Semantic annotation of urban scenes: Skyline and window detection

Speaker: Tjerk Kostelijk

Supervisors:
Isaac Esteban
Prof. dr ir Frans C. A. Groen

July 5, 2012

Experiment

Experiment

Did you see?

- building

Did you see?

- building
- tree

Did you see?

- building
- tree
- bicycle

Did you see?

- building
- tree
- bicycle
- street lamb

Did you see?

- building
- tree
- bicycle
- street lamb
- blue car

Did you see?

- building
- tree
- bicycle
- street lamb
- blue car
- red car

Did you see?

- building
- tree
- bicycle
- street lamb
- blue car
- red car
- brand of the car?

Experiment

Q

- Why are we so good at depth recognition/object detection?

Q

- Why are we so good at depth recognition/object detection?
- How can we apply this to a computer system?

Q

- Why are we so good at depth recognition/object detection?
- How can we apply this to a computer system?
  - *Computer Vision*

Introduction

Skylinedetection

Extracting the 3D model

Window detection

Outline

Human perception

- Why are we so good at depth recognition/object detection?

Human perception

- Why are we so good at depth recognition/object detection?
- Depth cues

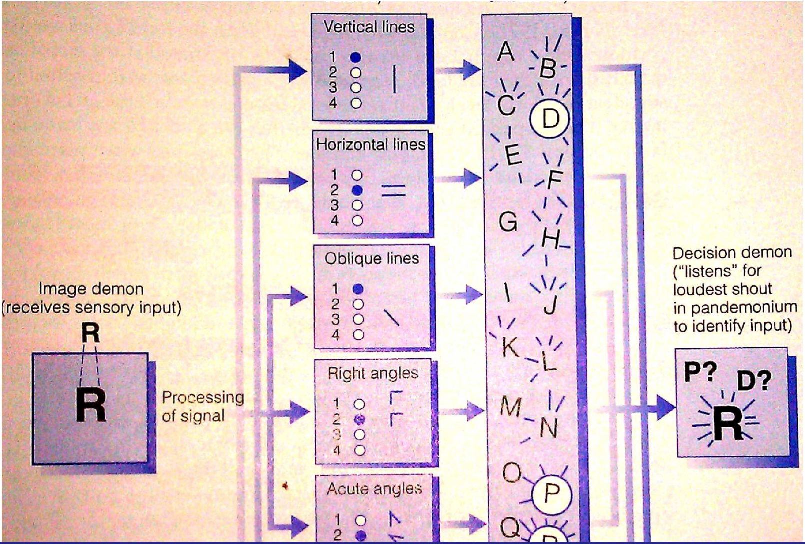
Human perception

- Why are we so good at depth recognition/object detection?
- Depth cues
- Binocular disparity

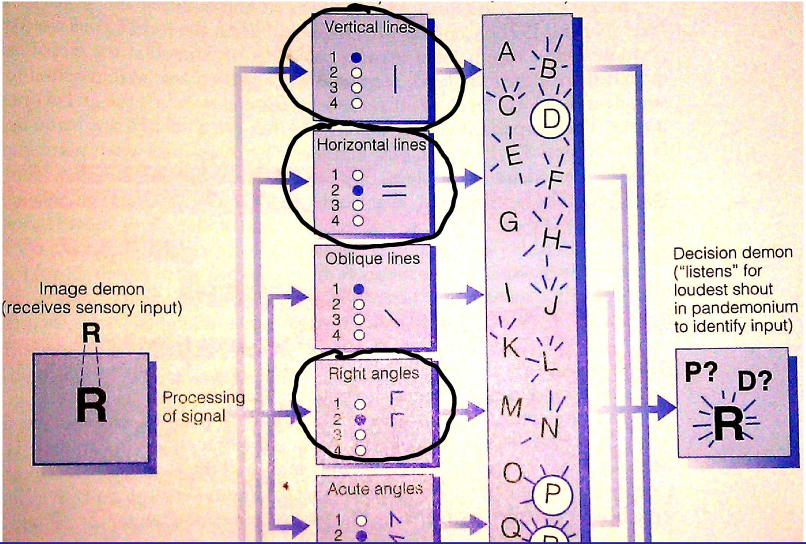
Human perception

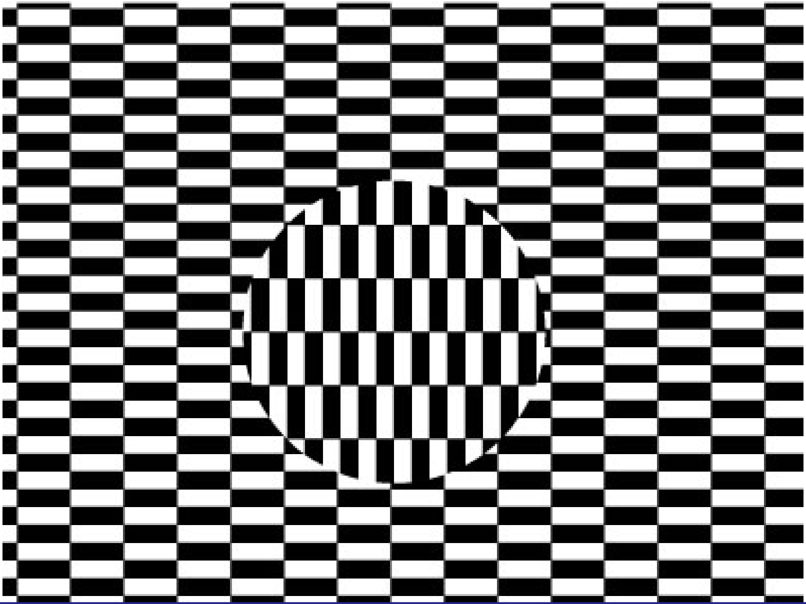
- Why are we so good at depth recognition/object detection?
- Depth cues
- Binocular disparity
- Classify objects: feature detection

Classify objects by feature detection



Classify objects by feature detection





Where is my research about?

- Annotation of urban scenes

Where is my research about?

- Annotation of urban scenes
  - Skyline detection

Where is my research about?

- Annotation of urban scenes
  - Skyline detection

Where is my research about?

- Annotation of urban scenes
  - Skyline detection

Application examples of annotation of urban scenes

- Augmented reality

Application examples of annotation of urban scenes

- Augmented reality
- Building recognition

Application examples of annotation of urban scenes

- Augmented reality
- Building recognition
- Analysis building deformation

Application examples of annotation of urban scenes

- Augmented reality
- Building recognition
- Analysis building deformation
  - 'noord-zuidlijn'

Application examples of annotation of urban scenes

- Augmented reality
- Building recognition
- Analysis building deformation
    - 'noord-zuidlijn'
    - Driving simulation

Application examples of annotation of urban scenes

- Augmented reality
- Building recognition
- Analysis building deformation
  - 'noord-zuidlijn'
  - Driving simulation
  - 3D city models

Outline

Skyline detection application example

- Horizon detection for Unmanned Air Vehicles

Skyline detection in urban scenes

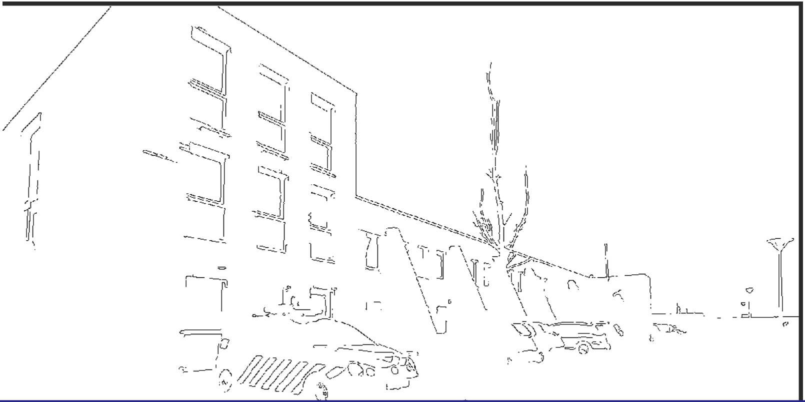
- Edges detection based on intensity change

Skyline detection in urban scenes

- Edges detection based on intensity change
- Canny edge detection

Skyline detection in urban scenes

- Edges detection based on intensity change
- Canny edge detection
- Result: Binary image (edge or no edge)



Skyline detection algorithm

- Top sharp edge assumption
  "The first sharp edge (seen from top to bottom) in the image represents the skyline."
- Algorithm:

Skyline detection algorithm

- Top sharp edge assumption
  "The first sharp edge (seen from top to bottom) in the image represents the skyline."
- Algorithm:
- Apply gaussian smoothing

Skyline detection algorithm

- Top sharp edge assumption
  "The first sharp edge (seen from top to bottom) in the image represents the skyline."
- Algorithm:
- Apply gaussian smoothing
- The image is sliced in #w pixelcolumns

Skyline detection algorithm

- Top sharp edge assumption
  "The first sharp edge (seen from top to bottom) in the image represents the skyline."
- Algorithm:
- Apply gaussian smoothing
- The image is sliced in #w pixelcolumns
- Each column present #h binary edge values (edge or no edge)

Skyline detection algorithm

- Top sharp edge assumption
  "The first sharp edge (seen from top to bottom) in the image represents the skyline."
- Algorithm:
- Apply gaussian smoothing
- The image is sliced in #w pixelcolumns
- Each column present #h binary edge values (edge or no edge)
- y-location of the first edge value is stored

Skyline detection result

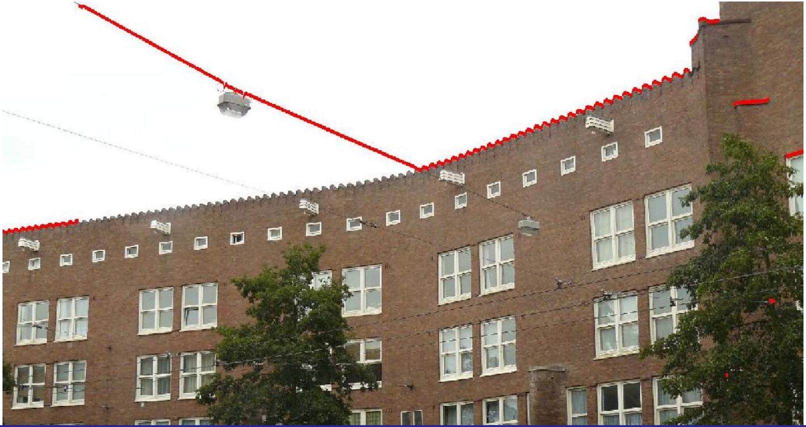
Skyline detection result

Future research

Hypothesis based skyline detection, assumption

"The first sharp edge (seen from top to bottom) in the image represents the skyline."

