Robotics 2
AdaBoost for People and Place Detection

Giorgio Grisetti, Cyrill Stachniss,
Kai Arras, Wolfram Burgard

v.1.0, Kai Arras, Oct 09, including material by Luciano Spinello and Oscar Martinez Mozos
Contents

- Motivation
- AdaBoost
- Boosting Features for People Detection
- Boosting Features for Place Labeling
Motivation: People Detection

**People tracking** is a key technology for many problems and robotics-related fields:

- Human-Robot-Interaction (HRI)
- Social Robotics: social learning, learning by imitation and observation
- Motion planning in populated environments
- Human activity and intent recognition
- Abnormal behavior detection
- Crowd behavior analysis and control

**People detection** is a key component therein.
Motivation: People Detection

- Where are the people?
Motivation: People Detection

- Where are the people?
- Why is it hard?

Range data contain little information on people

Hard in cluttered environments
Motivation: People Detection

- **Appearance of people** in range data **changes drastically** with:
  - Body pose
  - Distance to sensor
  - Partial occlusion and self-occlusion

- **2d data** from a SICK laser scanner
Motivation: People Detection

- **Appearance of people** in range data:
  **3d data** from a Velodyne sensor
Motivation: People Detection
Motivation: People Detection

Freiburg main station
Motivation: People Detection

- Freiburg Main Station data set: raw data
Motivation: People Detection

- Freiburg Main Station data set: **labeled data**
Motivation: People Detection

- Freiburg Main Station data set: labeled data
Contents

- Motivation

- AdaBoost
  - Boosting Features for People Detection
  - Boosting Features for Place Labeling
Boosting

- **Supervised Learning** technique
  user provides \( \langle \text{input data } x, \text{ label } y \rangle \)

- Learning an **accurate** classifier by combining moderately **inaccurate** “rules of thumb”

- Inaccurate rules: **weak classifiers**

- Accurate rule: **strong classifier**

- Most popular algorithm: **AdaBoost**
  [Freund et al. 95], [Schapire et al. 99]

**AdaBoost in Robotics:**

[Viola et al. 01], [Treptow et al. 04], [Martínez-Mozos et al. 05],
[Rottmann et al. 05], [Monteiro et al. 06], [Arras et al. 07]
AdaBoost

- What AdaBoost can do for you:
  1. It tells you what the best "features" are
  2. What the best thresholds are, and
  3. How to combine this to a classifier

- It's a non-linear classifier
- Robust to overfitting
- Very simple to implement
- Classifier design becomes science, not art
AdaBoost

- A machine learning ensemble technique (also called committee methods)

- There are other ensemble technique such as Bagging, Voting etc.

- Combines an ensemble of weak classifiers (weak learners) to create a strong classifier

- Prerequisite: weak classifier better than chance, that is, error < 0.5 (in a binary classification problem)
Classification

Linear vs Non-Linear Classifier
Classification

Overfitting

- Overfitting occurs when a model begins to memorize the training data rather than learning the underlying relationship.
- Occurs typically when fitting a statistical model with too many parameters.
- Overfitted models explain training data perfectly.
- But: model does not generalize!
- There are techniques to avoid overfitting such as regularization or cross-validation.
Classification

Error types

<table>
<thead>
<tr>
<th>True value</th>
<th>Predicted value of the classifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T$</td>
<td>$T'$</td>
</tr>
<tr>
<td>True Positive</td>
<td>False Positive</td>
</tr>
<tr>
<td>False Negative</td>
<td>True Negative</td>
</tr>
</tbody>
</table>

- Sensitivity (True Positive Rate) = $\frac{TP}{T}$
- Specificity (True Negative Rate) = $\frac{TN}{N}$

many more measures...
Classification

- Classification algorithms are **supervised algorithms** to predict **categorical labels**
- Differs from **regression** which is a supervised technique to predict **real-valued labels**

**Formal problem statement:**

- **Produce a function** that maps
  \[
  C : \mathcal{X} \rightarrow \mathcal{Y}
  \]
- **Given a training set**
  \[
  \{(x_1, y_1), \ldots, (x_n, y_n)\} \quad y \in \mathcal{Y} \quad \text{labels}
  \]
  \[
  x \in \mathcal{X} \quad \text{values} 
  \]
AdaBoost: Weak Classifier

