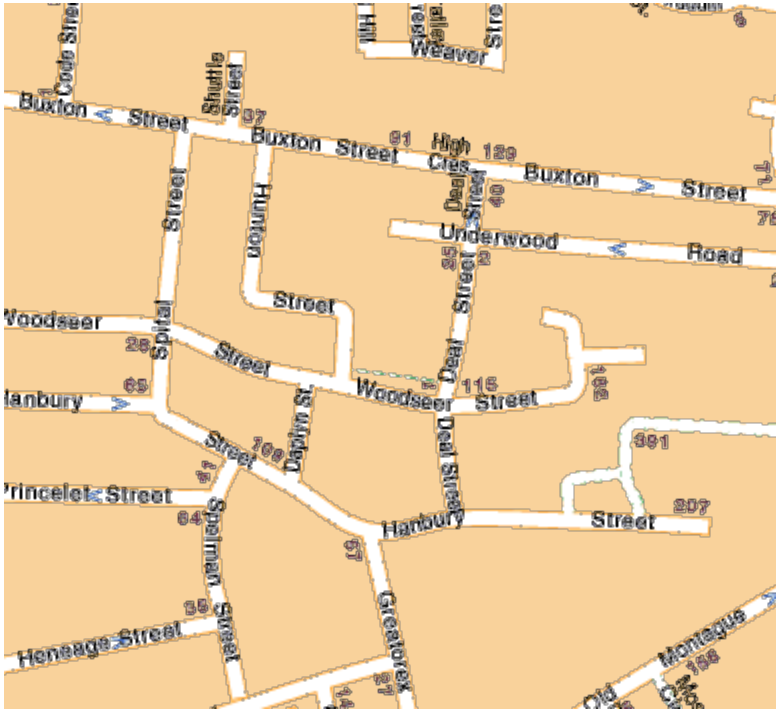
Classification of Line and Character Pixels on Raster Maps Using Discrete Cosine Transformation Coefficients and Support Vector Machines

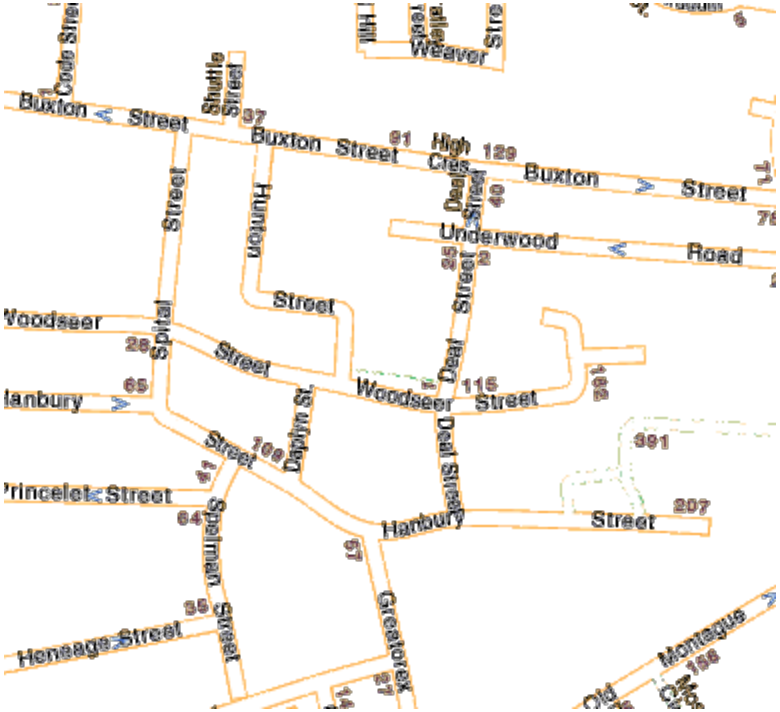
The Problem

- To understand the information on raster maps
  - How? Recognize the line and characters on the raster map for further processing