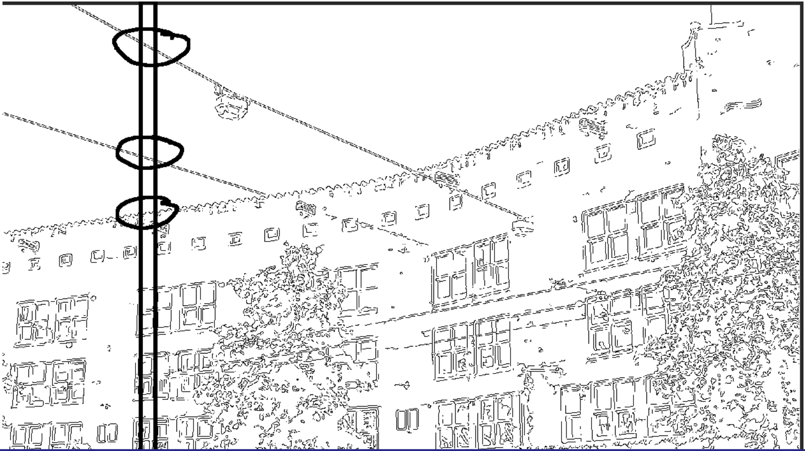
- Example of a scene where this assumption is violated



Hypothesis based skyline detection, assumption

- Change of assumption

"The skyline is part of the first n sharp edges (e.g. $n = 3$)

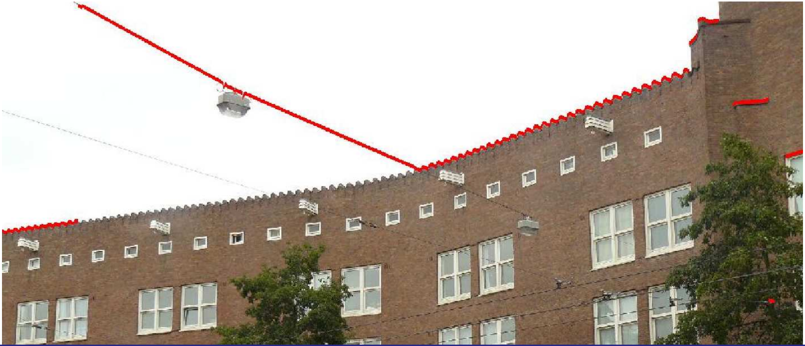


Hypothesis based skyline detection, algorithm

- Generate n hypothesis

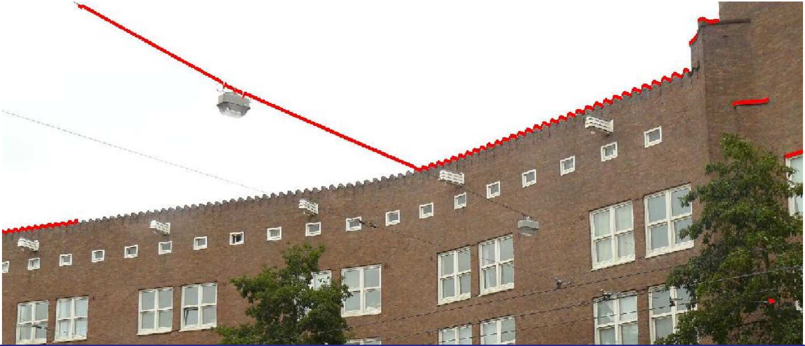
Hypothesis based skyline detection, algorithm

- Generate *n* hypothesis
- Classify hypothesis with additional info



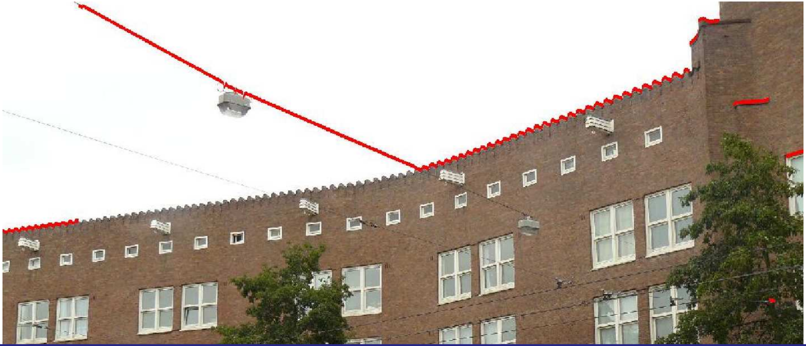
Hypothesis based skyline detection, algorithm

- Generate *n* hypothesis
- Classify hypothesis with additional info
  - texture



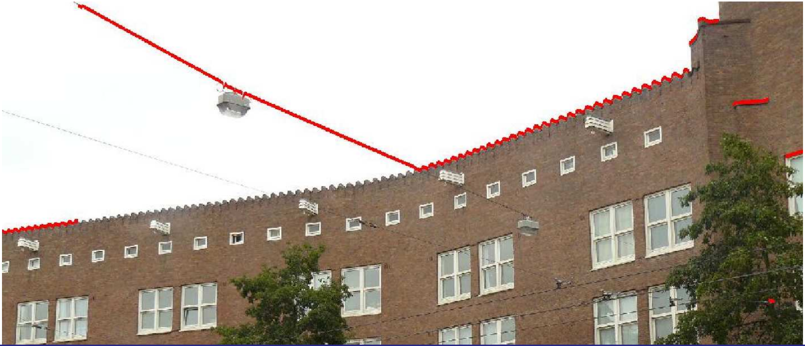
Hypothesis based skyline detection, algorithm

- Generate *n* hypothesis
- Classify hypothesis with additional info
  - texture
  - color



Hypothesis based skyline detection, algorithm

- Generate *n* hypothesis
- Classify hypothesis with additional info
  - texture
  - color
  - height variation



Expected result on hypothesis classification based on color

Expected result classification based on heigth variation

Expected result classification based on heigth variation

Conclusion

- The skyline deteciton algorithm

Conclusion

- The skyline deteciton algorithm
  - is simple and has a low complexity

Conclusion

- The skyline deteciton algorithm
  - is simple and has a low complexity
  - works wel under the assumption skyline is upper edge

Conclusion

- The skyline deteciton algorithm
  - is simple and has a low complexity
  - works wel under the assumption skyline is upper edge
  - needs future research

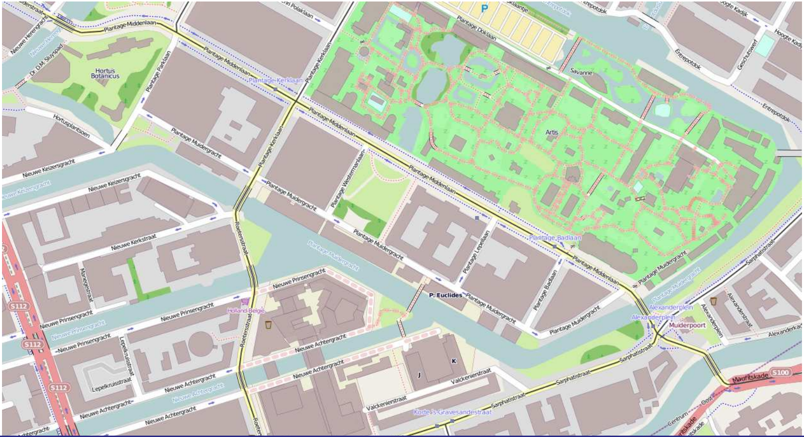
Conclusion

- The skyline deteciton algorithm
  - is simple and has a low complexity
  - works wel under the assumption skyline is upper edge
  - needs future research
    - can provide a set of hypothesis which should be evaluated using additional features (e.g. color and height variation)

Outline

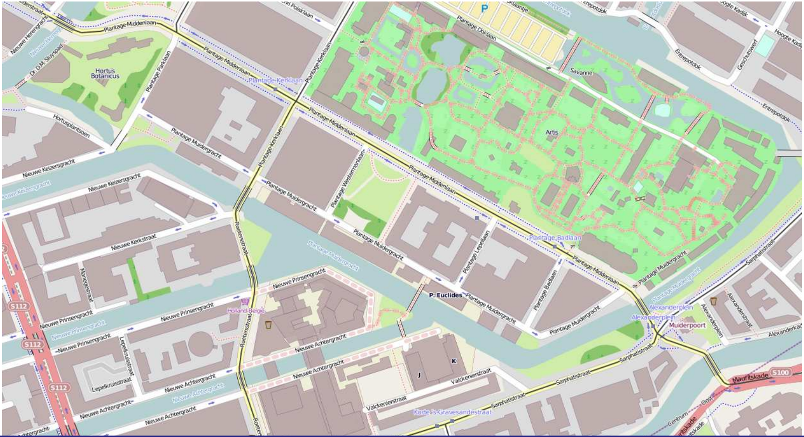
Overview

- Create (top view) 2D model of the scene using *Openstreetmap*



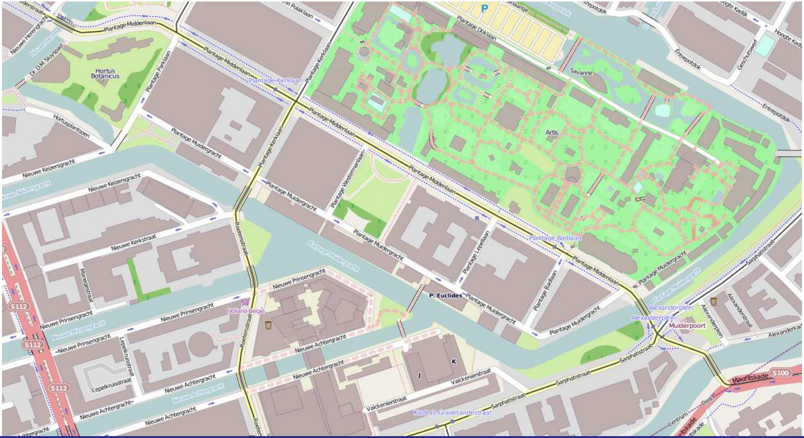
Overview

- Create (top view) 2D model of the scene using *Openstreetmap*



Overview

- Create (top view) 2D model of the scene using *Openstreetmap*

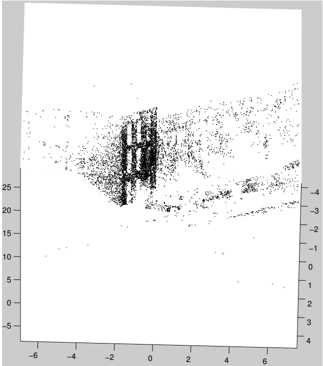


Extract 2D model

Align 2D model

- *FIT3D toolbox*

(a)

