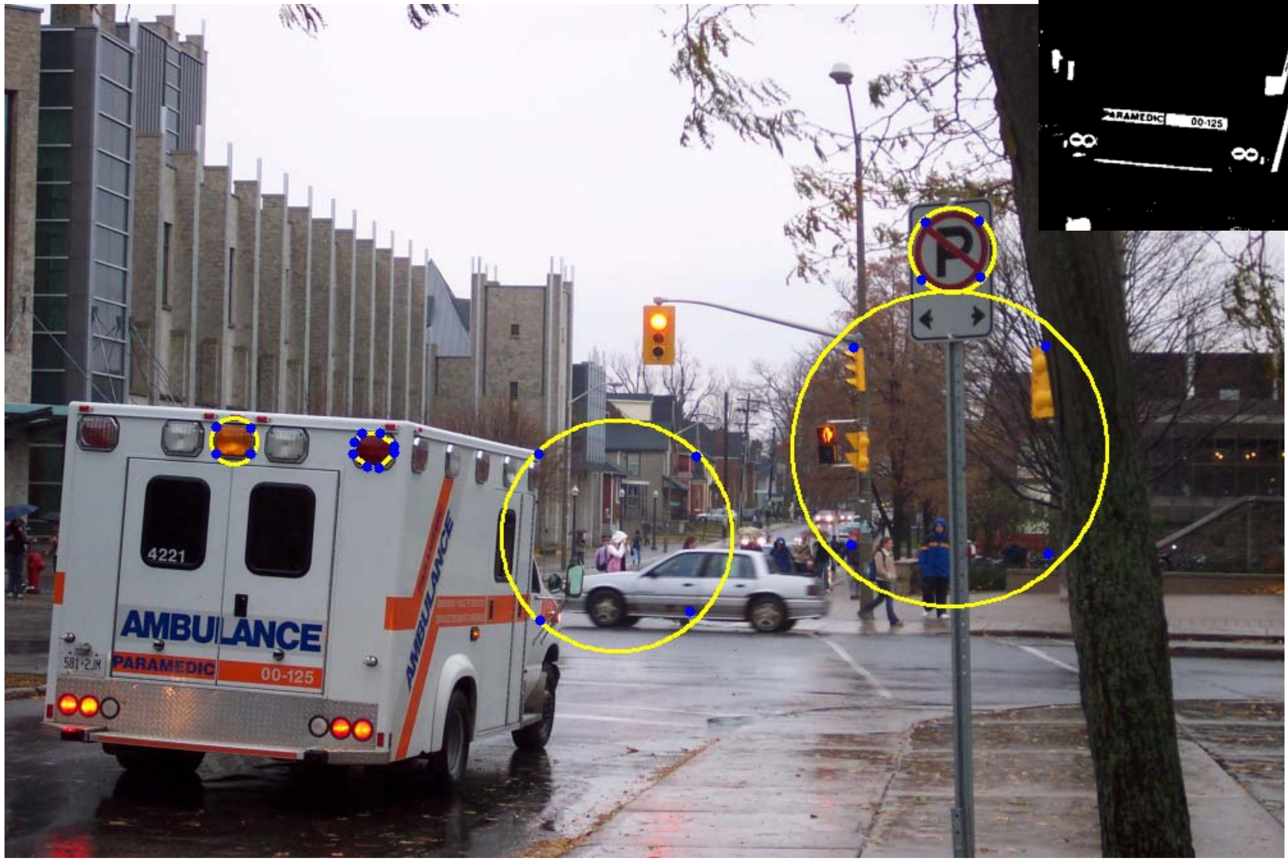


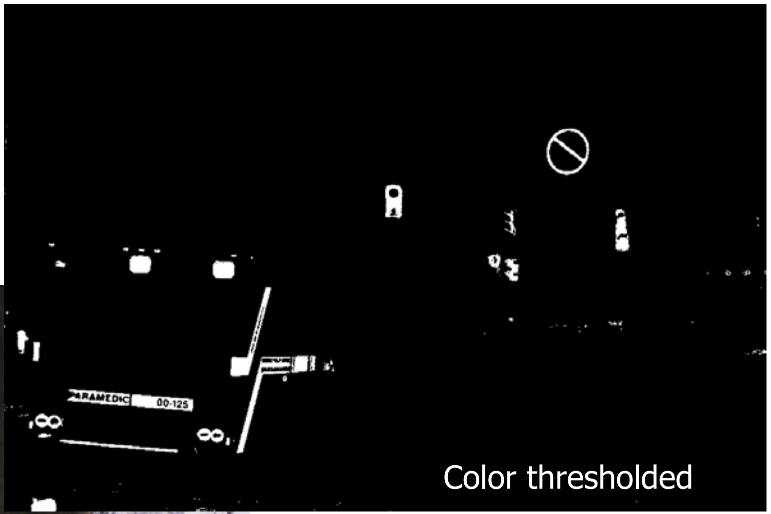
Results


PLEASE LEAVE BY REAR EXIT DOOR
PLEASE STAND TO REAR OF DRIVER
8867

Results

- See outliers:

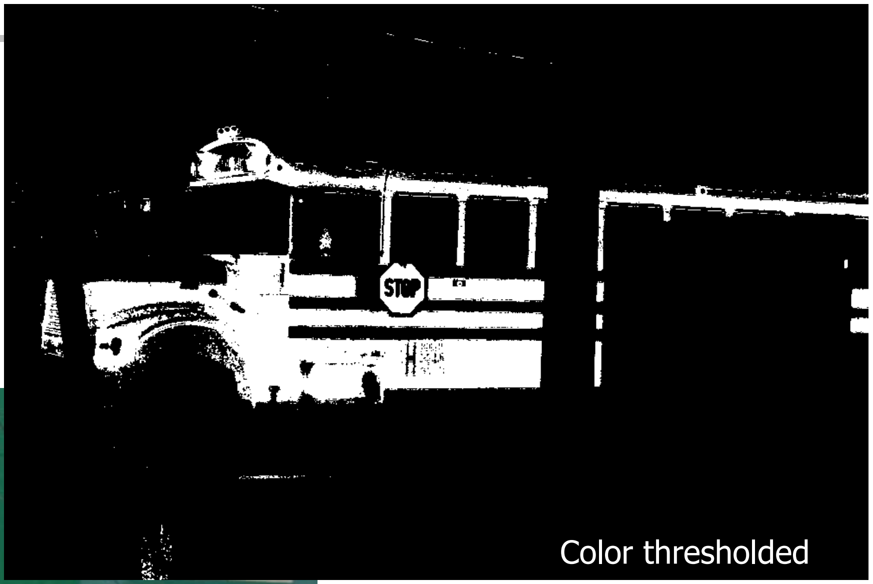


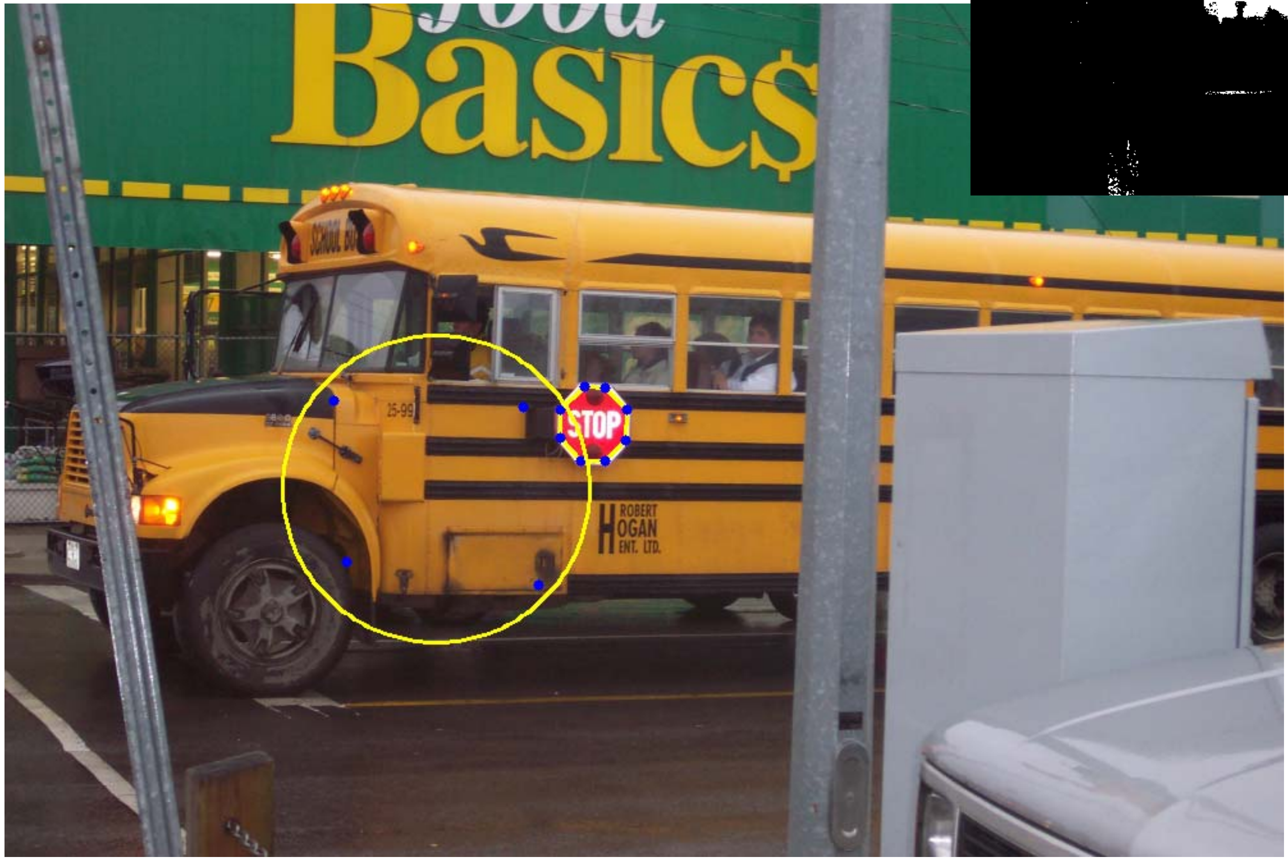


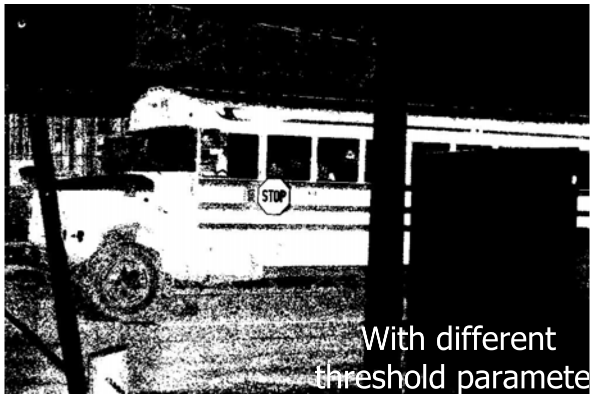
Color thresholded

Results

- Bigger portion of bus yellow color should have been suppressed in color segmentation phase.

