

Introduction

Object Detection

Problem Description

The Adaboost Cascade

Status of the work

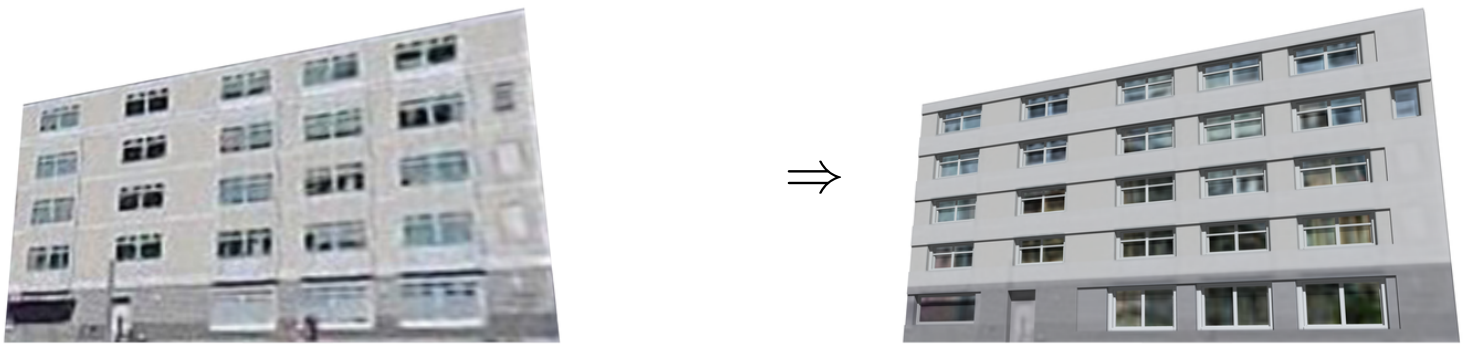
Towards Facade Interpretation

February 12, 2009

Goal

Starting Point: Images of several hundred kilometers of streets. Data is provided by GeoAutomation (www.geoautomation.be).

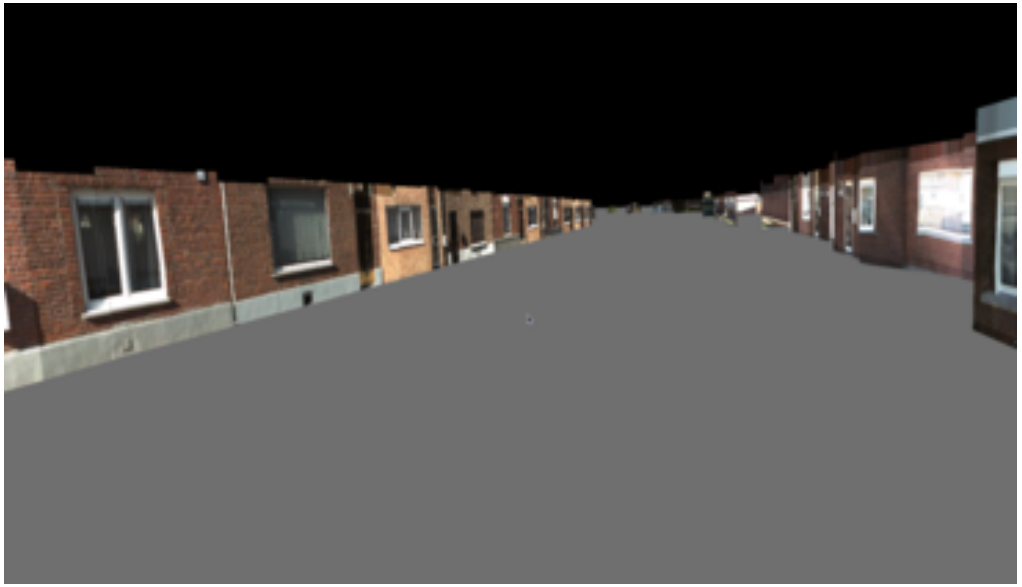
Goal: Build a virtual Model of the city, based on the procedural software CityEngine (www.procedural.com).



Figures from *Müller et al.,SIGGRAPH2007*

Workflow

- Fitting of the images to planes.
- Detection of facade Elements.
- Inpainting to correct errors in the texture.
- Use of the semantic information to derive a procedural model.

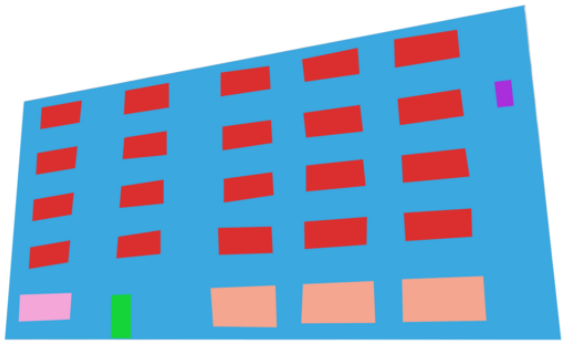


Workflow

- Fitting of the images to planes.
- Detection of facade Elements.
- Inpainting to correct errors in the texture.
- Use of the semantic information to derive a procedural model.