Alternatives

- **Decision stump:**
  Single axis-parallel partition of space

- **Decision tree:**
  Hierarchical partition of space

- **Multi-layer perceptron:**
  General non-linear function approximators

- **Radial basis function:**
  Non-linear expansions based on kernels
AdaBoost: Weak Classifier

Decision stump

- Simple-most type of decision tree
- Equivalent to linear classifier defined by affine hyperplane
- Hyperplane is orthogonal to axis with which it intersects in threshold $\theta$
- Commonly not used on its own
- Formally,

$$h(x; j, \theta) = \begin{cases} +1 & x_j > \theta \\ -1 & \text{else} \end{cases}$$

where $x$ is training sample, $j$ is dimension
AdaBoost: Algorithm

Given the training data \(\{(x_1, y_1), \ldots, (x_n, y_n)\}\) \(x \in X \quad y \in Y\)

1. Initialize weights \(w_i = 1/n\)

2. For \(m = 1, \ldots, M\)
   - Train a weak classifier \(h_m(x)\) on weighted training data minimizing \(\sum_i w_i I(y_i \neq h_m(x_i))\)
   - Compute error of \(h_m(x)\): \(e_m = \frac{\sum_{i=1}^{n} w_i I(y_i \neq h_m(x_i))}{\sum_{i=1}^{n} w_i}\)
   - Compute voting weight of \(h_m(x)\): \(\alpha_m = \log\left(\frac{1 - e_m}{e_m}\right)\)
   - Recompute weights: \(w_i = w_i \exp\{\alpha_m \cdot I(y_i \neq h_m(x_i))\}\)

3. Make predictions using the final strong classifier
AdaBoost: Strong Classifier

- Computing the **voting weight** $\alpha_m$ of a weak classifier

\[
\alpha_m = \log \left( \frac{1 - e_m}{e_m} \right)
\]

error

chance: $e_m = 0.5$
AdaBoost: Strong Classifer

- Training is completed...
  The weak classifiers \( h_{1...M}(x) \) and their voting weight \( \alpha_{1...M} \) are now fix

- The resulting strong classifier is

\[
H(x_i) = \text{sgn}\left( \sum_{m=1}^{M} \alpha_m h_m(x_i) \right) \quad \text{Class Result \{+1, -1\}}
\]

Put your data here

Weighted majority voting voting scheme
AdaBoost: Weak Classifier

- **Train a decision stump on weighted data**

\[
(j^*, \theta^*) = \arg\min_{j, \theta} \frac{\sum_{i=1}^{n} w_i I(y_i \neq h_m(x_i))}{\sum_{i=1}^{n} w_i}
\]

- **This consists in...**

Finding an optimum parameter \( \theta^* \) for each dimension \( j = 1 \ldots d \) and then select the \( j^* \) for which the cost is minimal.
AdaBoost: Weak Classifier

A simple training algorithm for stumps:

\[
\forall \ j = 1...d
\]

Sort samples \( x_i \) in ascending order along dimension \( j \)

\[
\forall \ i = 1...n
\]

Compute \( n \) cumulative sums \( w_{\text{cum}}^j(i) = \sum_{k=1}^{i} w_k y_k \)

end

Threshold \( \theta_j \) is at extremum of \( w_{\text{cum}}^j \)

Sign of extremum gives direction \( p_j \) of inequality

end

Global extremum in all \( d \) sums \( w_{\text{cum}} \) gives

**threshold** \( \theta^* \) and **dimension** \( j^* \)
AdaBoost: Weak Classifier

Training algorithm for stumps: Intuition

- **Label** $y$:
  - red: +
  - blue: –

- Assuming all weights = 1

\[
W_{\text{cum}}^{j}(i) = \sum_{k=1}^{i} w_k y_k
\]
**AdaBoost: Algorithm**

Given the **training data** \( \{(x_1, y_1), \ldots, (x_n, y_n)\} \quad x \in \mathcal{X} \quad y \in \mathcal{Y} \)