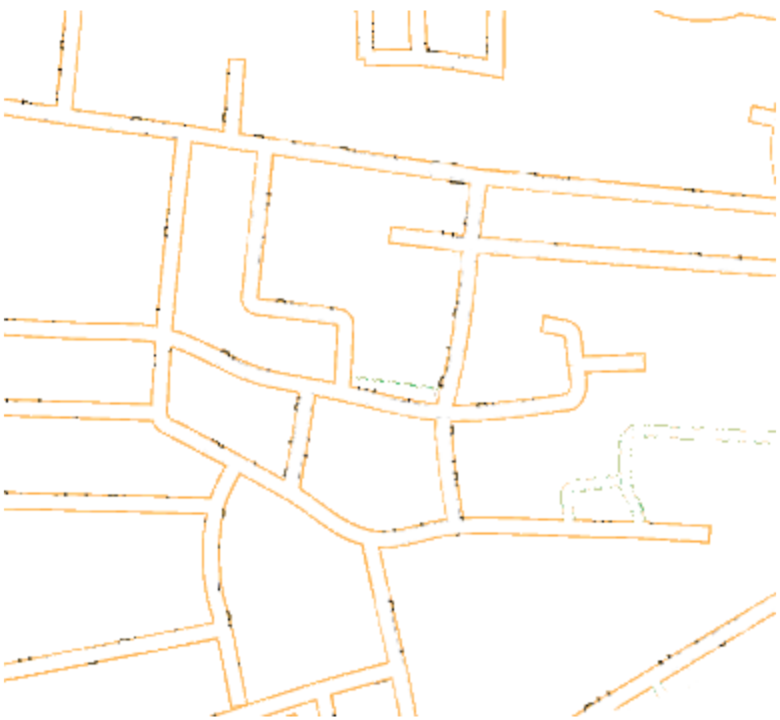
The Problem

- To understand the information on raster maps
  - How? Recognize the line and characters on the raster map for further processing



The Problem

- To understand the information on raster maps
  - How? Recognize the line and characters on the raster map for further processing



Related Work

- Steps to recognize the lines and characters:
  - **FIND AREAS** of characters
  - For each area, **SEPARATE** and **REBUILD** lines and characters
  - Send characters to **Optical Character Recognition** component
  - Send lines to **Vectorization** component
- These steps are interrelated

Related Work

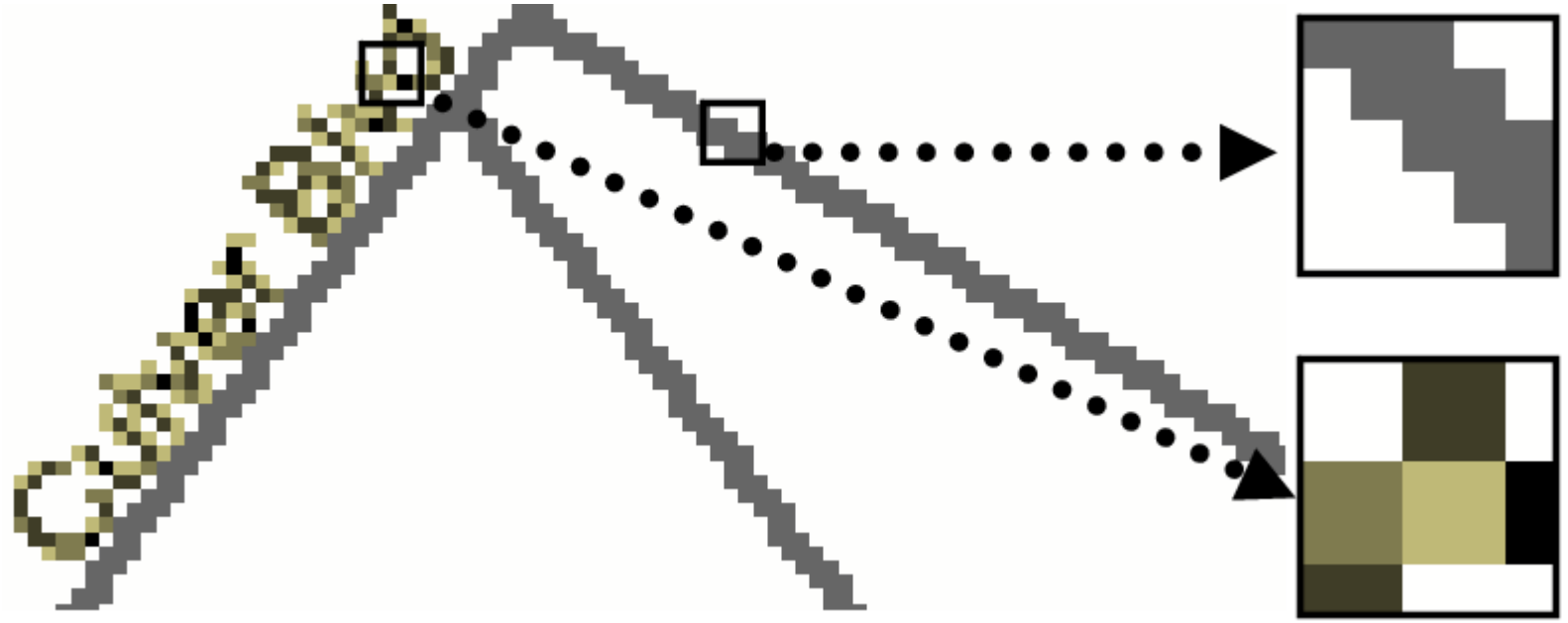
- Some of the work assume that the line and character pixels are not overlapping (Bixler00, Fletcher88, Velazquez03)
- Li et al. work in local areas to separate the characters from lines
- Cao et al. use the different length of line segments to separate characters from line arts