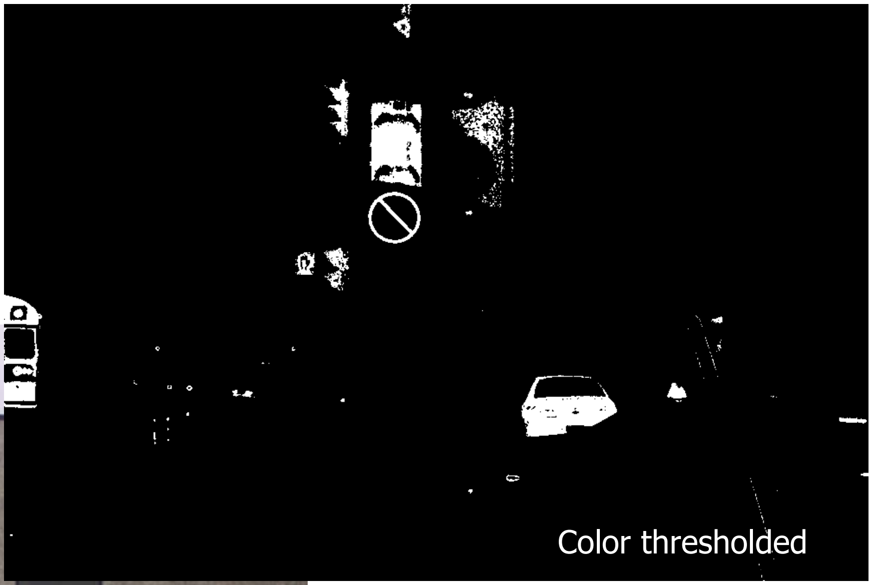

Color thresholded

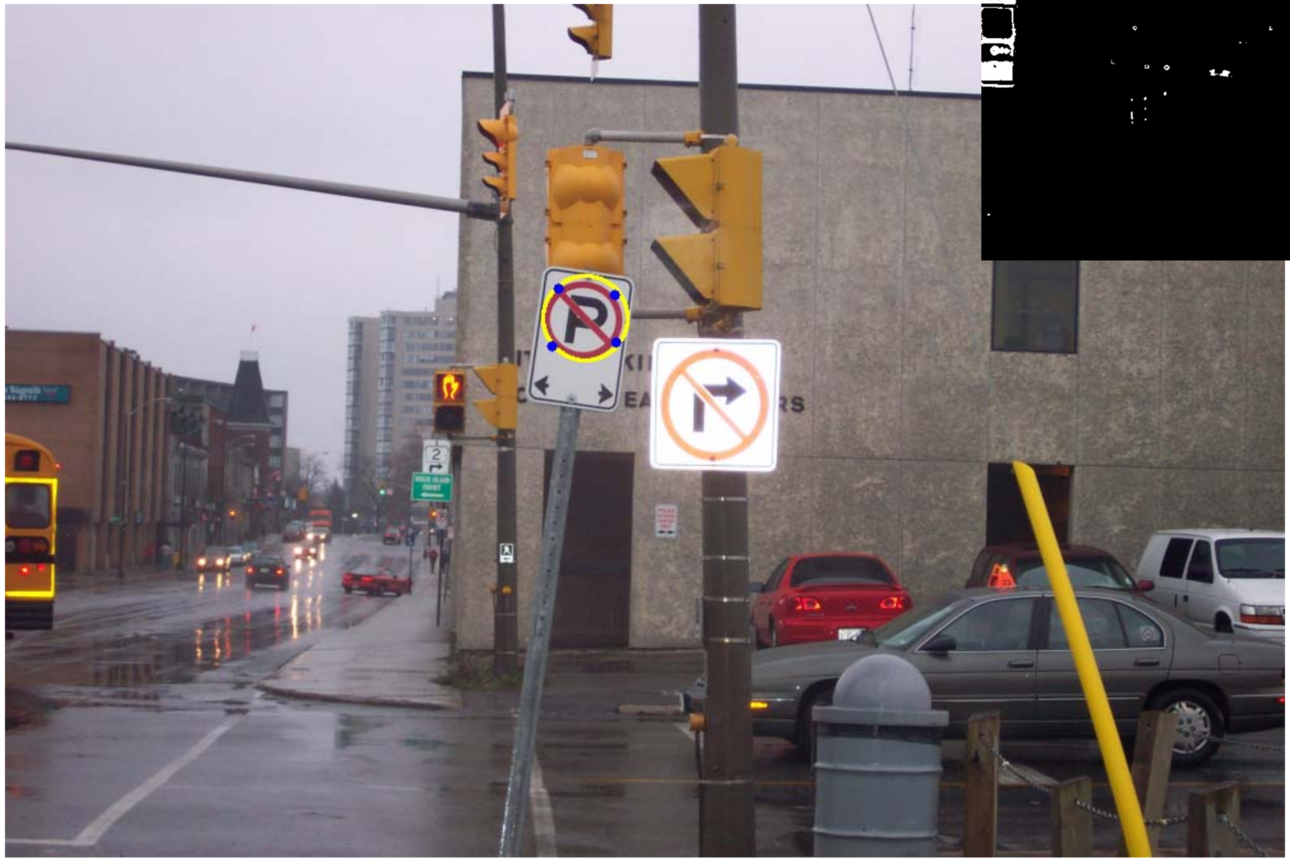