1. Initialize weights \( w_i = 1/n \)

2. For \( m = 1, \ldots, M \)
   
   - Train a **weak classifier** \( h_m(x) \) on weighted training data minimizing \( \sum_i w_i I(y_i \neq h_m(x_i)) \)
   
   - Compute error of \( h_m(x) \): \( e_m = \frac{\sum_{i=1}^{n} w_i I(y_i \neq h_m(x_i))}{\sum_{i=1}^{n} w_i} \)
   
   - Compute voting weight of \( h_m(x) \): \( \alpha_m = \log\left(\frac{1 - e_m}{e_m}\right) \)
   
   - Recompute weights: \( w_i = w_i \exp\{\alpha_m \cdot I(y_i \neq h_m(x_i))\} \)

3. Make predictions using the final **strong classifier**
AdaBoost: Step-By-Step

- Training data
AdaBoost: Step-By-Step

- **Iteration 1, train weak classifier 1**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Threshold</td>
<td>$\theta^* = 0.37$</td>
</tr>
<tr>
<td>Dimension</td>
<td>$j^* = 1$</td>
</tr>
<tr>
<td>Weighted error</td>
<td>$e_m = 0.2$</td>
</tr>
<tr>
<td>Voting weight</td>
<td>$\alpha_m = 1.39$</td>
</tr>
<tr>
<td>Total error</td>
<td>$= 4$</td>
</tr>
</tbody>
</table>
AdaBoost: Step-By-Step

- **Iteration 1, recompute weights**

Threshold
\[ \theta^* = 0.37 \]

Dimension
\[ j^* = 1 \]

Weighted error
\[ e_m = 0.2 \]

Voting weight
\[ \alpha_m = 1.39 \]

Total error = 4
AdaBoost: Step-By-Step

- **Iteration 2, train weak classifier 2**

  - **Threshold**
    \[ \theta^* = 0.47 \]

  - **Dimension**
    \[ j^* = 2 \]

  - **Weighted error**
    \[ e_m = 0.16 \]

  - **Voting weight**
    \[ \alpha_m = 1.69 \]

  - **Total error** = 5
AdaBoost: Step-By-Step

- **Iteration 2, recompute weights**

\[
\begin{align*}
\text{Threshold} & \quad \theta^* = 0.47 \\
\text{Dimension} & \quad j^* = 2 \\
\text{Weighted error} & \quad e_m = 0.16 \\
\text{Voting weight} & \quad \alpha_m = 1.69 \\
\text{Total error} & = 5
\end{align*}
\]
AdaBoost: Step-By-Step

- Iteration 3, train weak classifier 3

Threshold
\( \theta^* = 0.14 \)

Dimension, sign
\( j^* = 2, \text{ neg} \)

Weighted error
\( e_m = 0.25 \)

Voting weight
\( \alpha_m = 1.11 \)

Total error = 1
AdaBoost: Step-By-Step

- **Iteration 3, recompute weights**

  - **Threshold**
    \[ \theta^* = 0.14 \]
  - **Dimension, sign**
    \[ j^* = 2 \text{, neg} \]
  - **Weighted error**
    \[ e_m = 0.25 \]
  - **Voting weight**
    \[ \alpha_m = 1.11 \]
  - **Total error** = 1
AdaBoost: Step-By-Step

- Iteration 4, train weak classifier 4

Threshold
\( \theta^* = 0.37 \)

Dimension
\( j^* = 1 \)

Weighted error
\( e_m = 0.20 \)

Voting weight
\( \alpha_m = 1.40 \)

Total error = 1
AdaBoost: Step-By-Step

- Iteration 4, recompute weights

Threshold
\( \theta^* = 0.37 \)

Dimension
\( j^* = 1 \)

Weighted error
\( e_m = 0.20 \)

Voting weight
\( \alpha_m = 1.40 \)

Total error = 1
AdaBoost: Step-By-Step

- Iteration 5, train weak classifier 5

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Threshold ( \theta^* )</td>
<td>0.81</td>
</tr>
<tr>
<td>Dimension ( j^* )</td>
<td>1</td>
</tr>
<tr>
<td>Weighted error ( e_m )</td>
<td>0.28</td>
</tr>
<tr>
<td>Voting weight ( \alpha_m )</td>
<td>0.96</td>
</tr>
</tbody>
</table>

