

Introduction
The Detection System
The Adaboost Detector
Configuration
Optimization
Results

Window detection in Facades

December 7, 2009

Overview

Window detection in Facades

Introduction

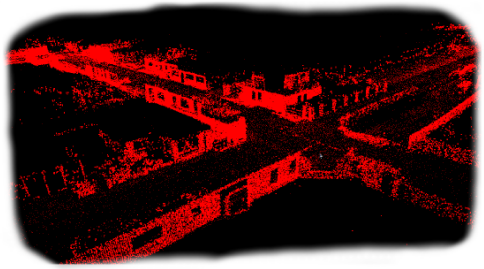
The Detection System

The Adaboost Detector

Configuration

Optimization

Results



Pointcloud



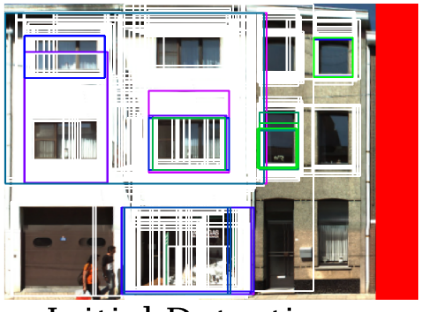
Textured Model

Virtual Camera

Feedback



Rectified Image



Initial Detections



Optimized Detections

Overview

Window detection in Facades

Introduction

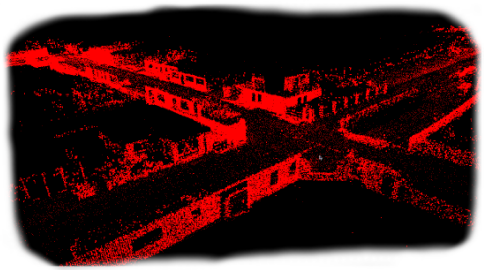
The Detection System

The Adaboost Detector

Configuration

Optimization

Results



Pointcloud



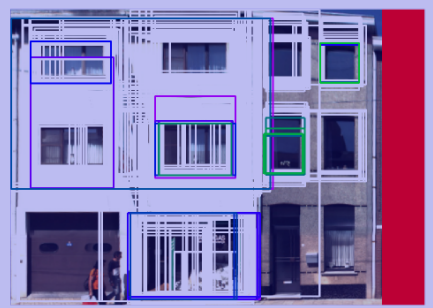
Textured Model

Virtual Camera

Feedback



Rectified Image



Initial Detections



Optimized Detections

Basic Adaboost: Idea

- **General:** given a task that requires expert knowledge to perform. k experts may be better than one if their individual judgments are appropriately combined.
- **For Adaboost:** experts are weak learners, their weighted linear combination is called a strong learner, the final classifier
- **Task:** Find a highly accurate classifier by combining many simple and moderately accurate weak learners. The weak learners are selected sequentially. Each new weak learner is found on training samples that were most difficult to classify by preceding rules.

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 a feature that separates the data the best.
- Re-weight the training data: samples that are wrongly classified by using the selected 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.

Multiclass Adaboost

- Facades usually contain many different types/classes of windows.
- Variation of samples of each class
- Similatity of samples of different classes

$\Rightarrow$ Suitable detector should handle both, different classes of windows and also alow some overlap between these classes.
$\Rightarrow$ Schapire and Singer's Adaboost.MR [] adresses exactly this multi-class multi-label problem, where multiple classes are handled but do not have to be mutually exclusive.

Multiclass Adaboost

For Training:

- k-class problem is transfered into a two-class problem k times as large.
- each input sample x from class l : (x, l) is derived into k samples:

$$((x, 1), 0), \ldots, ((x, l), 1), \ldots, ((x, k), 0)$$

For Classification:

- magnitude of the prediction $H|(x, l)|$ is interpreted as "confidence".
- the Classifier produces a ranking for possible labels.

Cascades of Adaboost Detectors

Window detection in Facades

Introduction

The Detection System

The Adaboost Detector

Configuration

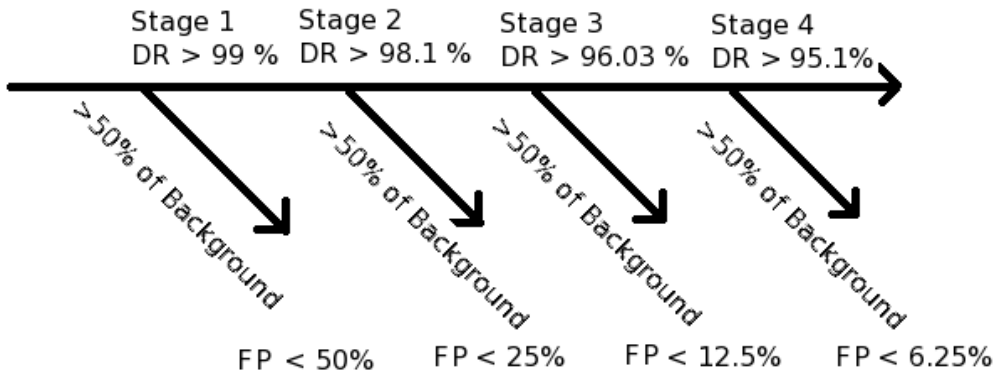
Optimization

Results

For every sample to be classified, all rules/features have to be evaluated.