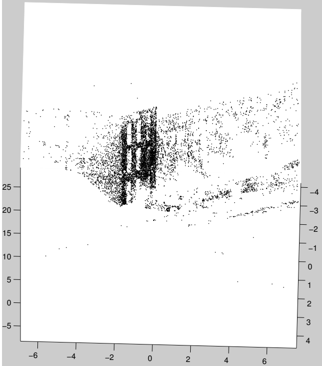


(b)

Align 2D model

- *FIT3D toolbox*
  - Input: sequence of images that contain different views of the building

(c)

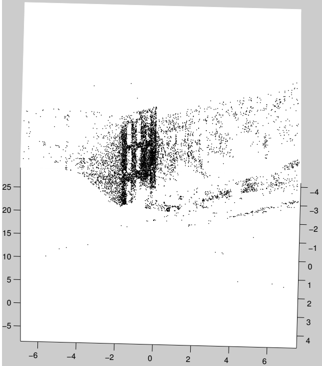


(d)

Align 2D model

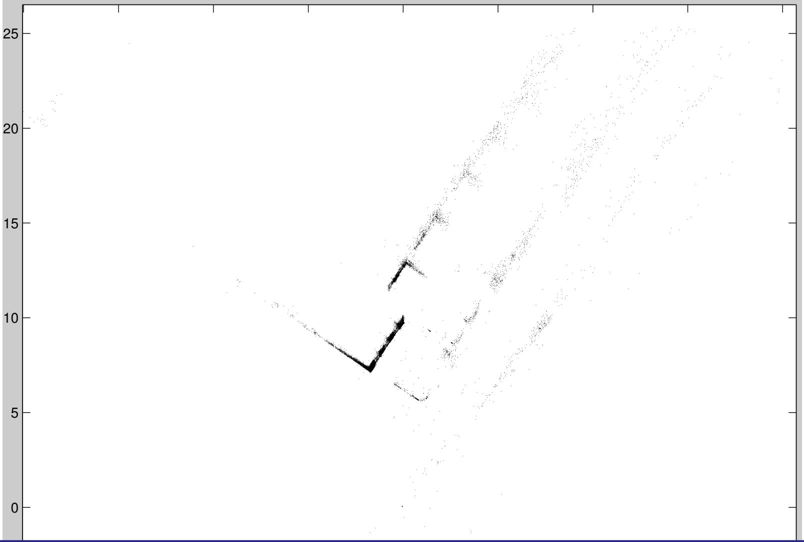
- *FIT3D toolbox*
  - Input: sequence of images that contain different views of the building
  - Output: a 3D point cloud of the building

(e)

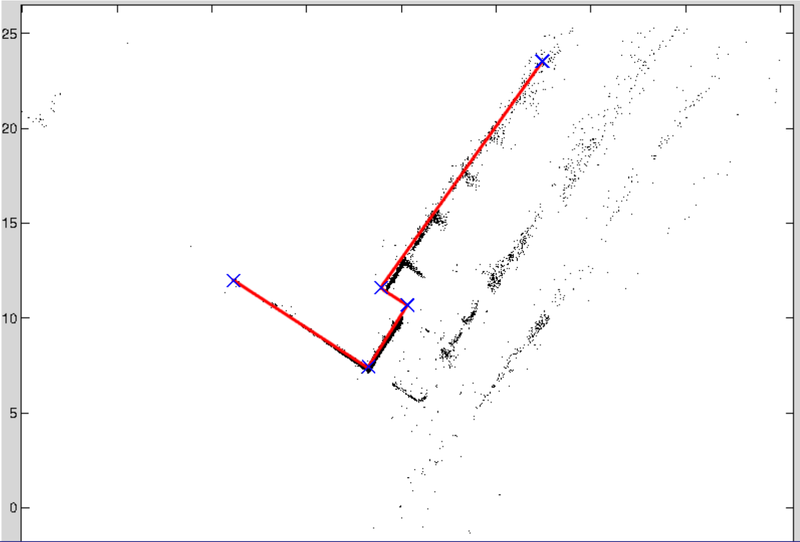


(f)

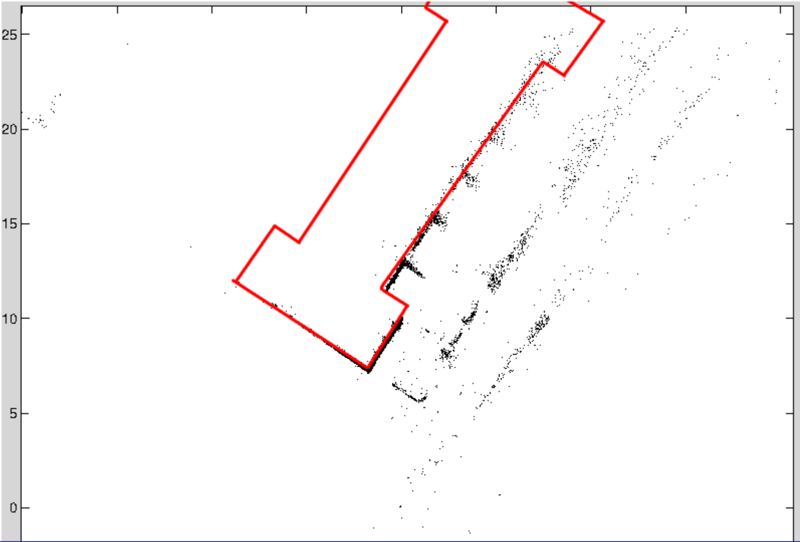
Align 2D model



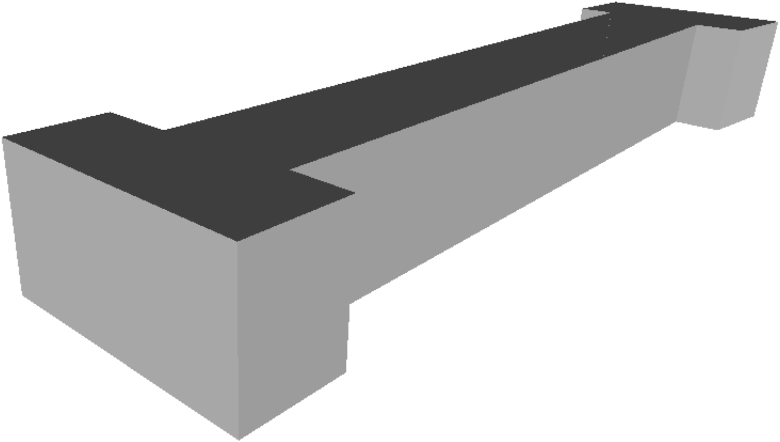
Align 2D model



Align 2D model

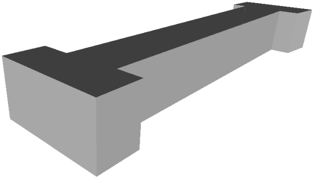


Results



Wall height estimation

- Wall heights of 3D model are not accurate

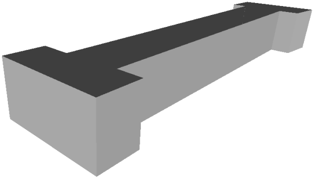


(g)

(h)

Wall height estimation

- Wall heights of 3D model are not accurate



(i)

(j)

- Improve the 3D model by wall height estimation

Assumptions

Straight lines in the skyline are likely to come from the building contour Flat roof assumption -¿ building contour is equal to upper side of building walls

Extracting line segments

- Output of skyline detector

Extracting line segments

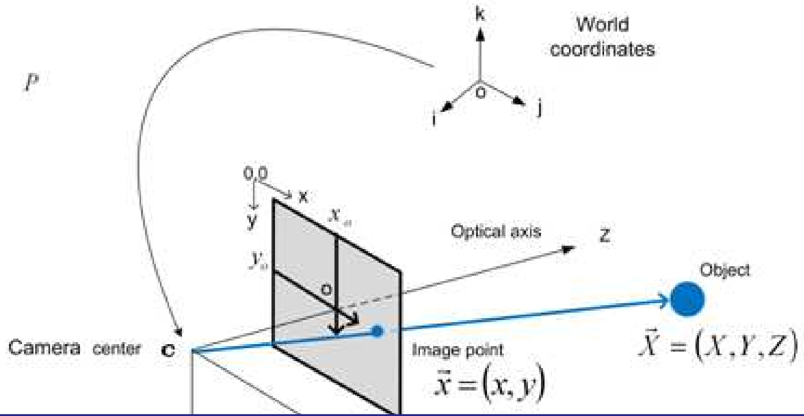
- Output of skyline detector
- Hough transform