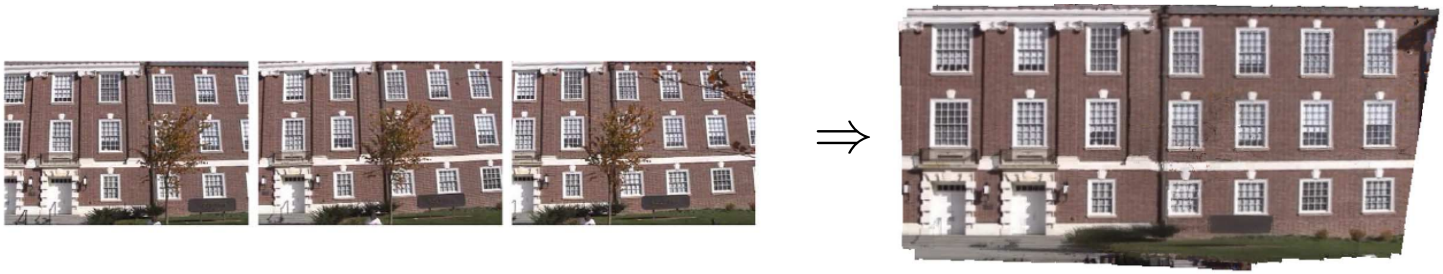


$\Rightarrow$



Workflow

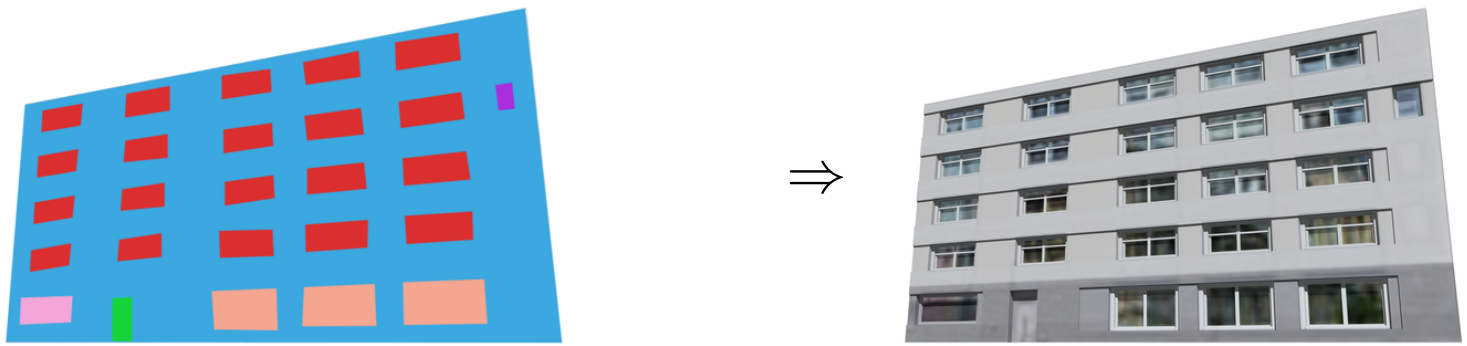
- Fitting of the images to planes.
- Detection of facade Elements.
- Inpainting to correct errors in the texture.
- Use of the semantic information to derive a procedural model.



Figures from *Korah/Rasmussen,2007*

Workflow

- Fitting of the images to planes.
- Detection of facade Elements.
- Inpainting to correct errors in the texture.
- Use of the semantic information to derive a procedural model.



Figures from *Müller et al.,SIGGRAPH2007*



Problem definition

Elements to Detect: Windows, Doors, Garages, (Steps)

Problems to detect elements of these classes:

- Variation **inside** of a class
- Not much variation **between** the classes
- Very different appearances of the glass part of the objects
- Many occlusions caused by trees, balconies, air conditioners...

Variation **inside** of the “windows“ class

Towards Facade Interpretation

Introduction

Object Detection

Problem Description

The Adaboost Cascade

Status of the work



Variation **inside** of the “door“ class

Towards Facade Interpretation

Introduction

Object Detection

Problem Description

The Adaboost Cascade

Status of the work



Variation because of different angles/illumination





How to detect?

- Features are not easy to find manually.
- If features are found, how should they be weighted?
- Standard approach based on lines doesn't work well.

$\Rightarrow$ **Use of a Cascaded Adaboost Detector, based on Haar-like Features.**

Advantages

Main advantages of such an Detector:

- Features are chosen and weighted automatically.
- Fast operation because of cascading and the use of Haar-like features.
- Publication of window detector using Adaboost shows first results but improvement necessary [Ali2007].