With different threshold paramete

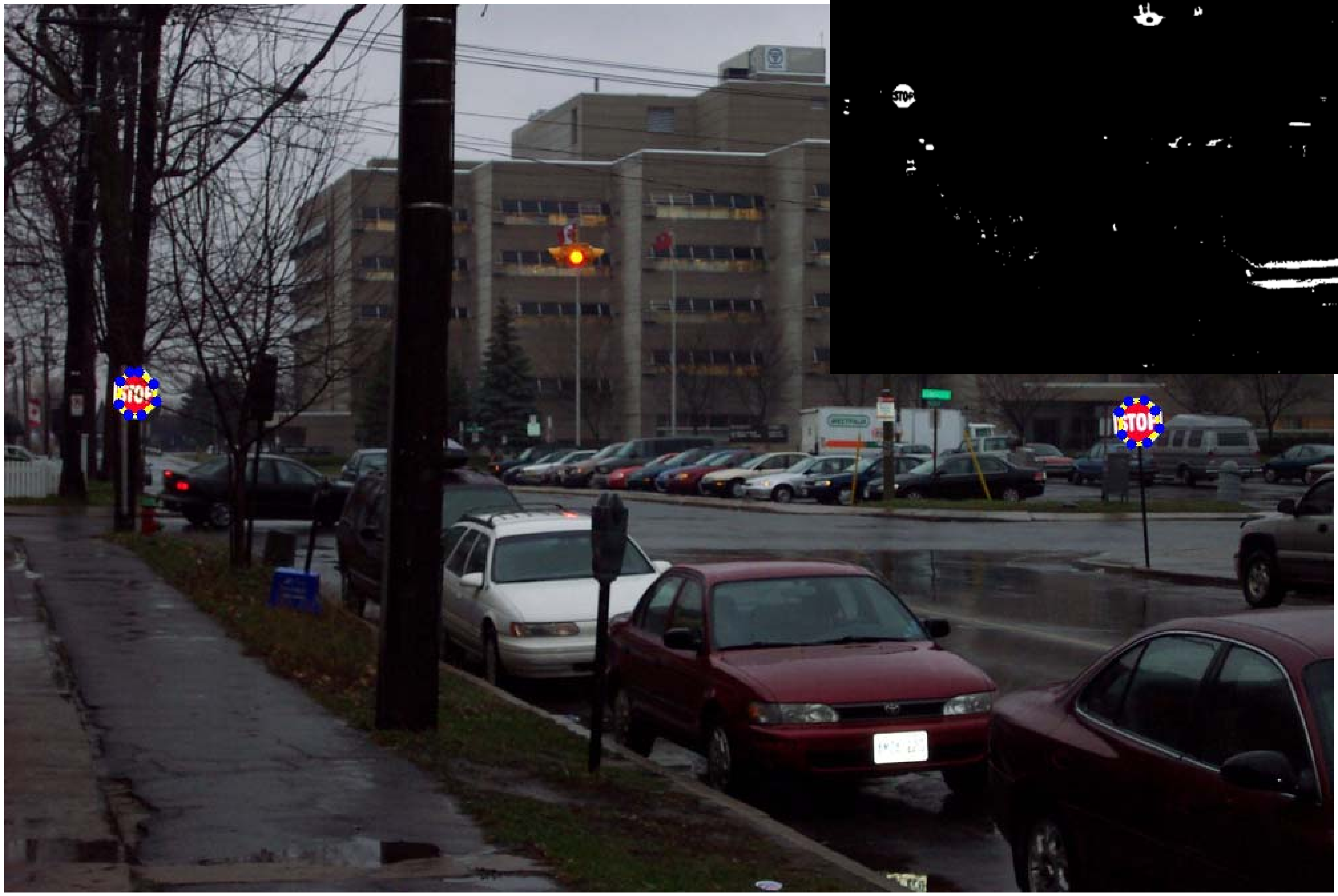
Results