Total error = 1
AdaBoost: Step-By-Step

- **Iteration 5, recompute weights**

  - Threshold $\theta^* = 0.81$
  - Dimension $j^* = 1$
  - Weighted error $e_m = 0.28$
  - Voting weight $\alpha_m = 0.96$
  - Total error = 1
AdaBoost: Step-By-Step

- **Iteration 6, train weak classifier 6**

  - **Threshold** \( \theta^* = 0.47 \)
  - **Dimension** \( j^* = 2 \)
  - **Weighted error** \( e_m = 0.29 \)
  - **Voting weight** \( \alpha_m = 0.88 \)
  - **Total error** = 1
### AdaBoost: Step-By-Step

- **Iteration 6, recompute weights**

  - **Threshold**
    \[ \theta^* = 0.47 \]

  - **Dimension**
    \[ j^* = 2 \]

  - **Weighted error**
    \[ e_m = 0.29 \]

  - **Voting weight**
    \[ \alpha_m = 0.88 \]

  - **Total error** = 1
AdaBoost: Step-By-Step

- Iteration 7, train weak classifier 7

Threshold
\[ \theta^* = 0.14 \]

Dimension, sign
\[ j^* = 2, \text{ neg} \]

Weighted error
\[ e_m = 0.29 \]

Voting weight
\[ \alpha_m = 0.88 \]

Total error = 1
AdaBoost: Step-By-Step

- **Iteration 7, recompute weights**

Threshold
\[ \theta^* = 0.14 \]

Dimension, sign
\[ j^* = 2, \text{ neg} \]

Weighted error
\[ e_m = 0.29 \]

Voting weight
\[ \alpha_m = 0.88 \]

Total error = 1
AdaBoost: Step-By-Step

- Iteration 8, train weak classifier 8

Threshold
\( \theta^* = 0.93 \)

Dimension, sign
\( j^* = 1, \text{ neg} \)

Weighted error
\( e_m = 0.25 \)

Voting weight
\( \alpha_m = 1.12 \)

Total error = 0
AdaBoost: Step-By-Step

- **Iteration 8, recompute weights**

  - **Threshold** \( \theta^* = 0.93 \)
  - **Dimension, sign** \( j^* = 1, \text{ neg} \)
  - **Weighted error** \( e_m = 0.25 \)
  - **Voting weight** \( \alpha_m = 1.12 \)
  - **Total error** = 0
AdaBoost: Step-By-Step

- Final Strong Classifier

Total error = 0!

(rarely met in practice)

Colored back-ground dots in figure is test set.
AdaBoost: Wrap Up

- **Misclassified** samples receive higher weight. The higher the weight the "more attention" a training sample gets.

- Algorithm generates weak classifier by training the next learner **on the mistakes** of the previous one.

- Now we also understand the name: **adaptive Boosting → AdaBoost**

- Once training is done, AdaBoost is a **voting method**.

- **Large impact** on ML community and beyond.
Contents

- Motivation
- AdaBoost
- **Boosting Features for People Detection**
- Boosting Features for Place Labeling
Problem and Approach

- Can we find **robust features** for people, legs and groups of people in 2D range data?
- What are the **best features** for people detection?
- Can we find people that **do not move**?

**Approach:**
- Classifying **groups of adjacent** beams
- Computing a set of simple (scalar) features on these groups
- **Boosting** the features
Related Work

- **People Tracking**
  - [Fod et al. 2002]
  - [Kleinhagenbrock et al. 2002]
  - [Schulz et al. 2003]
  - [Scheutz et al. 2004]
  - [Topp et al. 2005]
  - [Cui et al. 2005]
  - [Schulz 2006]
  - [Mucientes et al. 2006]

- SLAM in dyn. env.
  - [Montemerlo et al. 2002]
  - [Hähnel et al. 2003]
  - [Wang et al. 2003]
  - ...