Project to the 3D model

- Estimate wall heights

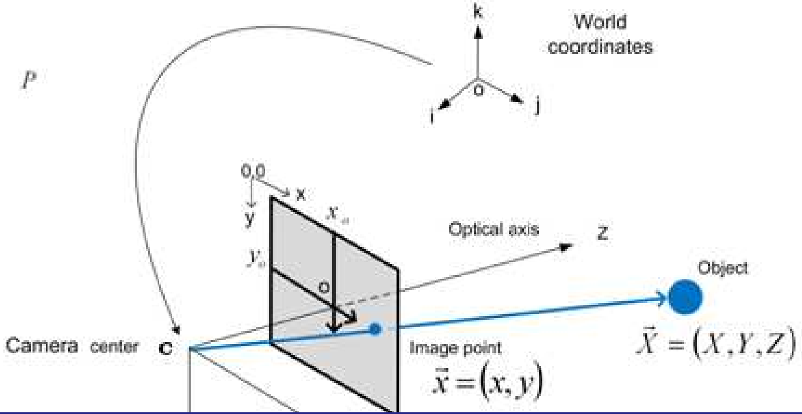
Project to the 3D model

- Estimate wall heights
- project line segments to specific walls of 3D model

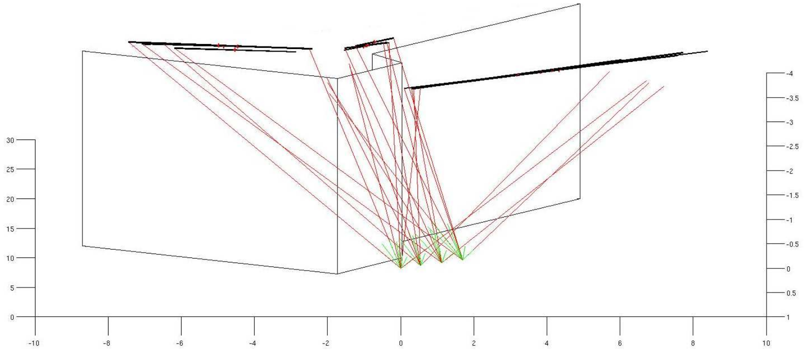


Project to the 3D model

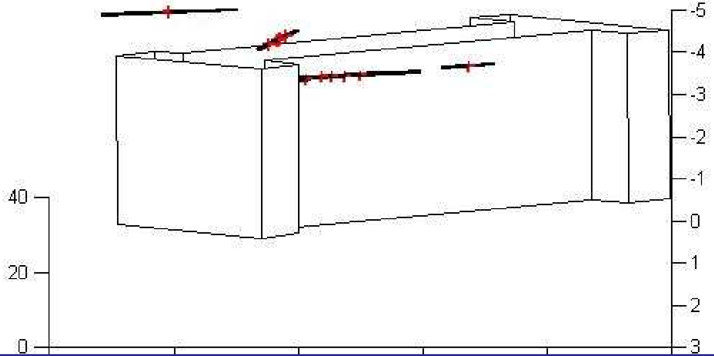
- Estimate wall heights
- project line segments to specific walls of 3D model



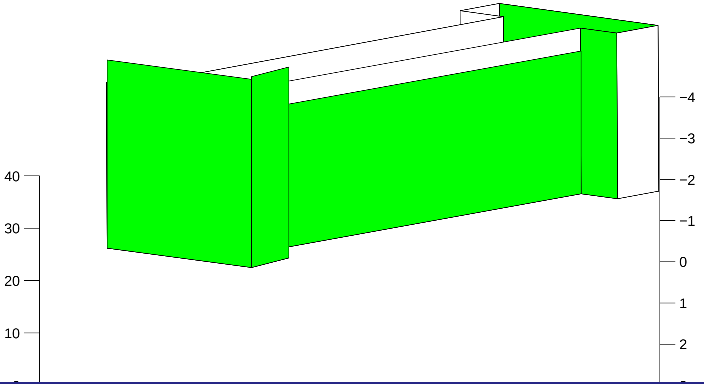
Results 1/3



Results 2/3



Results 3/3



Future research

Overview

Outline

Introduction

Skylinedetection

Extracting the 3D model

Window detection

Outline Window detection

- Method I: Connected corner approach

Outline Window detection

- Method I: Connected corner approach
  - invariant to viewing direction

Outline Window detection

- Method I: Connected corner approach
  - invariant to viewing direction
- Facade rectification

Outline Window detection

- Method I: Connected corner approach
  - invariant to viewing direction
- Facade rectification
- Method II: Histogram based approach

Outline Window detection

- Method I: Connected corner approach
  - invariant to viewing direction
- Facade rectification
- Method II: Histogram based approach
  - rectified facades

Outline Window detection

- Method I: Connected corner approach
  - invariant to viewing direction
- Facade rectification
- Method II: Histogram based approach
  - rectified facades
  - uses Histograms of Houghlines

Method I: Connected corner approach

Situation

- Window frames

Situation

- Window frames
- Difference color window frame, color glass

Situation

- Window frames
- Difference color window frame, color glass
- Produce edges in horizontal and vertical direction

The idea

- Connected corner

The idea

- Connected corner
- *"horizontal and vertical edges that come from the same (sub)window frame share a corner"*

Edge detection and Houghline extraction

- Canny edge detector

Edge detection and Houghline extraction

- Canny edge detector
- Two groups of straight lines (Houghlines)

Edge detection and Houghline extraction

- Canny edge detector
- Two groups of straight lines (Houghlines)
  - Horizontal, $\theta = [-30..0..30)$ degrees

Edge detection and Houghline extraction

- Canny edge detector
- Two groups of straight lines (Houghlines)
  - Horizontal, $\theta = [-30..0..30)$ degrees
  - Vertical, $\theta = [80..90..100)$ degrees

Edge detection and Houghline extraction

- Canny edge detector
- Two groups of straight lines (Houghlines)
  - Horizontal, $\theta = [-30..0..30)$ degrees
  - Vertical, $\theta = [80..90..100)$ degrees
- Why the use of angle ranges?

Edge detection and Houghline extraction

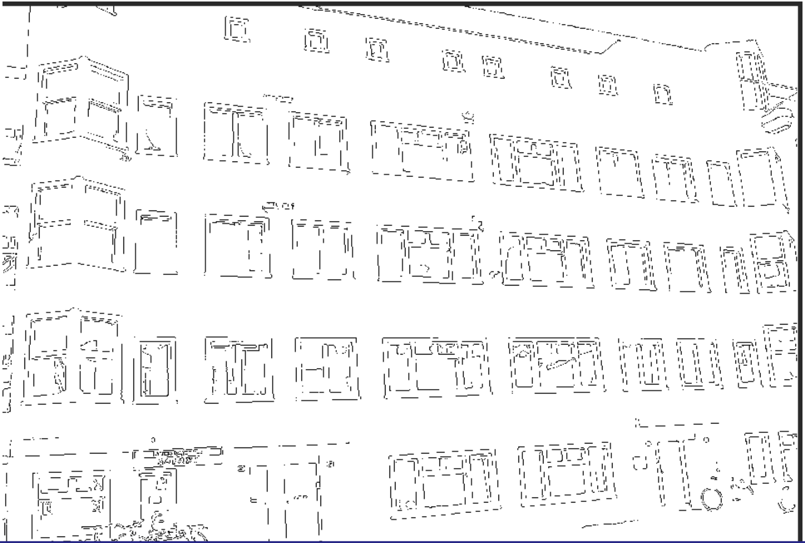
- Canny edge detector
- Two groups of straight lines (Houghlines)
  - Horizontal, $\theta = [-30..0..30)$ degrees
  - Vertical, $\theta = [80..90..100)$ degrees
- Why the use of angle ranges?
  - Camera not exactly upright

Edge detection and Houghline extraction

- Canny edge detector
- Two groups of straight lines (Houghlines)
  - Horizontal, $\theta = [-30..0..30)$ degrees
  - Vertical, $\theta = [80..90..100)$ degrees
- Why the use of angle ranges?
  - Camera not exactly upright
  - Facade view contains an angle, perspective distortion

Original

Edge detection



Result of θ constrained Hough transform

Connected corner

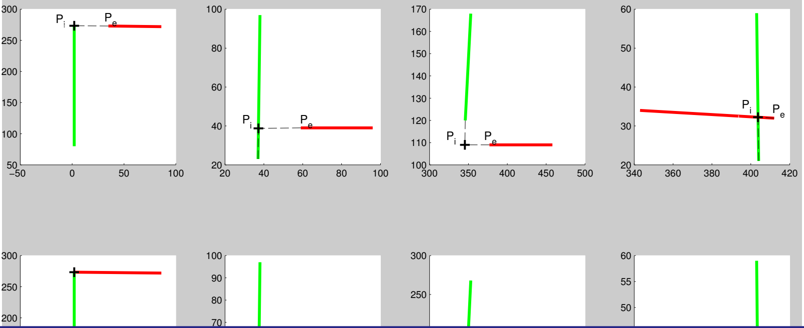
- Connected corner:a horizontal and vertical edge that share the same corner e.g. L

Connected corner

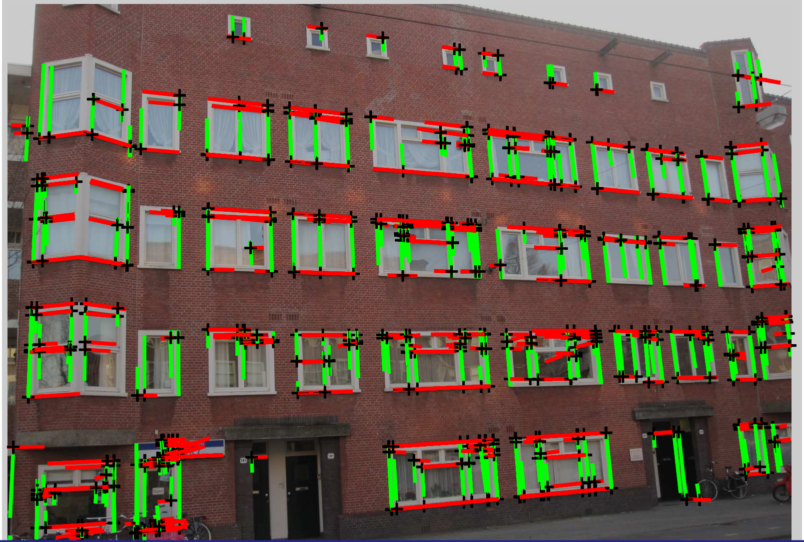
- Connected corner:a horizontal and vertical edge that share the same corner e.g. L
- Algorithm: If line segments endpoints are close enough a connected corner is formed