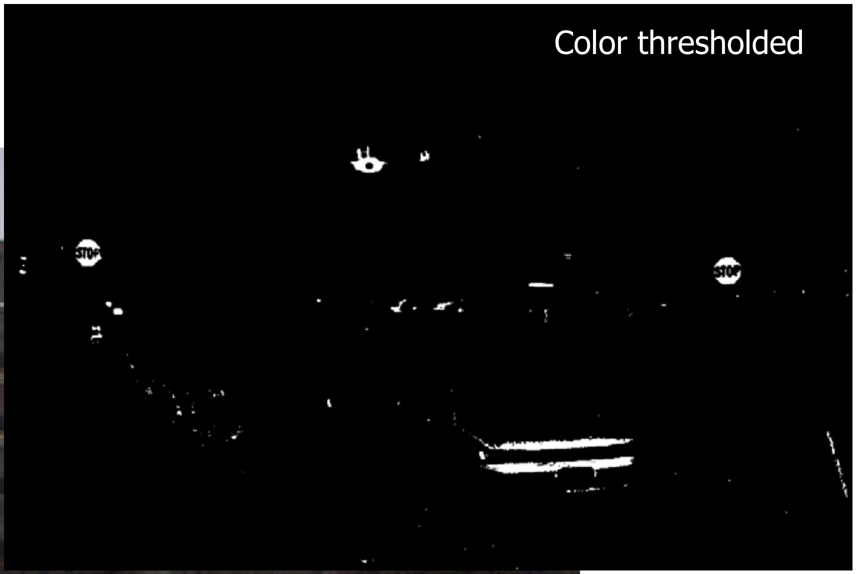
- “No right turn” sign is not detected because of bad color thresholding.