⇒ Start with small classifiers (not many features) that are **not** tuned for the lowest total classification error, but for a low false negative rate.

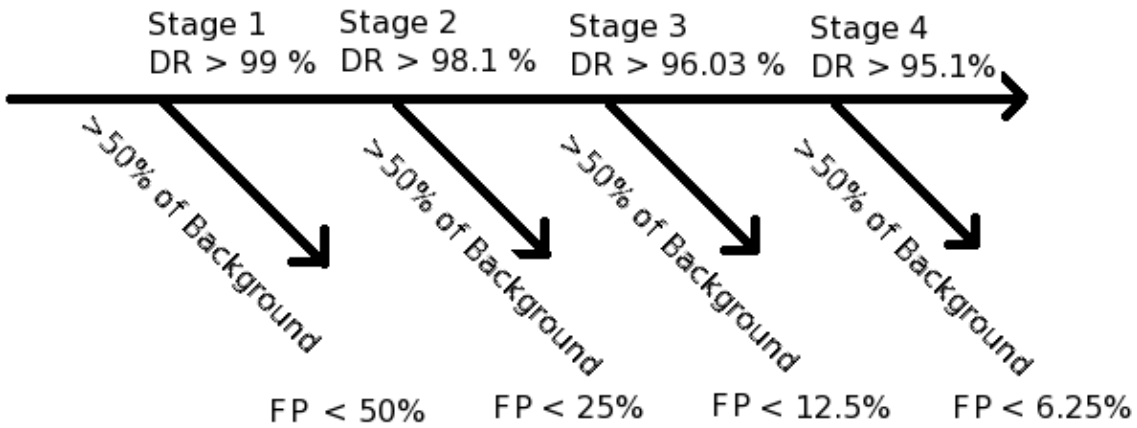
Goal: Drop out as many negative samples as possible and keep all positive ones. Only the "good" samples go through the whole cascade.



Example

Cascade trained with:

- Per stage detection rate: 0.99
- Max per stage false positive rate: 0.5
- After 4 stages:
  - Detection Rate: $0.99^4 = 0.951$
  - False Positive Rate: $0.5^4 = 0.0625$