- People detection done with very simple classifiers:
  - **manual** feature selection, **heuristic** thresholds

- **Typically:** narrow local-minima blobs that move
Segmentation

- Divide the scan into segments

"Range image segmentation"
Segmentation

- **Method:** Jump distance condition
  See [Premebida et al. 2005] for survey

- **Size filter:**
  rejection of too small segments
Segmentation

- **Method:** Jump distance condition
  See [Premebida et al. 2005] for survey

- **Size filter:**
  rejection of too small segments
**Segmentation**

- **Method:** Jump distance condition
  -See [*Premebida et al. 2005*] for survey

- **Size filter:** rejection of too small segments
Features

Segment $S_i$

1. Number of points $n = |S_i|$
2. Standard Deviation $\sigma = \sqrt{\frac{1}{n-1} \sum ||x_j - \bar{x}||^2}$
3. Mean avg. deviation from median $\mu = \frac{1}{n} \sum ||x_j - \tilde{x}||$
4. Jump dist. to preceding segment $\delta_{j-1,j}$
5. Jump dist. to succeeding segment $\delta_{j,j+1}$
6. Width $w_i = ||x_1 - x_n||$
Features

Segment $S_i$

7. Linearity $s_l = \sum (x_j \cos(\alpha) + y_j \sin(\alpha) - r)^2$

8. Circularity $s_c = \sum (r_c - \sqrt{(x_c - x_i)^2 + (y_c - y_i)^2})^2$

9. Radius $r_c$
Features

Segment $S_i$

10. Boundary Length $\quad l = \sum_j d_{j,j-1}$

11. Boundary Regularity $\quad \sigma_d = \sqrt{\frac{1}{n-1} \sum (d_{j,j-1} - \bar{d})^2}$

12. Mean curvature $\quad \bar{k} = \frac{1}{n} \sum \hat{k}_j$

13. Mean angular difference $\quad \bar{\beta} = \frac{1}{n} \sum \beta_j$

14. Mean speed $\quad \bar{v} = \frac{1}{n} \sum \frac{\rho_{j}^{k+1} - \rho_{j}^{k}}{\Delta T}$
Features

- Resulting **feature signature** for each segment
Training: Data Labeling

- **Mark segments** that correspond to people
- Either **manually** or **automatically**
Training: Data Labeling

- **Automatic labeling**: obvious approach, define area of interest

  Here: discrimination from clutter is interesting. Features include spatial relation between fore- and background. Thus: labeling is done **by hand**
Training

- Resulting **Training Set**

- Segments corresponding to people (foreground segments)

- Segments corresponding to other objects (background segments)
AdaBoost: Weak Classifiers

- Each binary weak classifier $h_j(x)$ is created using a single-valued feature $f_j$ in the form:

$$h_j(x) = \begin{cases} 
1 & \text{if } p_j f_j(x) > p_j \theta_j \\
-1 & \text{otherwise,}
\end{cases}$$

- Where $\theta_j$ is a threshold and $p_j \in \{-1, 1\}$ indicates the direction of the inequality

- Weak classifier must be better than random
**AdaBoost: Final Strong Classifer**

Vocabulary of features

- \( f_{#1} \)
- \( \ldots \)
- \( f_{#14} \)

Boosting

- \( example_1 \)
- \( \ldots \)
- \( example_N \)

Weighted majority vote classifier

- \( w_1 h_1 \)
- \( \ldots \)
- \( w_T h_T \)

\[ h_s(x) = \sum_{t=1}^{T} w_t h_t(x) \]
Experiments

**Env. 1**: Corridor, no clutter

<table>
<thead>
<tr>
<th>True Label</th>
<th>Detected Label</th>
<th>Person</th>
<th>No Person</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Person</td>
<td>239 (99.58%)</td>
<td></td>
<td>1 (0.42%)</td>
<td>240</td>
</tr>
<tr>
<td>No Person</td>
<td>27 (1.03%)</td>
<td></td>
<td>2589 (98.97%)</td>
<td>2616</td>
</tr>
</tbody>
</table>

**Env. 2**: Office, very cluttered

<table>
<thead>
<tr>
<th>True Label</th>
<th>Detected Label</th>
<th>Person</th>
<th>No Person</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Person</td>
<td>497 (97.45%)</td>
<td></td>
<td>13 (2.55%)</td>
<td>510</td>
</tr>
<tr>
<td>No Person</td>
<td>171 (2.73%)</td>
<td></td>
<td>6073 (96.26%)</td>
<td>6244</td>
</tr>
</tbody>
</table>
Experiments