Related Work

- They all based on geometric properties
  - The size of a character
  - The size of a word (string)
  - The size of the gap between characters
  - The size of the gap between words
  - etc.
- They assume the foreground can be easily separated from the background

Our Approach

- We use texture classification approach to classify pixels on the raster maps



Our Approach

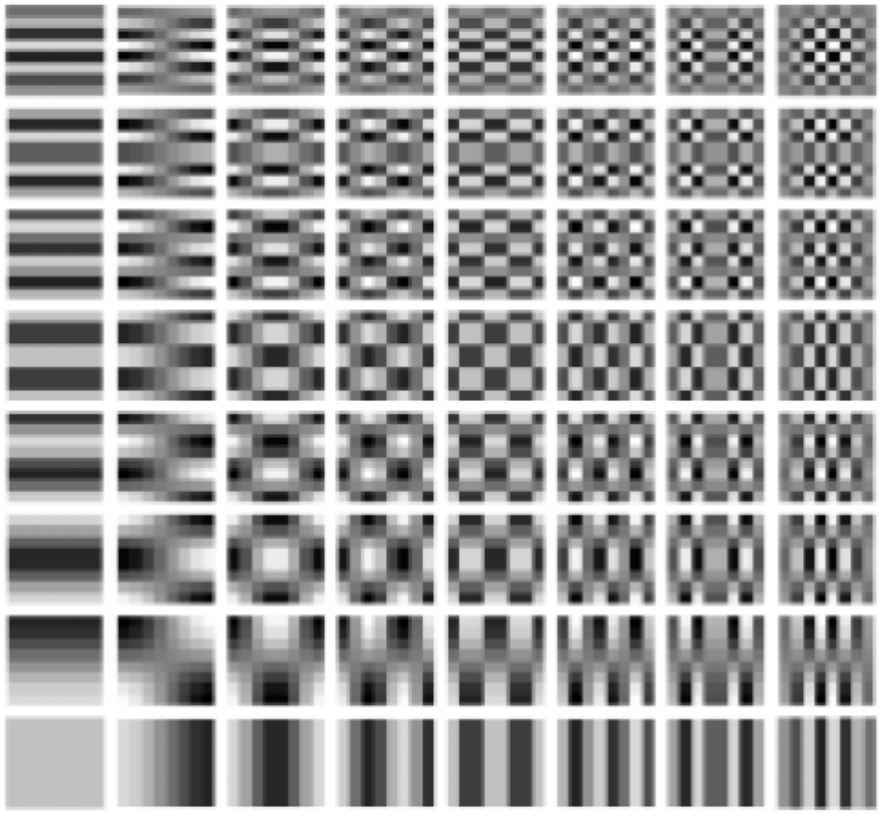
- Features:
  - Discrete Cosine Transformation (DCT) coefficients
- Classifier:
  - Support vector machine

Discrete Cosine Transformation

- DCT – Discrete Cosine Transformation
  - DCT is closely related to the discrete Fourier transform (DFT)
  - The discrete cosine transform (DCT) is a technique for converting a signal into elementary frequency components