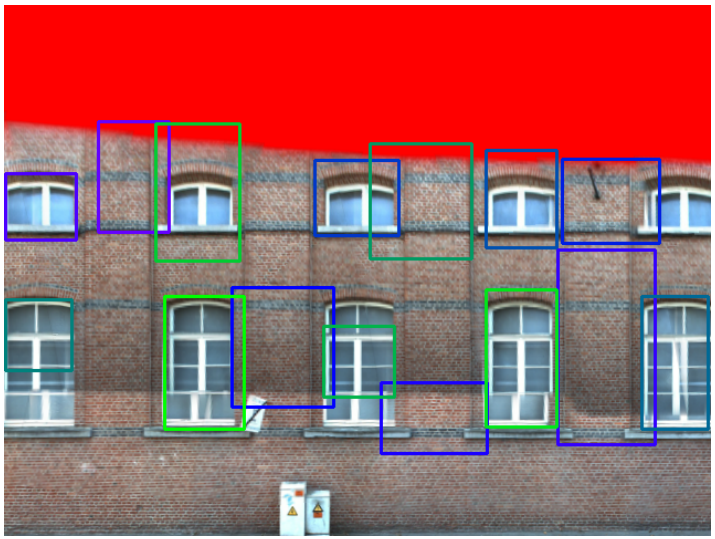




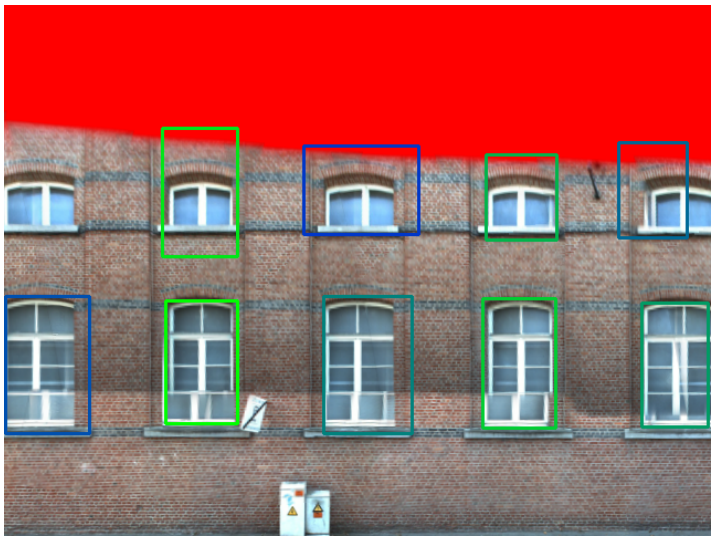
Configuration of the used Cascade

- Multiclass window detector with subclasses:
  - tall windows (#: 520)
  - wide windows (#: 246)
  - windows with closed shutters (#: 64)
  - windows with side shutters (#: 40)
  - background samples (#: 175500)
- Parameters of the cascade:
  - Stage dectection rate: 0.99
  - Stage false positive rate: > 0.5
  - Total number of stage: 7
  - Final detection rate on validation set: 0.9367
  - Final false positive rate: 0.0017

Optimization: Introduction



$\Rightarrow$



Single classifier often not strong enough.

$\Rightarrow$ prune multiple detections of the same window.

$\Rightarrow$ incorporate further information about typical facades:

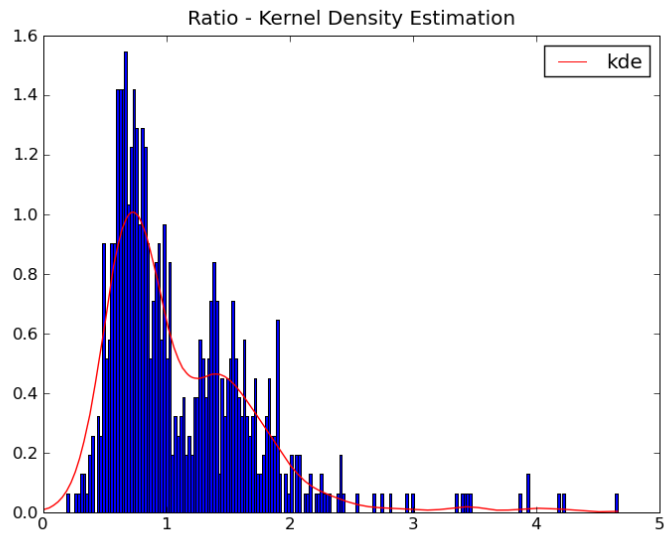
- the structure of facades (sizes, positions, configurations)
- sizes and ratios of windows
- information about symmetry.

Optimization: global and local

- Only the confidences of already detected windows are updated $\Rightarrow$ no inference.
- Local optimizations reweight the detector output of a single detection:
  - ratio (width/height)
  - typical sizes of window
  - vertical symmetry using MI
  - vertical line crossings
- Global optimizations consider the whole image(soon a facade of one house):
  - window alignment (horizontal and vertical)
  - window position

Optimizations: local

Ratio



- Kernel Densitiy Estimation from annotated Images
- used as weights for given ratios

Optimizations: local

Typical sizes of window

- In example images: 1px $\approx$ 20 cm
- Threshold for maximum window sizes
- Max height: 2m
- Max width: 5m

Optimizations: local

Vertical Symmetry

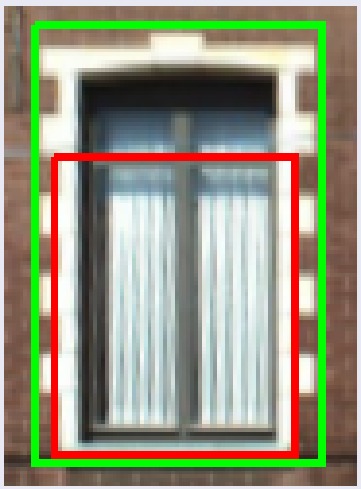


Mutual Information as a measure of symmetry:

$$I(X;Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) \log \left(\frac{p(x,y)}{p_1(x)\, p_2(y)} \right)$$