Env. 1 & 2: Corridor and Office

<table>
<thead>
<tr>
<th>Detected Label</th>
<th>Person</th>
<th>No Person</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>True Label</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Person</td>
<td>722 (96.27%)</td>
<td>28 (3.73%)</td>
<td>750</td>
</tr>
<tr>
<td>No Person</td>
<td>225 (2.54%)</td>
<td>8649 (99.88%)</td>
<td>8860</td>
</tr>
</tbody>
</table>

Env. 1→2: Cross-evaluation
Trained in corridor, applied in office

<table>
<thead>
<tr>
<th>Detected Label</th>
<th>Person</th>
<th>No Person</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>True Label</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Person</td>
<td>217 (90.42%)</td>
<td>23 (9.58%)</td>
<td>240</td>
</tr>
<tr>
<td>No Person</td>
<td>112 (4.28%)</td>
<td>2504 (95.72%)</td>
<td>2616</td>
</tr>
</tbody>
</table>
Experiments

Adding motion feature (mean speed, f#14)

<table>
<thead>
<tr>
<th></th>
<th>Without Motion Feature</th>
<th>With Motion Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>False Negatives (%)</td>
<td>3.73</td>
<td>3.47</td>
</tr>
<tr>
<td>False Positives (%)</td>
<td>2.54</td>
<td>3.13</td>
</tr>
<tr>
<td>Total Error (%)</td>
<td>2.63</td>
<td>3.15</td>
</tr>
</tbody>
</table>

→ Motion feature has no contribution

Experimental setup:
- Robot Herbert
- SICK LMS 200 laser range finder, 1 deg angular resolution
Experiments

- Comparison with **hand-tuned classifier**
  - **Jump distance** \( \theta_\delta = 30 \text{ cm} \)
  - **Width** \( \theta_{w,m} = 5 \text{ cm}, \ \theta_{w,M} = 50 \text{ cm} \)
  - **Number of points** \( \theta_n = 4 \)
  - **Standard deviation** \( \theta_\sigma = 50 \text{ cm} \)
  - **Motion of points** \( \theta_v = 2 \text{ cm} \)

<table>
<thead>
<tr>
<th></th>
<th>Heuristic Approach</th>
<th>AdaBoost</th>
</tr>
</thead>
<tbody>
<tr>
<td>False Negatives (%)</td>
<td>34.67</td>
<td>3.73</td>
</tr>
<tr>
<td>False Positives (%)</td>
<td>9.06</td>
<td>2.54</td>
</tr>
<tr>
<td>Overall Error (%)</td>
<td><strong>11.06</strong></td>
<td><strong>2.63</strong></td>
</tr>
</tbody>
</table>

People are often not detected
Experiments

Five **best features:**

1: **Radius** $r_c$
   of LSQ-fitted circle, robust size measure (#9)

2: **Mean angular difference**
   Convexity measure (#13)

3/4: **Jump distances**
   Local minima measure (#4 and #5)

5: **Mad from median**
   Robust compactness measure (#3)

<table>
<thead>
<tr>
<th>Environment</th>
<th>Five Best Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Corridor</td>
<td>9, 4, 5, 2, 4</td>
</tr>
<tr>
<td>Office</td>
<td>9, 13, 3, 4, 5</td>
</tr>
<tr>
<td>Both</td>
<td>9, 13, 4, 3, 5</td>
</tr>
</tbody>
</table>
Result: Classification

<table>
<thead>
<tr>
<th>T</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>🔵</td>
</tr>
<tr>
<td>F</td>
<td>🔴</td>
</tr>
</tbody>
</table>
Summary

- People detection phrased as a classification problem of groups of neighboring beams
- **AdaBoost** allows for a systematic approach to perform this task
- **One-shot/single-frame** people detection with over 90% accuracy
- Learned classifier clear superior to hand-tuned classifier
- **No background knowledge** such as an a priori map is needed (e.g. to perform background substraction)
Contents

- Motivation
- AdaBoost
- Boosting Features for People Detection
- Boosting Features for Place Labeling
Place Labeling: Motivation

- A map is a **metric** and **topological** model of the environment
Place Labeling: Motivation

- Wanted: **semantic** information about places
Scenario Example

Albert, where are you?