Connected corner

- Connected corner:a horizontal and vertical edge that share the same corner e.g. L
- Algorithm: If line segments endpoints are close enough a connected corner is formed
- Tolerate gab or extension



Connected Corners



Window area extraction

- Mirror from diagonal through endpoints

Window area extraction

- Mirror from diagonal through endpoints
- L shape -¿ Quadrangle window area

Results

Future research

- L-shapes, U-shapes

Future research

- L-shapes, U-shapes
- analysis of substructure of windows

Future research

- L-shapes, U-shapes
- analysis of substructure of windows
  - define a contained in relations

Future research

- L-shapes, U-shapes
- analysis of substructure of windows
  - define a contained in relations
  - cluster connected corner on location and length

Future research

- L-shapes, U-shapes
- analysis of substructure of windows
  - define a contained in relations
  - cluster connected corner on location and length
    - maximum intercluster distance correlates with size subwindow

Future research

- L-shapes, U-shapes
- analysis of substructure of windows
  - define a contained in relations
  - cluster connected corner on location and length
    - maximum intercluster distance correlates with size subwindow
    - assume nr of subwindows to estimate size

Method II: window detection

- Assumes that the windows are

Method II: window detection

- Assumes that the windows are
  - orthogonal

Method II: window detection

- Assumes that the windows are
  - orthogonal
  - aligned

Aligned but not orthogonal

Rectified facade

Dataset: Anne1, Rectified image0.45

Facade rectification

- 3D model gives plane that corresponds to wall

Facade rectification

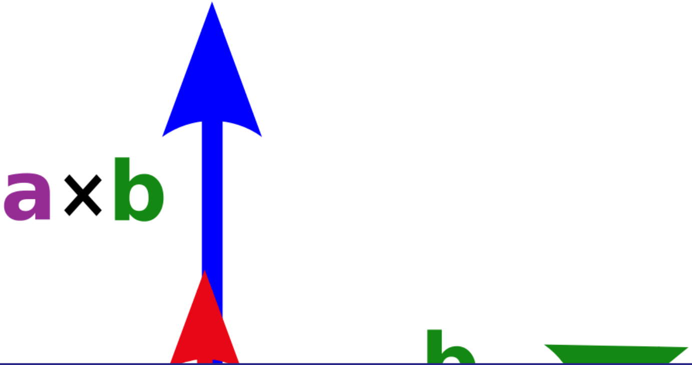
- 3D model gives plane that corresponds to wall
- wall produces a normal vector b

Facade rectification

- 3D model gives plane that corresponds to wall
- wall produces a normal vector b
- camera's heading, $\vec{a}$

Facade rectification

- 3D model gives plane that corresponds to wall
- wall produces a normal vector b
- camera's heading, $\vec{a}$
- rotation matrix R is calculated and applied



Unrectified facade

Rectified facade

Dataset: Anne1, Rectified image0.45


Heren Kapsalon
SAFARI


Heren Kapsalon
SAFARI

Extracting the window alignment

- Window alignment line

Extracting the window alignment

- Window alignment line
  - *"A horizontal or vertical line that aligns multiple windows"*

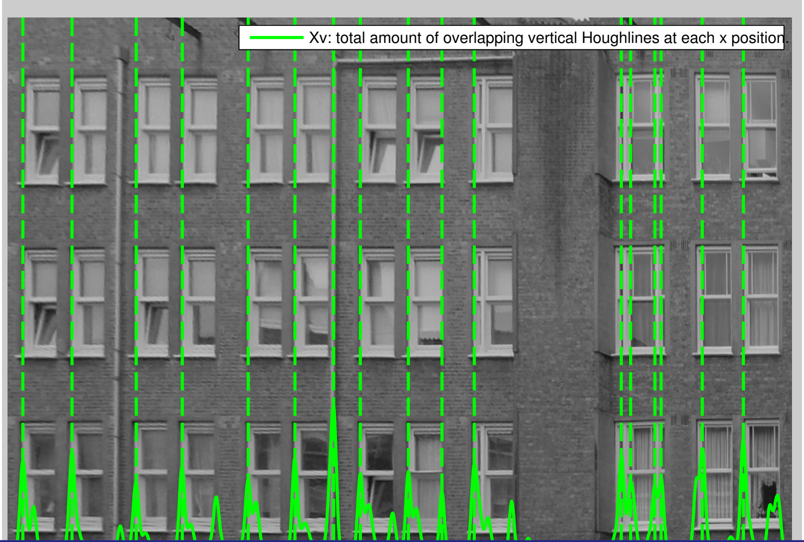
Extracting the window alignment

- Window alignment line
  - *"A horizontal or vertical line that aligns multiple windows"*
- Alignment lines provide grid of blocks

Extracting the window alignment

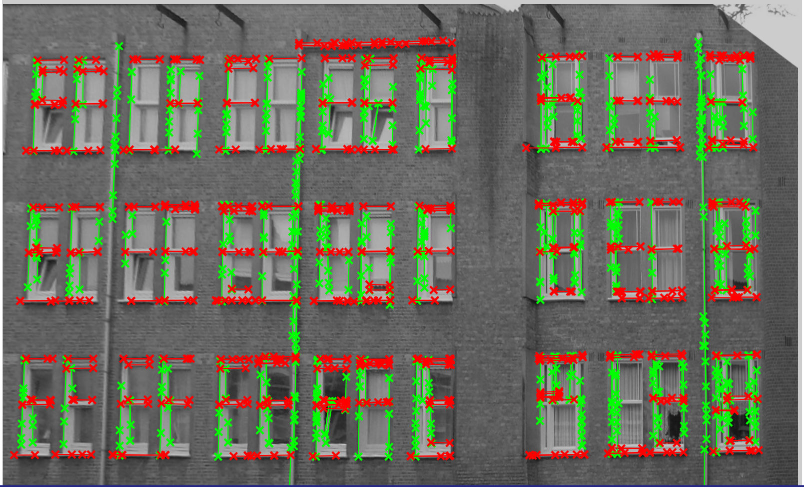
- Window alignment line
  - *"A horizontal or vertical line that aligns multiple windows"*
- Alignment lines provide grid of blocks
- Window non-window areas

Window alignment lines example



The idea

- Amount of Houghlines high at window locations



The algorithm for vertical alignment (columns)

- aim: get positions of window columns

The algorithm for vertical alignment (columns)

- aim: get positions of window columns
- isolate *n* horizontal Hough lines

The algorithm for vertical alignment (columns)

- aim: get positions of window columns
- isolate *n* horizontal Hough lines
- extract pixel coordinates of endpoints

The algorithm for vertical alignment (columns)

- aim: get positions of window columns
- isolate *n* horizontal Hough lines
- extract pixel coordinates of endpoints
  - -¿set of (2 dimensional) coordinates (x,y)

The algorithm for vertical alignment (columns)

- aim: get positions of window columns
- isolate *n* horizontal Hough lines
- extract pixel coordinates of endpoints
  - -¿set of (2 dimensional) coordinates (x,y)
- discard least informative dimension

The algorithm for vertical alignment (columns)

- aim: get positions of window columns
- isolate *n* horizontal Hough lines
- extract pixel coordinates of endpoints
  - -¿set of (2 dimensional) coordinates (x,y)
- discard least informative dimension
  - project to X-axis (by discard Y-value)

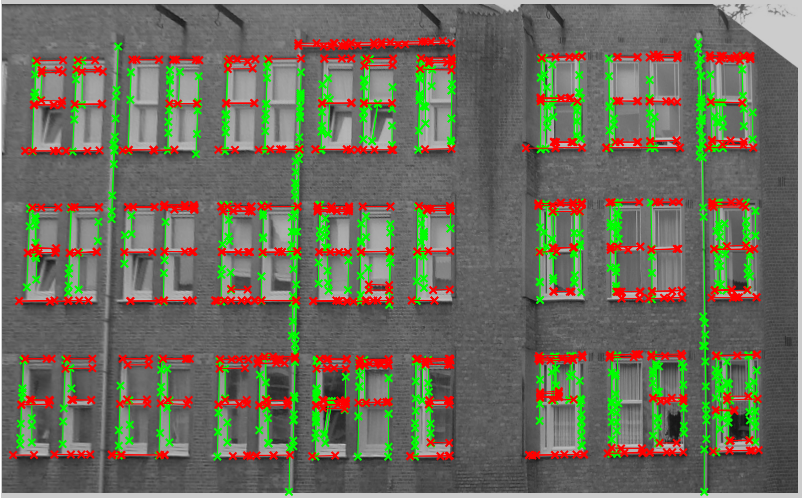
The algorithm for vertical alignment (columns)

- aim: get positions of window columns
- isolate *n* horizontal Hough lines
- extract pixel coordinates of endpoints
  - -¿set of (2 dimensional) coordinates (x,y)
- discard least informative dimension
  - project to X-axis (by discard Y-value)
  - gives a set of 2*n* x values

The algorithm for vertical alignment (columns)

- aim: get positions of window columns
- isolate *n* horizontal Hough lines
- extract pixel coordinates of endpoints
  - -¿set of (2 dimensional) coordinates (x,y)
- discard least informative dimension
  - project to X-axis (by discard Y-value)
  - gives a set of 2*n* x values
- Create histogram: count number of values on each x position

Window alignment lines



Peak (area) extraction

- smooth histogram function (red line)