Optimizations: local

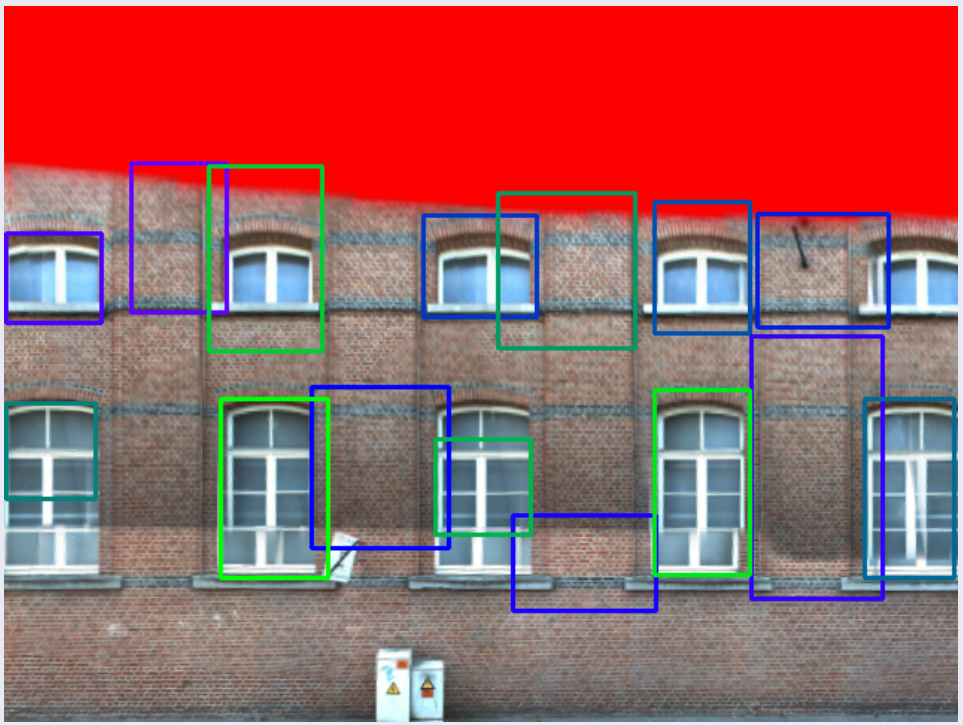
Vertical line Crossings



- Vertical lines are found via Sobel operator
- Count the edges bigger than a threshold that cut the horizontal lines of the detected bounding box
- Too many line crossings lead to reduction of weight for given bounding box