Discrete Cosine Transformation

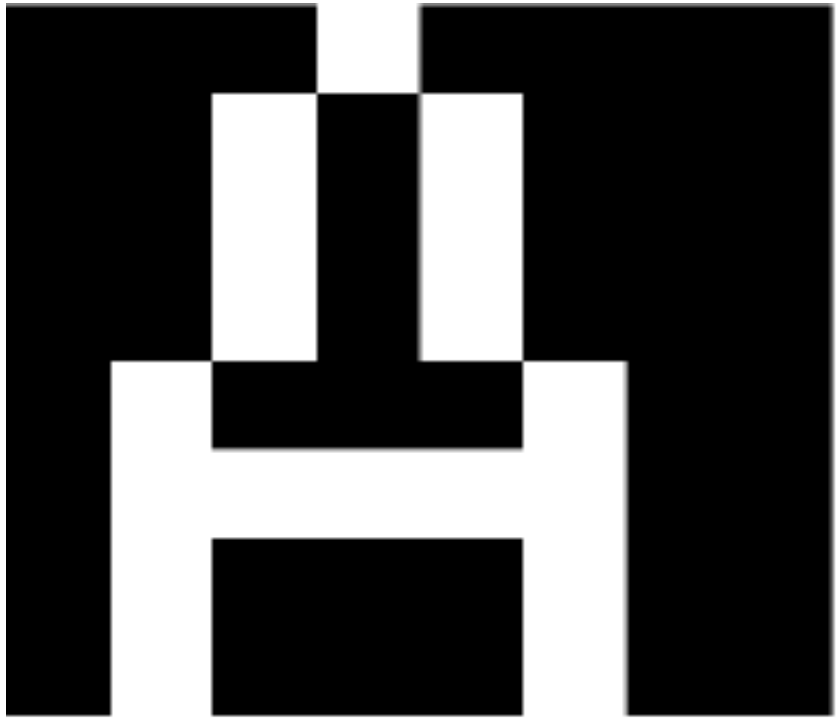
- DCT gives us the strength of each component to build a single image



$$S(u,v) = \frac{2}{\sqrt{nm}} C(u)C(v) \sum_{y=0}^{m-1} \sum_{x=0}^{n-1} s(x,y) \cos\frac{(2x+1)u\pi}{2n} \cos\frac{(2y+1)v\pi}{2m},$$