Peak (area) extraction

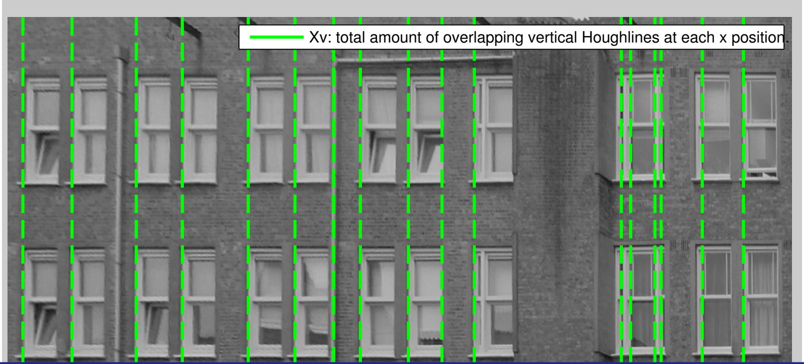
- smooth histogram function (red line)
- relative threshold 0.5 * max peak defines peak areas

Peak (area) extraction

- smooth histogram function (red line)
- relative threshold 0.5 * max peak defines peak areas
- locate maximum for each peak area

Peak (area) extraction

- smooth histogram function (red line)
- relative threshold 0.5 * max peak defines peak areas
- locate maximum for each peak area
- draw alignment line on peaks



Reasons for improvement

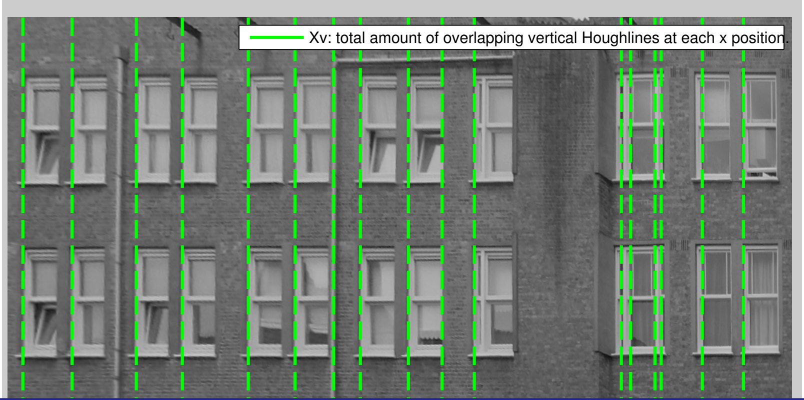
- window alignment at wrong locations

Reasons for improvement

- window alignment at wrong locations
- right side of window frame of first 4 columns not found

Reasons for improvement

- window alignment at wrong locations
- right side of window frame of first 4 columns not found
- main reason: perspective distortion creates occlusion effect




Heren Kapsalon
SAFARI

Alternative window alignment

- Previous method:

Alternative window alignment

- Previous method:
  - line segment endpoints

Alternative window alignment

- Previous method:
  - line segment endpoints
  - vertical lines -¿ window columns

Alternative window alignment

- Previous method:
  - line segment endpoints
  - vertical lines -¿ window columns
  - histogram peak detection

Alternative window alignment

- Previous method:
  - line segment endpoints
  - vertical lines -¿ window columns
  - histogram peak detection
- Alternative method:

Alternative window alignment

- Previous method:
  - line segment endpoints
  - vertical lines -¿ window columns
  - histogram peak detection
- Alternative method:
  - entire line segment

Alternative window alignment

- Previous method:
  - line segment endpoints
  - vertical lines -¿ window columns
  - histogram peak detection
- Alternative method:
  - entire line segment
  - horizontal lines -¿ window columns

Alternative window alignment

- Previous method:
  - line segment endpoints
  - vertical lines -¿ window columns
  - histogram peak detection
- Alternative method:
  - entire line segment
  - horizontal lines -¿ window columns
  - histogram shape analysis

Alternative window alignment

- Idea

Alternative window alignment

- Idea
  - a window frame creates horizontal edges

Alternative window alignment

- Idea
  - a window frame creates horizontal edges
  - on vertical alignment positions appears a big increase/decrease of horizontal lines as the window starts/end

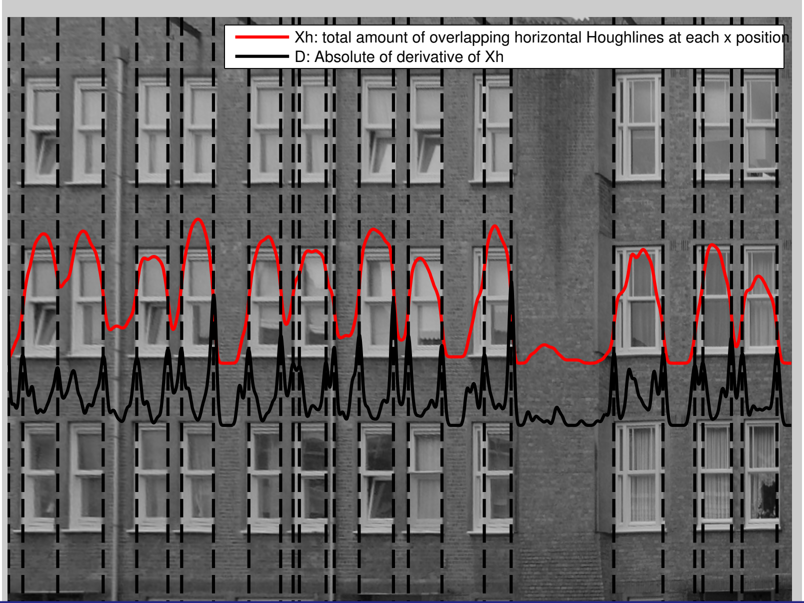
Alternative window alignment

- Idea
  - a window frame creates horizontal edges
  - on vertical alignment positions appears a big increase/decrease of horizontal lines as the window starts/end
- X_h defines number of overlapping horizontal lines on each x position

Alternative window alignment

- Idea
  - a window frame creates horizontal edges
  - on vertical alignment positions appears a big increase/decrease of horizontal lines as the window starts/end
- X_h defines number of overlapping horizontal lines on each x position
- peak function

$$D = abs(X_h')$$


Xh: total amount of overlapping horizontal Houghlines at each x position
D: Absolute of derivative of Xh

Fusing the window alignment methods

- plot both methods

Fusing the window alignment methods

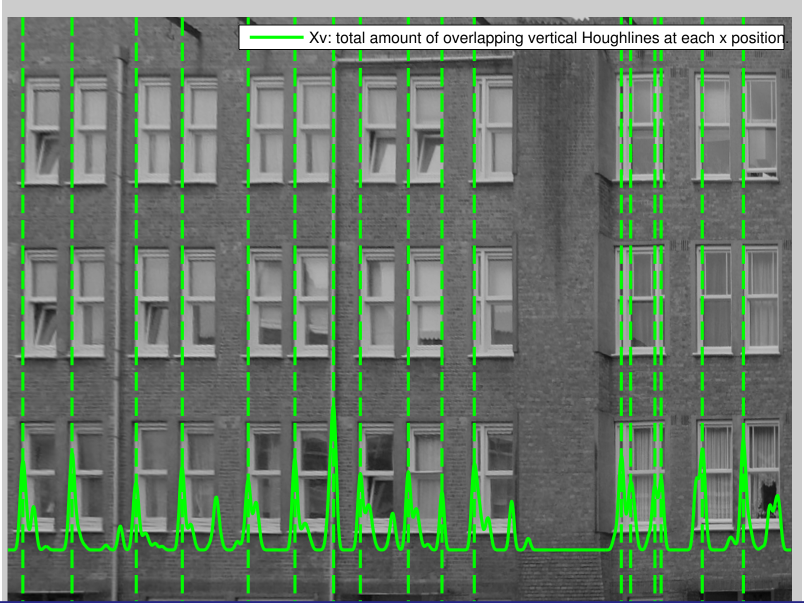
- plot both methods
- increase threshold (less but more certain alignment lines)

Fusing the window alignment methods

- plot both methods
- increase threshold (less but more certain alignment lines)
- merge peaks that are close

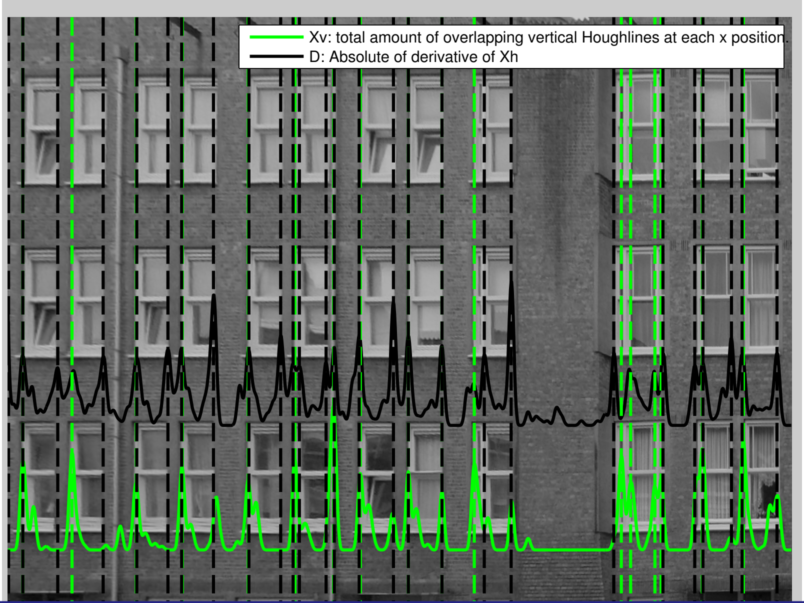
Fusing the window alignment methods

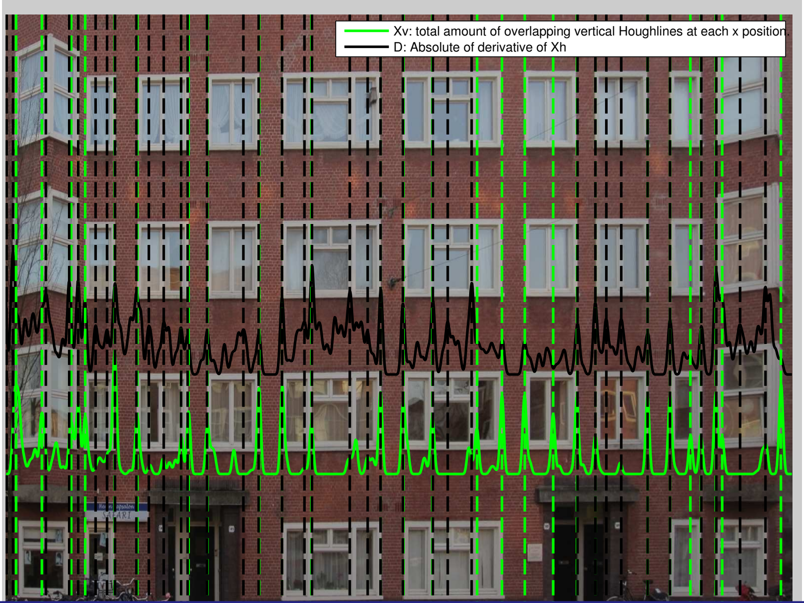
- plot both methods
- increase threshold (less but more certain alignment lines)
- merge peaks that are close
- plotting both methods


Xv: total amount of overlapping vertical Houghlines at each x position.