Optimizations: global

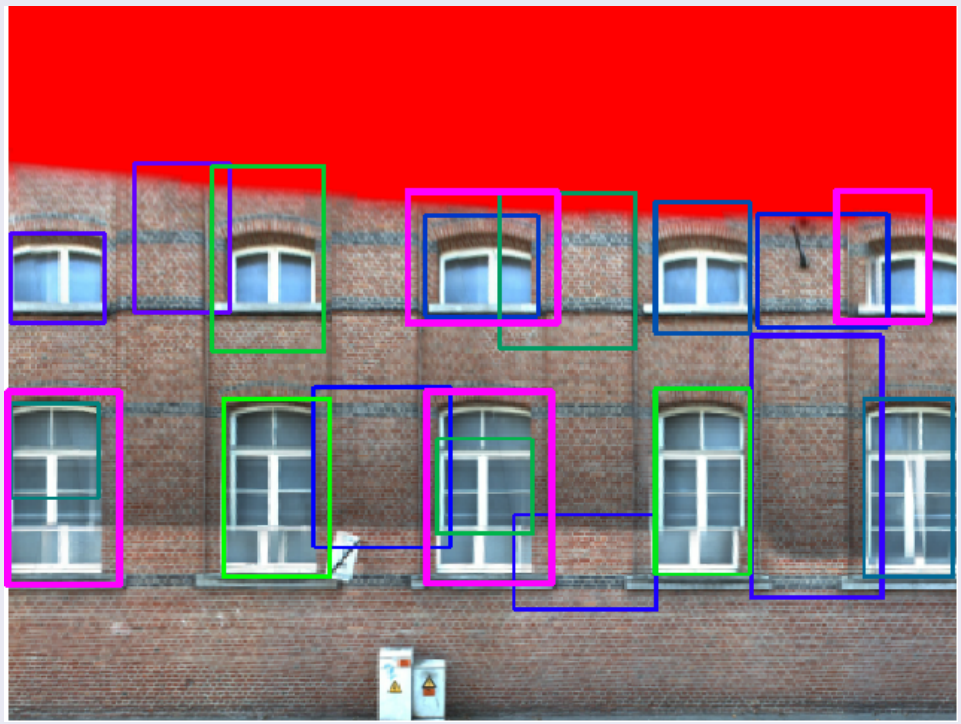
Window Alignment



- Original image

Optimizations: global

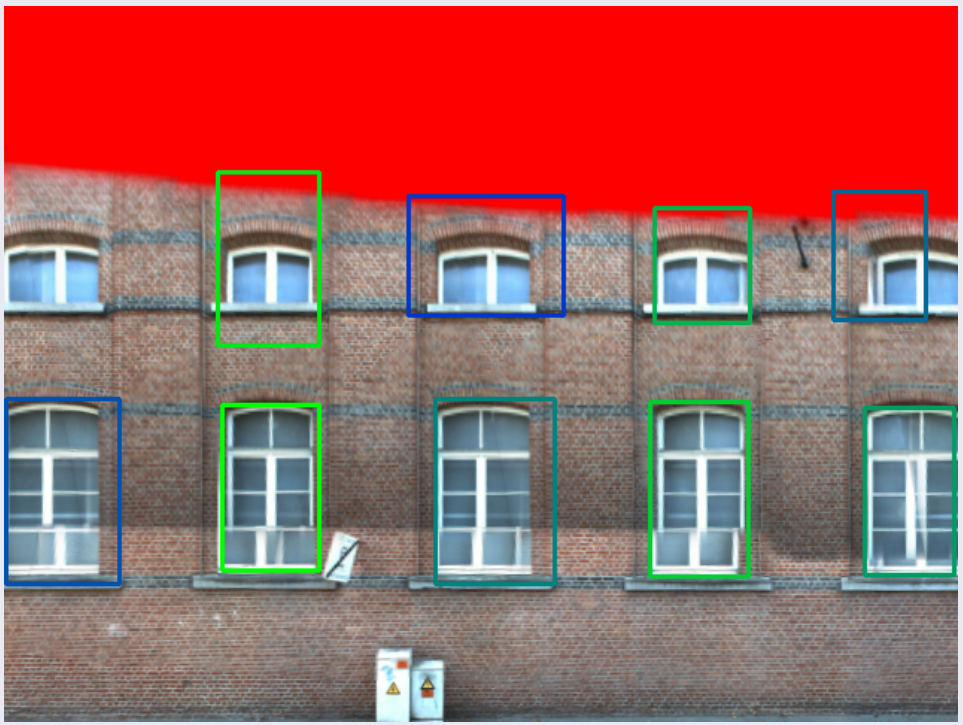
Window Alignment



- Bounding Boxes with low confidence
- weight them if they align well

Optimizations: global

Window Alignment



- Detector Result
- The weights of the Bounding Boxes is heigher

Optimizations: global

Window Position

- Windows usually do not touch ground
- No Overlap between windows
- regular distribution (not yet implemented)
- windows usually do not span over floors (not yet implemented)