I am in the corridor
Scenario Example 2

- Semantic mapping

- Human-Robot Interaction of type: "Robot, get out of my room, go into the corridor!"
Problem Statement

- **Classification of the position** of the robot using a single observation: a 360° laser range scan
Observations
Observations

Room
Observations

Room
Observations
Observations

Room

Doorway
Observations

Corridor

Room

Doorway
Similar Observations
Similar Observations

Corridor

Doorway
Classification Problem
Classification Problem

- Corridor
- Room
- Doorway
Representing the Observations

- How we represent the 360 laser beams for our classification task?

- As a list of beams \( z = \{b_1, b_2, \ldots, b_M\} \)

**Problem:** which beam is the first beam?

Not **invariant to rotation**!
Representing the Observations

- A list of **scalar geometrical features** of the scan

\[ x = \{b_1, b_2, \ldots, b_M\} \]

\[ \downarrow \]

\[ z = \{f_1, f_2, \ldots, f_N\} \quad f_i : f(x) \rightarrow \mathbb{R} \]

The features are all **invariant to rotation**
Simple Features

- \( f = \frac{1}{N} \sum d_i \)
- Gap = \( d > \theta \)
- \( f = \# \text{ Gaps} \)
- \( f = d \)
- \( f = \text{Area} \)
- \( f = \text{Perimeter} \)
- \( f = d \)
Simple Features

- Features of the raw beams

1) The average difference between the length of consecutive beams.
2) The standard deviation of the difference between the length of consecutive beams.
3) Same as 1), but considering different max-range values.
4) The average beam length.
5) The standard deviation of the length of the beams.
6) Number of gaps in the scan. Two consecutive beams build a gap if their difference is greater than a given threshold. Different features are used for different threshold values.
7) Number of beams lying on lines that are extracted from the range scan [16].
8) Euclidean distance between the two points corresponding to the two smallest local minima.
9) The angular distance between the beams corresponding to the local minima in feature 8).
Simple Features

- Features of the **closed polynom** $P(z)$ made up by the beams

1. Area of $P(z)$.
2. Perimeter of $P(z)$.
3. Area of $P(z)$ divided by Perimeter of $P(z)$.
4. Mean distance between the centroid to the shape boundary.
5. Standard deviation of the distances between the centroid to the shape boundary.
6. 200 similarity invariant descriptors based in the Fourier transformation.
7. Major axis $Ma$ of the ellipse that approximates $P(z)$ using the first two Fourier coefficients.
8. Minor axis $Mi$ of the ellipse that approximate $P(z)$ using the first two Fourier coefficients.
9. $Ma/Mi$.
10. Seven invariants calculated from the central moments of $P(z)$.
11. Normalized feature of compactness of $P(z)$.
12. Normalized feature of eccentricity of $P(z)$.
13. Form factor of $P(z)$. 
Multiple Classes

\[ z = \{ f_1, f_2, \ldots, f_N \} \]
Multiple Classes

\[ z = \{ f_1, f_2, \ldots, f_N \} \]

Classifier: \( A(z) \rightarrow \{1, 2, 3\} \)

- Corridor: 1
- Room: 2
- Doorway: 3
Multiple Classes

- **Sequence of binary classifiers** in a decision list

  - **Order matters** as accuracy differs
  - Order according to **error rate**
  - Generalizes to sequential AdaBoost for **K classes**
Experiments

Training (top)
# examples: 16045

Test (bottom)
# examples: 18726
classification: 93.94%

Building 079
Uni. Freiburg

Corridor Room Doorway
Experiments

Training (left)
# examples: 13906

Test (right)
# examples: 10445
classification: 89.52%

Building 101
Uni. Freiburg
Application to New Environment

Intel Research Lab in Seattle
Application to New Environment

Training map

Intel Research Lab in Seattle

Corridor  Room  Doorway
Training

- Learn a **strong classifier** from a set of previously labeled observations.