$$u = 0, \mathrm{K}, n-1; \quad v = 0, \mathrm{K}, m-1$$

Discrete Cosine Transformation



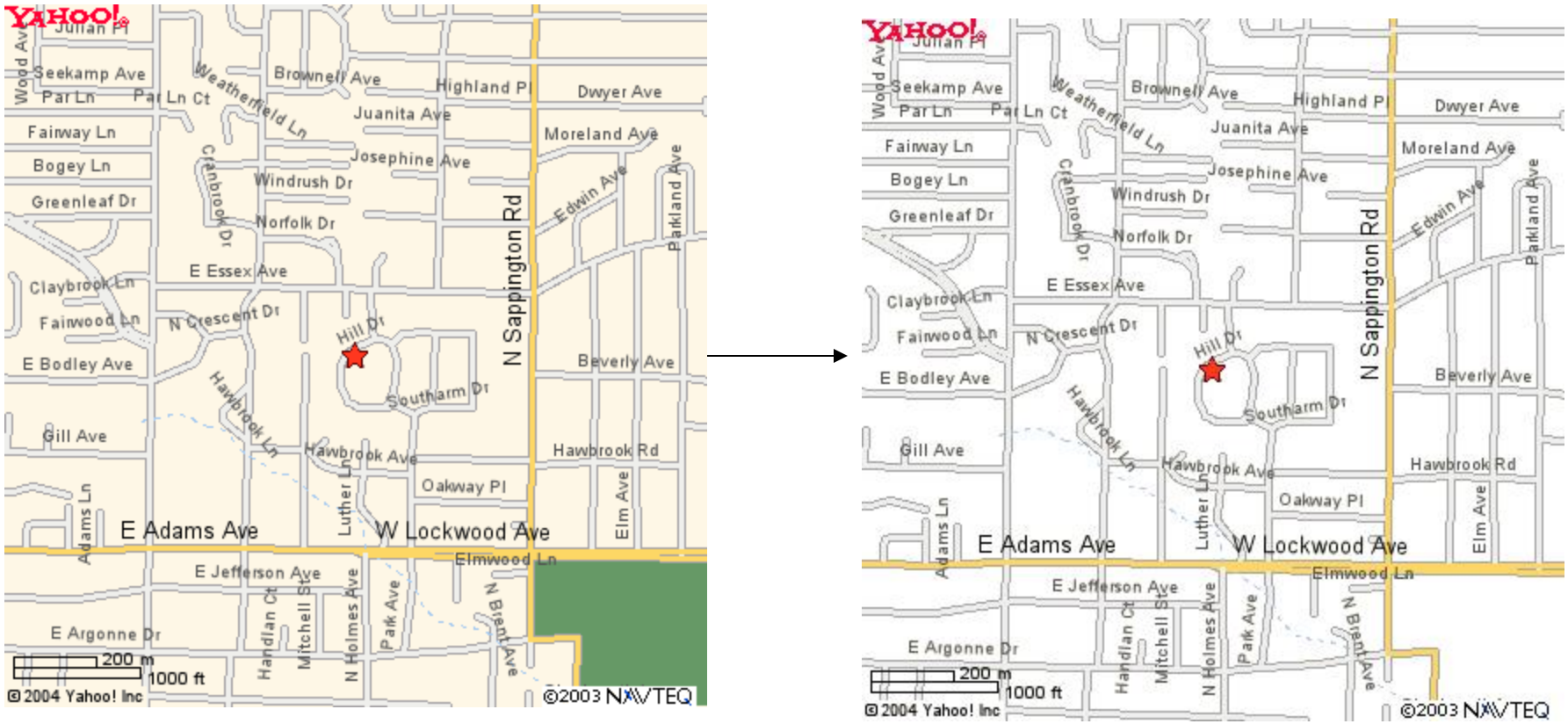
Remove background

- We apply DCT transformation for each pixel
- The DCT coefficients represent the variation around each pixel
- The pixels with low variation (near 0) around them are the background pixels

Remove background

- Now we have the color of the background pixels by DCT
- The probability of color C to be background P(B|C) and the probability of the color to be foreground P(F|C)
  - If P(B|C) > P(F|C) then color C is background color
  - Else color C is foreground color

Remove background

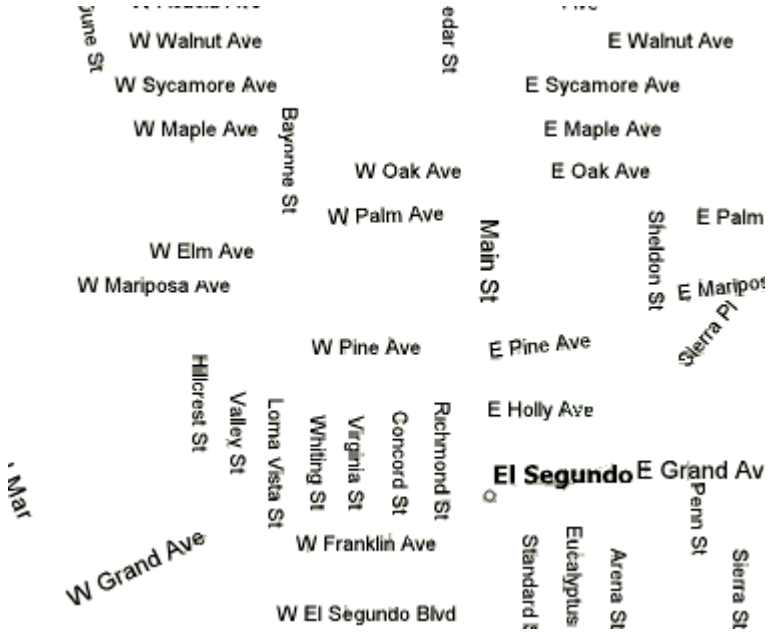


Classify Line and Character pixels

- We apply DCT transformation for each foreground pixel
- The DCT coefficients represent the variation around each foreground pixel
- We use the DCT coefficients as features for SVM to classify the pixels