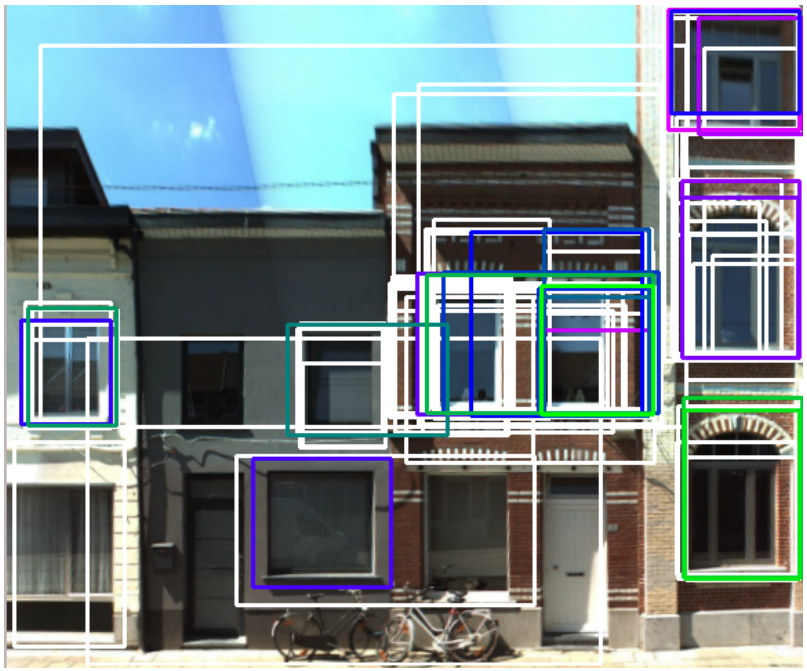
Optimization: Example

Original Image



Optimization: Example

Multiclass Detector Result



Optimization: Example

Pruning of overlapping Rectangles

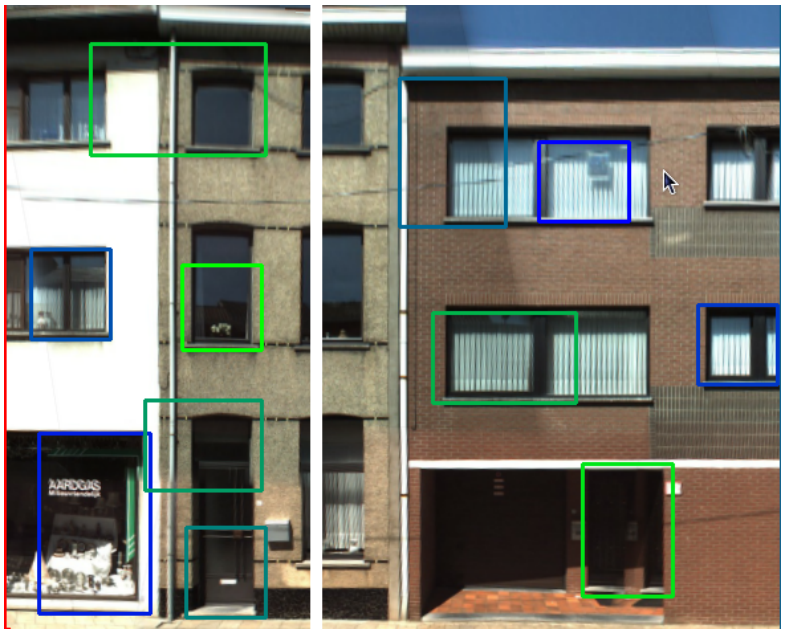


Optimization: Example

Result of the Optimization