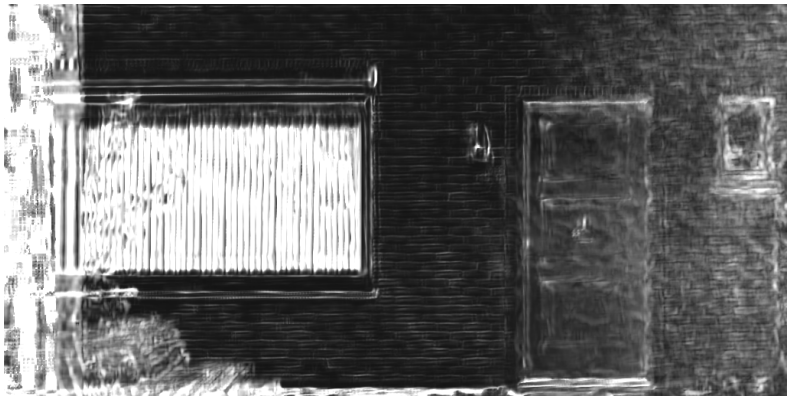
Extensions to get better results:

- The use of rectified images reduces the variance inside the classes.
- Using several detectors for subclasses inside the classes.
- Plane fitting gives measure of confidence for the fitted plane. This confidence is usually lower on glass.
  $\Rightarrow$ The 3 color channels of the image are extended by this confidence map.

[Ali2007] Ali et al.,ICIAP2007

Confidence Map

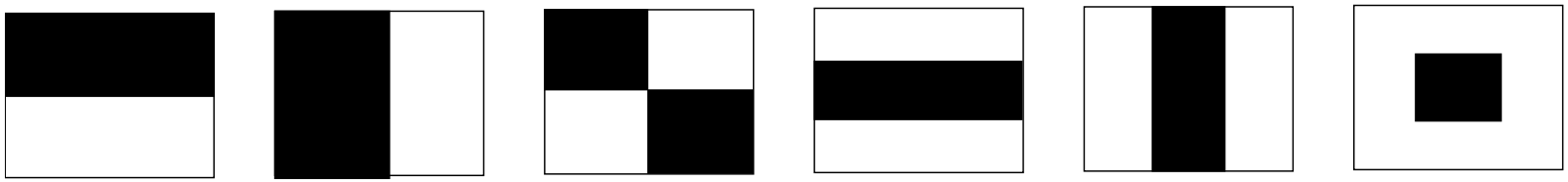




Haar-like Features

The Adaboost algorithm can be used with all kinds of features, but its often used with Haar-like Features.

- Based on integral-image there only a few (and fixed) number of calculations.
- The Number of possible Features is very large. For a 32x32 pixel image with 3 channels, the actual implementation calculates around 1,2 Million features.
- For scale invariant detection, the features can be scaled, instead of the image.



Basic Adaboost Algorithm

Given a set of training samples $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in X$, $y_i \in Y = \{-1, +1\}$

Search for a classifier: $H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$

Steps of the algorithm: for every iteration

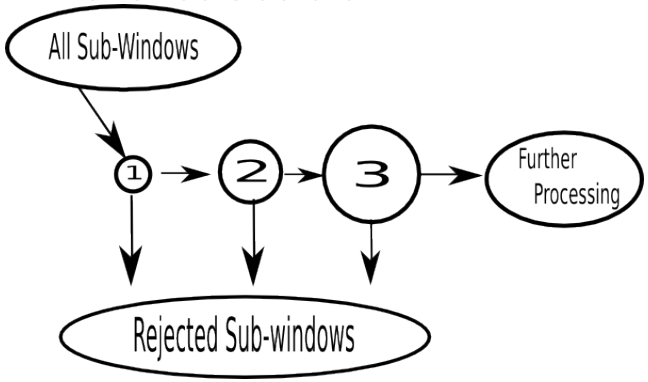
- Find the Haar-like feature that separates the data the best.
- Re-weight the training data, that samples that are wrongly classified by using this Haar-like feature get more weight, the others less. $\Rightarrow$ Concentration on "hard" samples.
- α_t is calculated based on the training error, it is a measure for the importance of the chosen Haar-like feature. The smaller the error the higher is α_t.

Freund and Schapire proved, that the training error drops exponentially fast, if every feature based decision is better than random guessing.

The cascade

So far, for every sample, that is given to the classifier, all features have to be calculated.
$\Rightarrow$ Start with small classifiers (not many features) that are **not** tuned for the lowest total classification error, but for a low false negative rate.

Goal: Drop out as many negative samples as possible and keep all positive ones. Only the "good" samples go through the whole cascade.



Status of the work

Where we are:

- Multi-threaded C++ implementation of cascaded multi-class Adaboost, tested on traffic sign dataset.
- Adjustment of the "VOC Pascal Challenge" Annotation tool and definition of the annotation rules.
- Job advertisement to hire jobstudents.

Ongoing work:

- Building of library of annotated Images (Jobstudents), maybe with collaboration with the university of Bonn.
- Extension of the plane fitting to the Montreal Dataset.
- Separation of Splitlines between Buildings.



Towards Facade Interpretation

Introduction

Object Detection

Problem Description

The Adaboost Cascade

Status of the work

Thank you!