Heren Kapsalon
SAFARI


Heren Kapsalon
SAFARI


Xv: total amount of overlapping vertical Houghlines at each x position.
D: Absolute of derivative of Xh


Xv: total amount of overlapping vertical Houghlines at each x position.
D: Absolute of derivative of Xh

Window classification

- alignment lines -¿ grid of blocks

Window classification

- alignment lines -¿ grid of blocks
- classify blocks

Window classification

- alignment lines -¿ grid of blocks
- classify blocks
  - window or non-window block

Window classification

- alignment lines -¿ grid of blocks
- classify blocks
  - window or non-window block
- Two methods:

Window classification

- alignment lines -¿ grid of blocks
- classify blocks
  - window or non-window block
- Two methods:
- Basic classification

Window classification

- alignment lines -¿ grid of blocks
- classify blocks
  - window or non-window block
- Two methods:
- Basic classification
  - based on amount of Houghlines

Window classification

- alignment lines -¿ grid of blocks
- classify blocks
  - window or non-window block
- Two methods:
- Basic classification
  - based on amount of Houghlines
- Improved classification

Window classification

- alignment lines -¿ grid of blocks
- classify blocks
  - window or non-window block
- Two methods:
- Basic classification
  - based on amount of Houghlines
- Improved classification
  - based on shape of Histogram function

Basic window classification assumptions

- Assumptions:

Basic window classification assumptions

- Assumptions:
  - a window block contains a high amount of Houghlines

Basic window classification assumptions

- Assumptions:
  - a window block contains a high amount of Houghlines
  - a non-window block contains low amount of Houghlines

Basic window classification assumptions

- Assumptions:
  - a window block contains a high amount of Houghlines
  - a non-window block contains low amount of Houghlines
  - the windows are aligned

Basic window classification assumptions

- Assumptions:
  - a window block contains a high amount of Houghlines
  - a non-window block contains low amount of Houghlines
  - the windows are aligned
    - a row or column contains either zero windows or multiple windows

Basic window classification algorithm

- exploit window alignment: classify full block rows and block columns

Basic window classification algorithm

- exploit window alignment: classify full block rows and block columns
- count the nr of Houghlines in each blockrow

Basic window classification algorithm

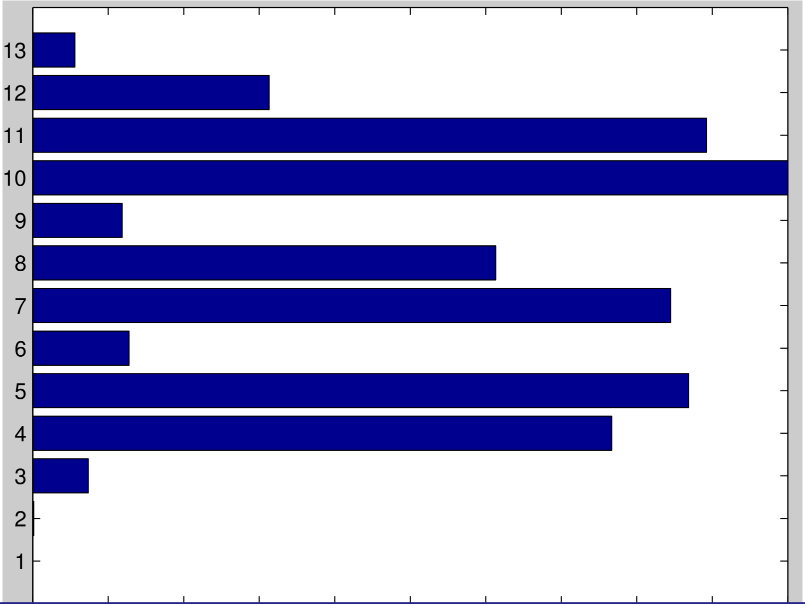
- exploit window alignment: classify full block rows and block columns
- count the nr of Houghlines in each blockrow
- discard blockrow size influence (normalize):

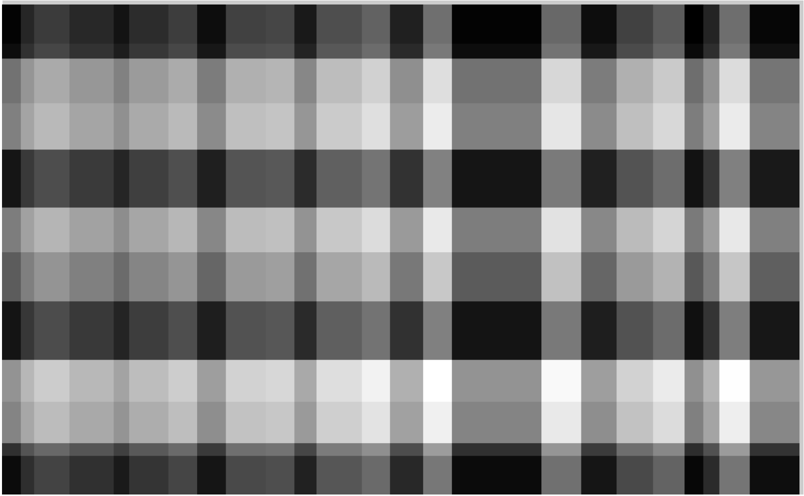
Basic window classification algorithm

- exploit window alignment: classify full block rows and block columns
- count the nr of Houghlines in each blockrow
- discard blockrow size influence (normalize):
- $$\forall Ri \in \{1..numRows\} : R_i = \frac{HoughlinePxCount}{R_i^{width} \cdot R_i^{height}}$$

Basic window classification algorithm

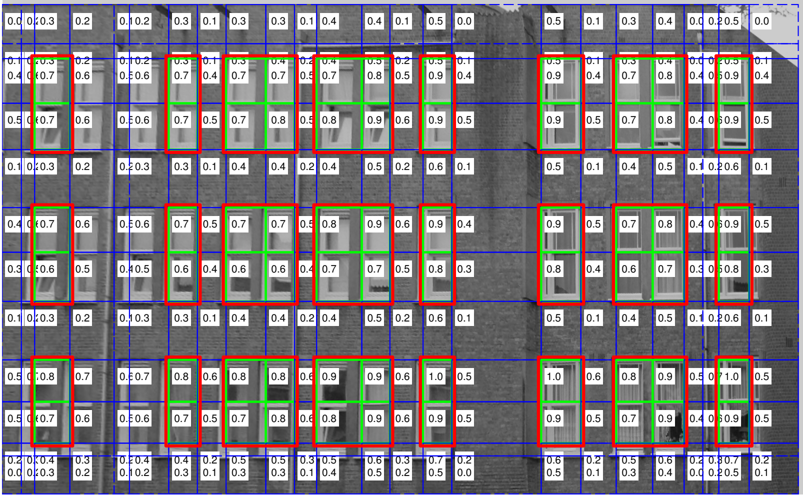
- exploit window alignment: classify full block rows and block columns
- count the nr of Houghlines in each blockrow
- discard blockrow size influence (normalize):
- $$\forall Ri \in \{1..numRows\} : R_i = \frac{HoughlinePxCount}{R_i^{width} \cdot R_i^{height}}$$
- output: ——R—— scalar values that ranks Rows being window area or not


13
12
11
10
9
8
7
6
5
4
3
2
1



From certainty to classification

- *k*-means classification



Improved window classification

- amount of Houghlines

Improved window classification

- amount of Houghlines
  - increases towards center of window area

Improved window classification

- amount of Houghlines
  - increases towards center of window area
  - decreases towards center of non-window area

Improved window classification

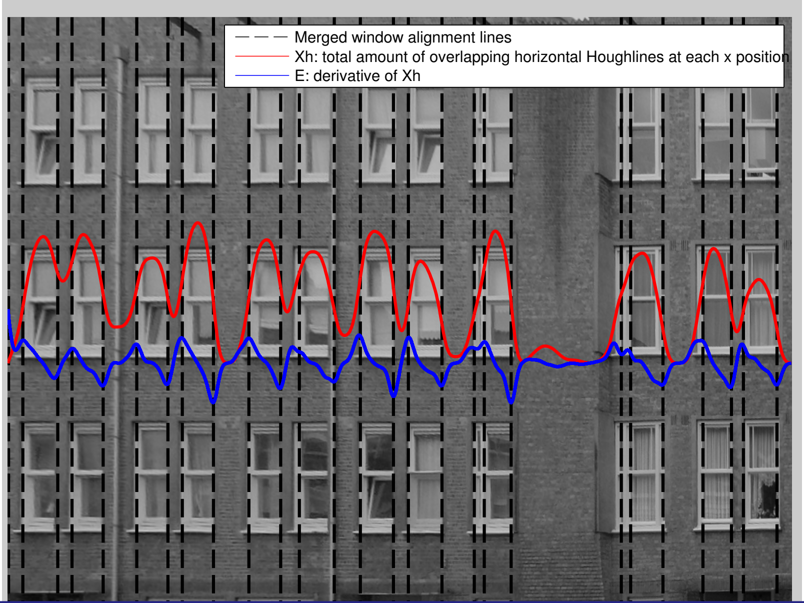
- amount of Houghlines
  - increases towards center of window area
  - decreases towards center of non-window area
- amount of Houghlines increases towards center of window