Optimization Results



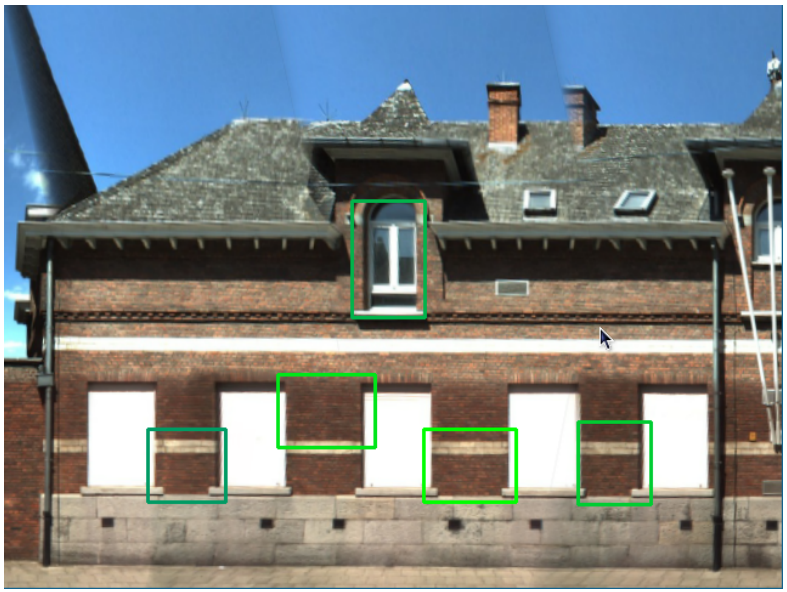


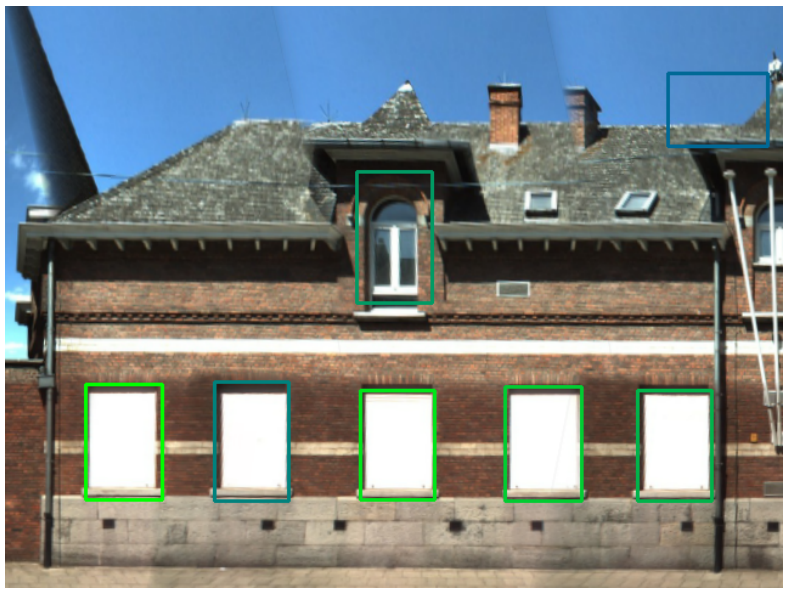
Optimization Results



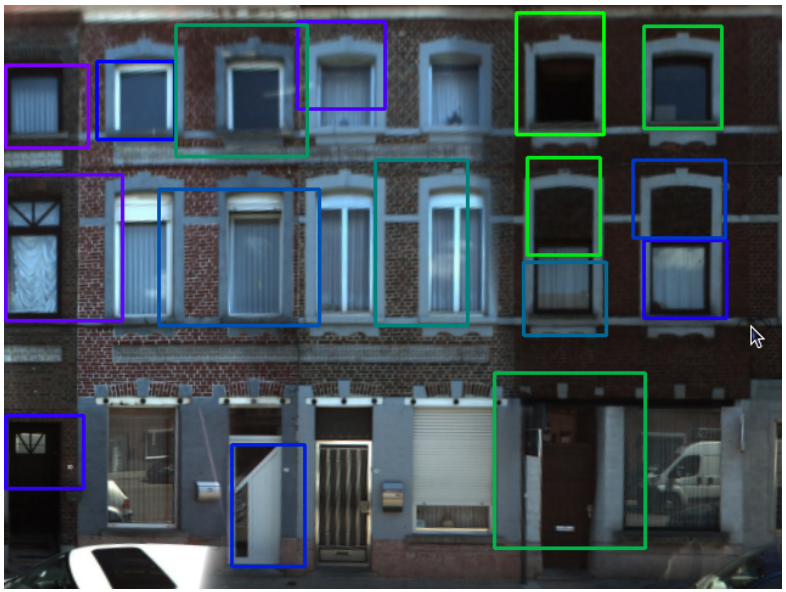


Optimization Results





Optimization Results





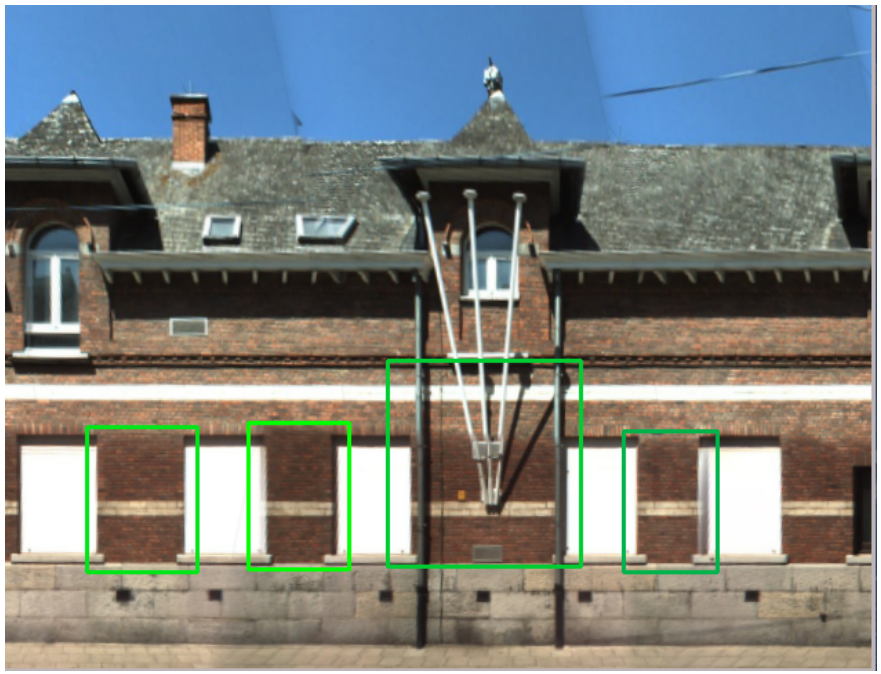
Optimization Results