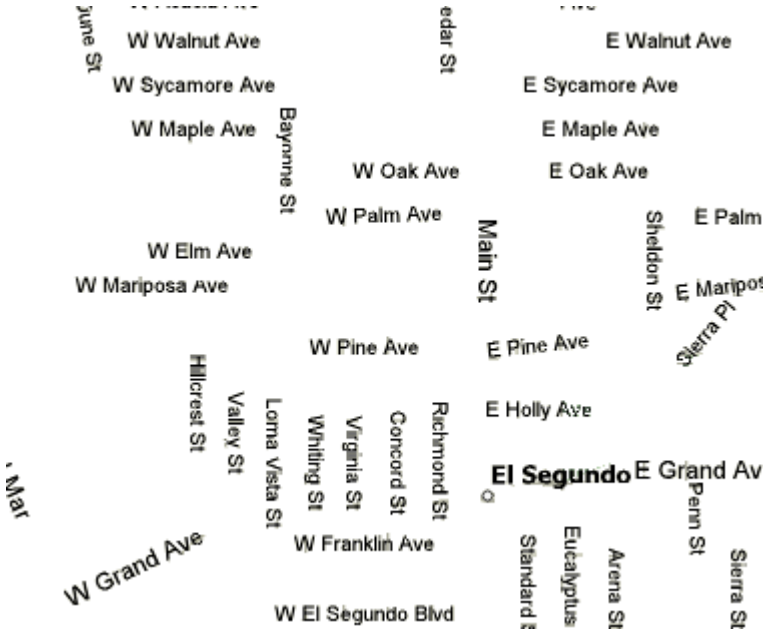
Classify Line and Character pixels

- Training
  - One MapQuest map for character samples
  - One Google map and one Viamichline map for line samples



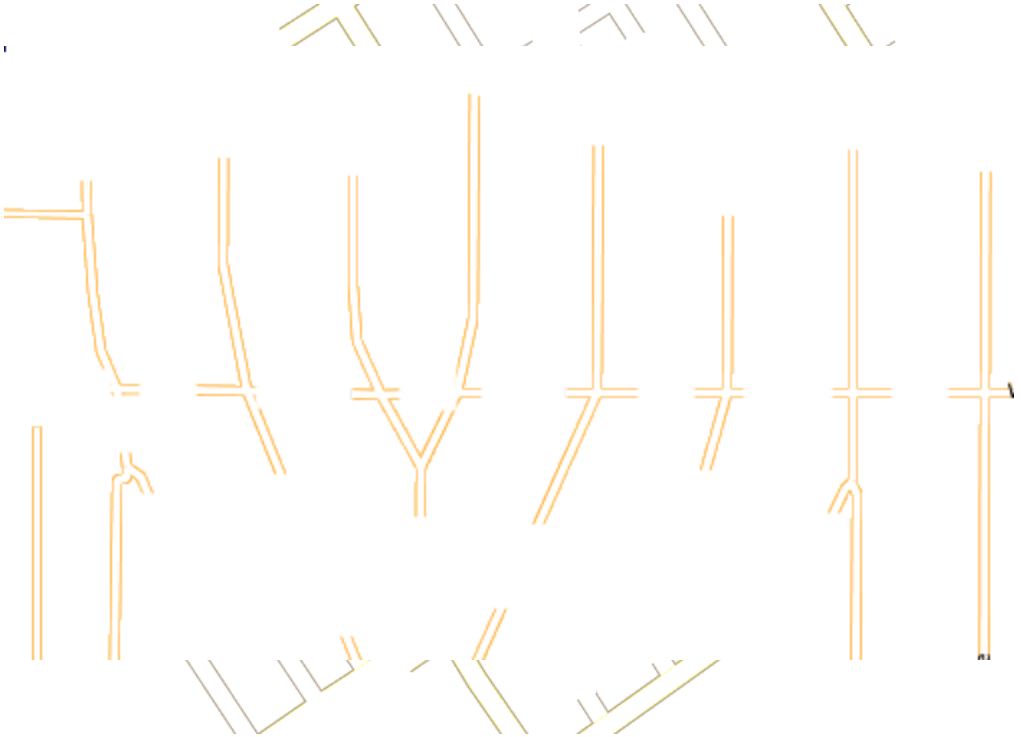
Classify Line and Character pixels

- Training
  - One MapQuest map for character samples
  - One Google map and one Viamichline map for line samples



Classify Line and Character pixels

- Training
  - One MapQuest map for character samples
  - One Google map and one Viamichline map for line samples