Results



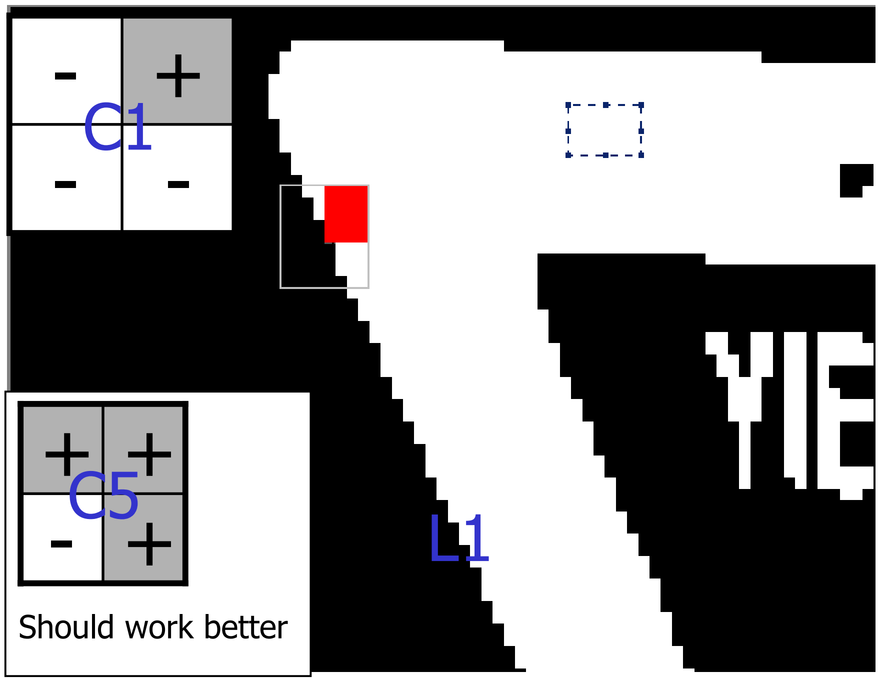


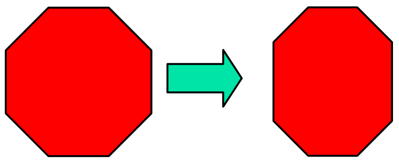
Observations

- Issues:
  - Center of mass calculation can cause shift of real corner. When the effect of multiple corners added together, it could screw up the detection specially for smaller signs.
  - Not easy to extend it to large number of shapes.
    - Code has grown fast; use OpenCV only for handling image file; more than 2000 lines ~ 60-70 hrs.
    - Many parameters to adjust. After each change, need to test the whole samples again to see it’s not breaking the previously working ones.
    - Difficult to debug.
  - How to handle outliers:
    - Not mentioned in the original paper. They probably rely on sign classifier to throw them out because their network won’t detect the pictographic part.
    - Thought about Color histogram after Mohan’s presentation… But won’t work for rotation and scaling.
    - Area of red pixels for stop sign?! Ideally 70%.
    - Detecting line segments for yield sign.
    - Circular signs could become elliptic.

Observations

- Mask doesn’t work for some slopes [was not intended to work but we were using it]:
  - C1 mask can’t detect L1 because of number of white pixels falling into the NW/SE non-corners quarter.
  - Need to find another way to verify yield sign.
- Importance of color thresholding parameters. Severely affects number of detected corners.
- More tolerance of parameters in horizontal rather than vertical direction.





Future work

- Circular sign detection and verification (different verification methods; e.g. for "do not enter" than "no left turn")
- Revising yield sign verification
- Revisiting color thresholding and search area parameters (allowing wider angle for short distance)
- Noisy images
- Algorithm optimization (in convolution calculation, shape detection, …)

- Beyond the scope of this project:
  - Signs of other colors and shapes
  - Pictogram classification
  - Considering occlusion, higher degree of rotation and projection