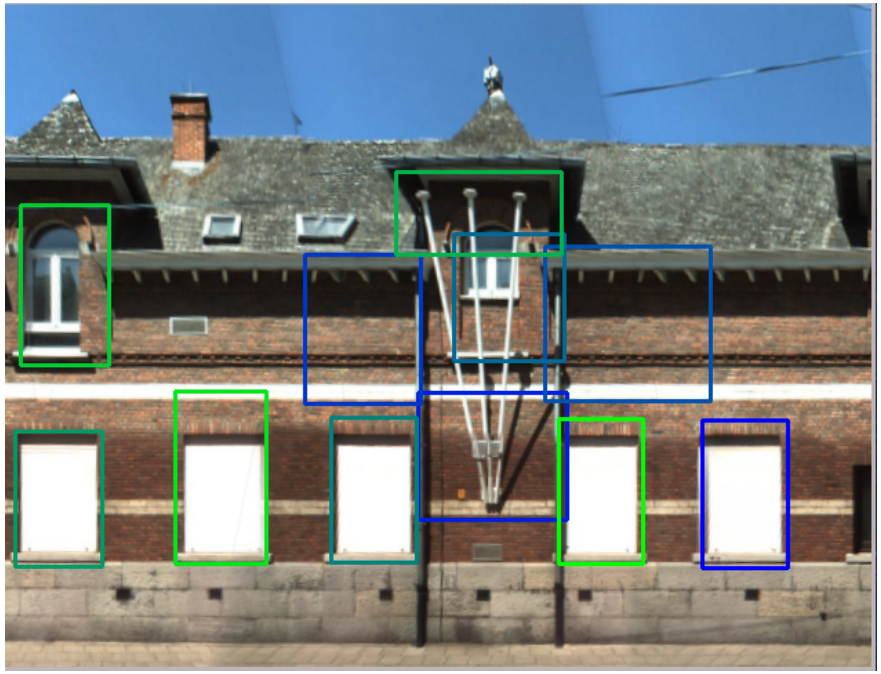
Wrongly choosen parameter

Symmetry of the detection received too much weight

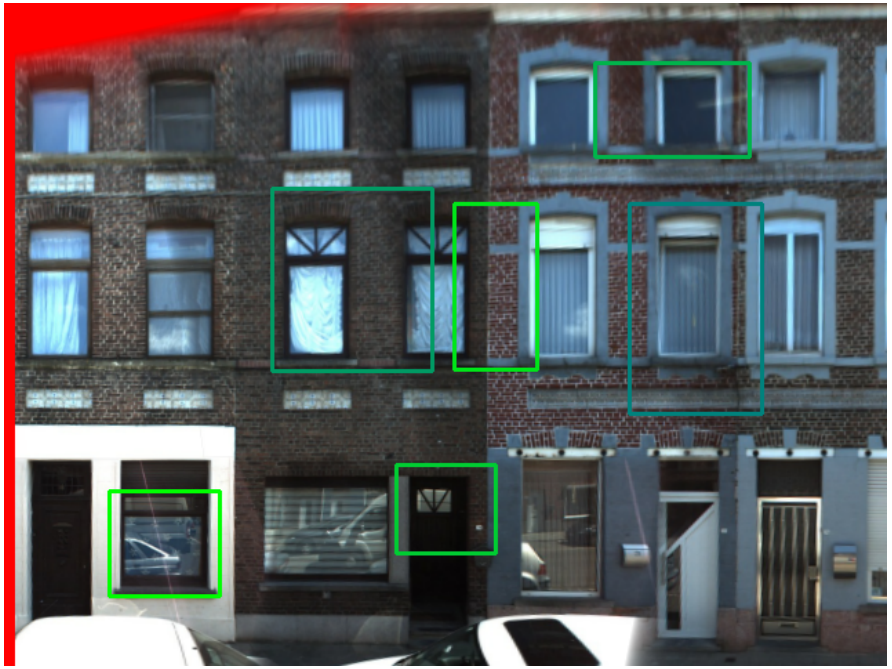


Wrongly choosen parameter

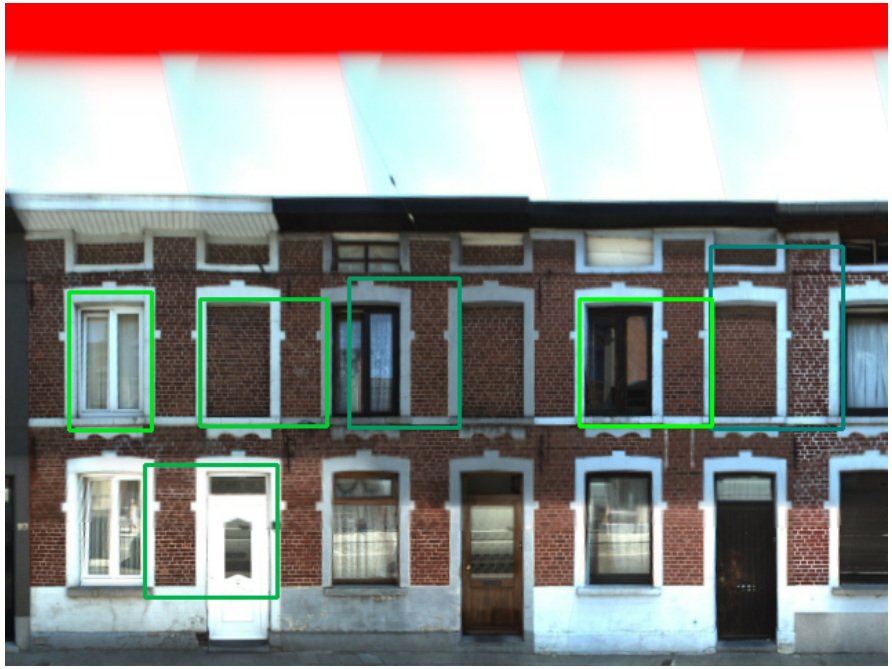
Better result



Detector failures



Detector failures





Thank you!