Classify Line and Character pixels

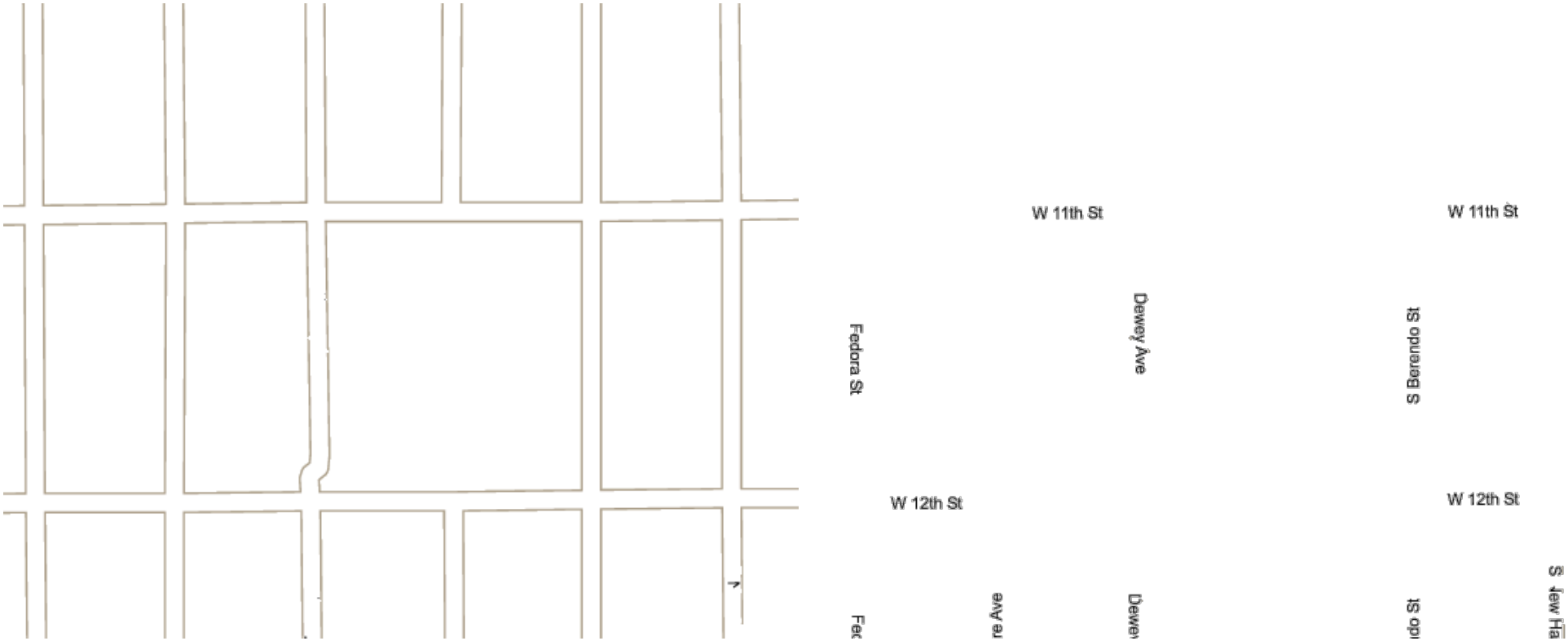
- Classification
  - The testing maps are disjoint from the training samples

Classify Line and Character pixels

- Classification
  - The testing maps are disjoint from the training samples

Classify Line and Character pixels

- Classification
  - The testing maps are disjoint from the training samples



Classify Line and Character pixels

Map Source	Precision/Recall of Classification			
	Line Pixels		Character Pixels	
	Ours	Cao's	Ours	Cao's
A9	**99/91%**	95/91%	**79/98%**	77/85%
MSN	**99/79%**	91/87%	**75/99%**	81/86%
Google	**99/99%**	95/99%	**98/99%**	95/72%
Yahoo	**95/91%**	70/96%	**91/92%**	88/30%
Mapquest	**99/78%**	88/73%	**84/98%**	76/85%
Map24	**95/74%**	97/70%	**73/96%**	70/98%
ViaMichelin	**83/34%**	44/57%	**87/96%**	90/68%
Multimap	**89/82%**	98/64%	**63/90%**	46/97%
TIGER/Line	**99/94%**	97/89%	**83/99%**	67/90%
Average	**98/85%**	85/82%	**83/96%**	71/71%

Discussion

- Computation time:
  - For a 400x400 Google Map:
    - 2 seconds to remove background
    - 4 seconds to classify line and character pixels
- No threshold needed
- Line and character pixels can be used in vectorization and OCR components