  - Mixing with other methods (e.g. geometric hashing)
- After all, more improvement on GPS systems could make traffic sign detection useless!!!

References

- [Escalera '97] A. Escalera, L. Moreno, M. Salichs, J. Armingol, "Road Traffic sign detection and classification," *IEEE transactions on industrial electronics*, vol. 44, no. 6, Dec. 1997.
- [Rangarajan '89] K. Rangarajan, M. Shah, and D. Van Brackle, "Optimal corner detector," *Computer Vision, Graphics and Image Processing*, vol. 48, no. 2, pp 230-245, Nov. 1989.
- [Sandoval '00] H. Sandoval, T. Hattori, S. Kitagawa, Y. Chigusa, "Angle-dependent edge detectionfor traffic sign recognition," in *Proceedings of the IEEE Intelligent Vehicles Symposium 2000*, Dearborn (MI), USA, Oct. 3-5, pp. 308-313.
- [Trapper '97] F. Heinze, L. Schafers; C. Scheidler, W. Obeloer, "Trapper: eliminating performance bottlenecks in a parallel embedded application," *IEEE Concurrency [see also IEEE Parallel & Distributed Technology]* , vol. 5, issue 3, July-Sept. 1997, pp. 28 –37.
- [Kehtarnavaz '95] N. Kehtarnavaz, A. Ahmad, "Traffic sign recognition in noisy outdoor scenes," *Proceedings of the Intelligent Vehicles '95 Symposium*, 25-26 Sept. 1995, pp. 460-465.
- [Kang '94] D.S. Kang, N.C. Griswold, N. Kehtarnavaz, "An invariant traffic sign recognition system based on sequential color processing and geometrical transformation,", Proceedings of the IEEE Southwest Symposium on Image Analysis and Interpretation, 21-24 April 1994, pp. 88-93.
- [Lamdan '88-1] Y. Lamdan, H.J. Wolfson, "Geometric Hashing: A General And Efficient Model-based Recognition Scheme," in *Proceedings of Second International Conference on Computer Vision*, , December 5-8, 1988, pp. 238-249.
- [Lamdan '88-2] Y. Lamdan, J.T. Schwatrtz, H.J. Wolfson, "On recognition of 3-D objects from 2-D images," in *Proceedings of IEEE International Conference on Robotics and Automation*, April 24-29, 1988, pp. 1407-1413.

- Thank Andy for camera!
- Update about masks used for stop sign.