Ferns for traffic sign detection

- Thomas Marschall, Sebastian Lerch, Christoph Ihrke, Sahand Sharifzadeh
- Team Triforce

[Source: trafficsignstore.com]
Training

Trained Image  Input Image

[Source: campar.in.tum.de]

Team Triforce  Ferns for traffic sign detection
We are looking for \( \arg\max_i P(C = c_i \mid \text{patch}) \).

If \text{patch} can be represented by a set of image features \( \{f_i\} \):

\[
P(C = c_i \mid \text{patch}) = P(C = c_i \mid f_1, f_2, \ldots, f_{n}, f_{n+1}, \ldots, f_{N})
\]

which is proportional to

\[
P(f_1, f_2, \ldots, f_{n}, f_{n+1}, \ldots, f_{N} \mid C = c_i)
\]

but complete representation of the joint distribution infeasible.

Naive Bayesian ignores the correlation:

\[
\approx \prod_j P(f_j \mid C = c_i)
\]

Compromise:

\[
\approx P(f_1, f_2, \ldots, f_{n} \mid C = c_i) \times P(f_{n+1}, \ldots, f_{2n} \mid C = c_i) \times \ldots
\]

[Source:web.eecs.umich.edu/~silvio/teaching/EECS598_2010]

Team Triforce

Ferns for traffic sign detection
Training

$\textbf{I}(m_1) \leq \textbf{I}(m_2)$

$\textbf{I}(m_1) \geq \textbf{I}(m_2)$

[Source: campar.in.tum.de]
The tests compare the intensities of two pixels around the keypoint:

\[
f_i = \begin{cases} 
1 & \text{if } I(m_{i,1}) \leq I(m_{i,2}) \\
0 & \text{otherwise}
\end{cases}
\]

Invariant to light change by any raising function.

Posterior probabilities:

\[P(f_1, f_2, \cdots f_n \mid C = c_j)\]

[Source: web.eecs.umich.edu/~silvio/teaching/EECS598_2010]
Implementation: Training

- Find the robust keypoints
  - Find original keypoints
  - Warp image and find keypoints
  - Transform back the warped keypoints
  - Match the original and back warped keypoints

- Train the Ferns
  - Take a patch around each robust keypoint
  - Warp the patch
  - Extract features
Team Triforce: Ferns for traffic sign detection

[Source: web.eecs.umich.edu/~silvio/teaching/EECS598_2010]
Implementation: Classification

- Find keypoints on the test image
- Take patches and extract features
- Calculate probabilities for classes
- Extract highest and apply threshold
Altered Roadmap

Milestone 1: Training (first week)
Keypoint extraction from Training data
Training the Ferns

Actual Milestone 1:
Tried to get comparable code to work (no success there)
Implemented robust keypoint extraction
Trained first few ferns, still buggy probabilities
Altered Roadmap

Milestone 2: Classify (second week)
Keypoint extraction and classification of test data

Actual Milestone 2:
  Finished fern creation
  Added classifier functionality
Altered Roadmap

Milestone 3: Finalizing (last week)
Testing and Tuning
Finding and fixing bugs
Extending

Actual Milestone 3:
Excessive testing
Finding bugs (fixing most?)
Added generic capabilities
Trying to figure out sensible parameters

Team Triforce
Ferns for traffic sign detection
Demonstration

Team Triforce
Ferns for traffic sign detection
Thank you for your attention