Improved window classification

- amount of Houghlines
  - increases towards center of window area
  - decreases towards center of non-window area
- amount of Houghlines increases towards center of window
- shape concave -¿ window area

Improved window classification

- amount of Houghlines
  - increases towards center of window area
  - decreases towards center of non-window area
- amount of Houghlines increases towards center of window
- shape concave -¿ window area
- shape convex -¿ non-window area


Merged window alignment lines
Xh: total amount of overlapping horizontal Houghlines at each x position
E: derivative of Xh

Algorithm

- detect concave and convex shape of histogram function Xh

Algorithm

- detect concave and convex shape of histogram function Xh
- X'_h

Algorithm

- detect concave and convex shape of histogram function Xh
- X_h'
- sign change of derivative indicates peak or valley

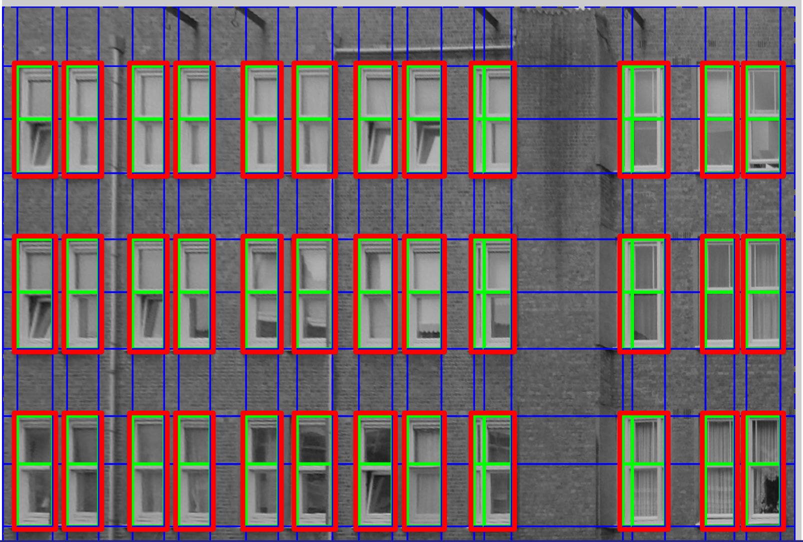
Algorithm

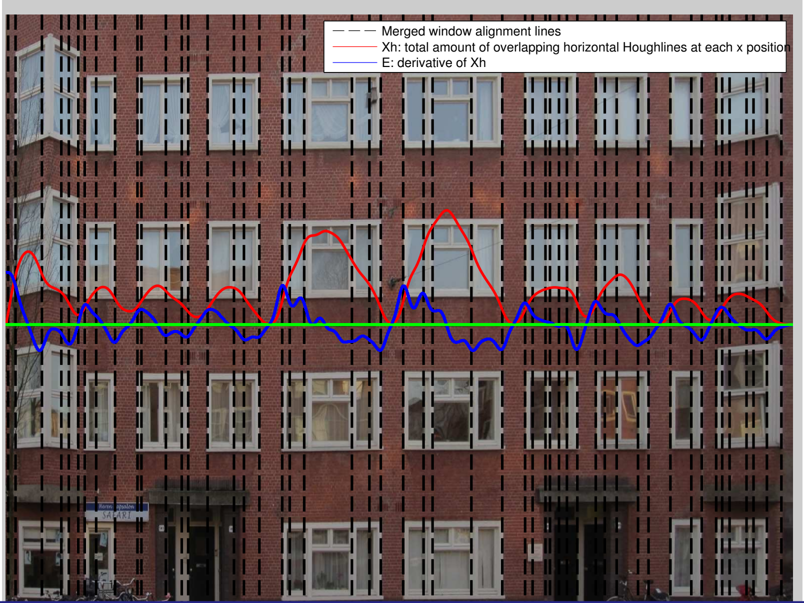
- detect concave and convex shape of histogram function Xh
- X_h'
- sign change of derivative indicates peak or valley
- (+,-) -¿ concave -¿ window area

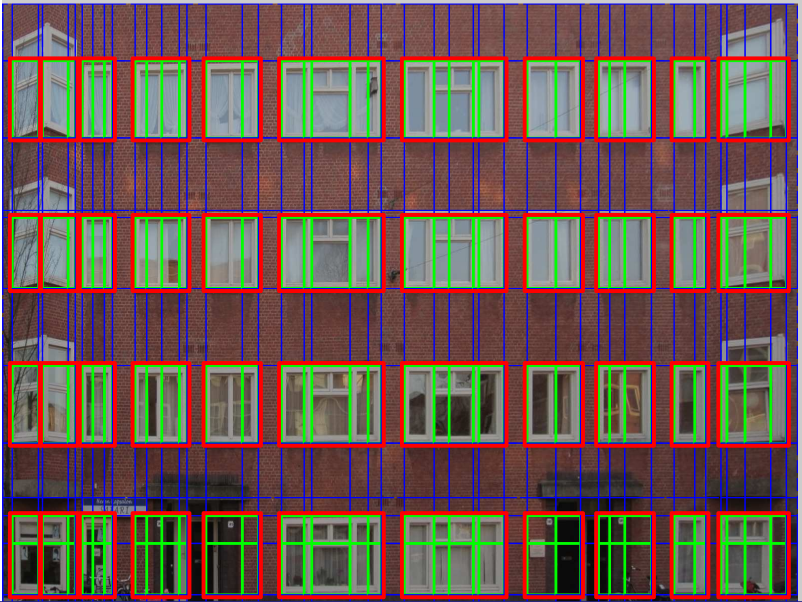
Algorithm

- detect concave and convex shape of histogram function Xh
- X_h'
- sign change of derivative indicates peak or valley
- (+,-) -¿ concave -¿ window area
- (-,+) -¿ convex -¿ non-window area

Result




Merged window alignment lines
Xh: total amount of overlapping horizontal Houghlines at each x position
E: derivative of Xh



Conclusions

- Connected corner

Conclusions

- Connected corner
  - detection rate 97%

Conclusions

- Connected corner
  - detection rate 97%
  - suitable for scenes with variation in window size/type

Conclusions

- Connected corner
  - detection rate 97%
  - suitable for scenes with variation in window size/type
  - small requirement on input data

Conclusions

- Connected corner
  - detection rate 97%
  - suitable for scenes with variation in window size/type
  - small requirement on input data
    - neither 3D information about building nor rectification is needed

Conclusions

- Connected corner
  - detection rate 97%
  - suitable for scenes with variation in window size/type
  - small requirement on input data
    - neither 3D information about building nor rectification is needed
  - future research

Conclusions

- Connected corner
  - detection rate 97%
  - suitable for scenes with variation in window size/type
  - small requirement on input data
    - neither 3D information about building nor rectification is needed
  - future research
    - U shapes

Conclusions

- Connected corner
  - detection rate 97%
  - suitable for scenes with variation in window size/type
  - small requirement on input data
    - neither 3D information about building nor rectification is needed
  - future research
    - U shapes
    - analysis of subwindow structure

Conclusion:Connected corner

- detection rate 97%

Conclusion:Connected corner

- detection rate 97%
- suitable for scenes with variation in window size/type

Conclusion:Connected corner

- detection rate 97%
- suitable for scenes with variation in window size/type
- small requirement on input data

Conclusion:Connected corner

- detection rate 97%
- suitable for scenes with variation in window size/type
- small requirement on input data
- neither 3D information about building nor rectification is needed

Conclusion:Connected corner

- detection rate 97%
- suitable for scenes with variation in window size/type
- small requirement on input data
- neither 3D information about building nor rectification is needed
- future research

Conclusion:Connected corner

- detection rate 97%
- suitable for scenes with variation in window size/type
- small requirement on input data
- neither 3D information about building nor rectification is needed
- future research
  - U shapes

Conclusion:Connected corner

- detection rate 97%
- suitable for scenes with variation in window size/type
- small requirement on input data
- neither 3D information about building nor rectification is needed
- future research
  - U shapes
  - analysis of subwindow structure

Conclusion:Histogram based window alignment

- interpreting amount of Houghlines strong approach towards window detection

Conclusion:Histogram based window alignment

- interpreting amount of Houghlines strong approach towards window detection
- two window detection methods

Conclusion:Histogram based window alignment

- interpreting amount of Houghlines strong approach towards window detection
- two window detection methods
- alternative window alignment performs better

Conclusion:Histogram based window alignment

- interpreting amount of Houghlines strong approach towards window detection
- two window detection methods
- alternative window alignment performs better
  - strong combination horizontal and vertical Houghlines histograms

Conclusion:Histogram based window alignment

- interpreting amount of Houghlines strong approach towards window detection
- two window detection methods
- alternative window alignment performs better
  - strong combination horizontal and vertical Houghlines histograms
  - high order shape interpretation (derivative) of Histogram function

Conclusion:window classification

- method I: based on amount of Houghlines

Conclusion:window classification

- method I: based on amount of Houghlines
  - performs quite good on the dataset we used

Conclusion:window classification

- method I: based on amount of Houghlines
  - performs quite good on the dataset we used
  - very sensitive to alignment errors

Conclusion:window classification

- method I: based on amount of Houghlines
  - performs quite good on the dataset we used
  - very sensitive to alignment errors
  - (errors in alignment propagate to classifications)

Conclusion:window classification

- method I: based on amount of Houghlines
  - performs quite good on the dataset we used
  - very sensitive to alignment errors
  - (errors in alignment propagate to classifications)
- method II:based on shape of Houghlines Histogram

Conclusion:window classification

- method I: based on amount of Houghlines
  - performs quite good on the dataset we used
  - very sensitive to alignment errors
  - (errors in alignment propagate to classifications)
- method II:based on shape of Houghlines Histogram
  - strong increase/decrease of Houghlines

Conclusion:window classification

- method I: based on amount of Houghlines
  - performs quite good on the dataset we used
  - very sensitive to alignment errors
  - (errors in alignment propagate to classifications)
- method II:based on shape of Houghlines Histogram
  - strong increase/decrease of Houghlines
  - very robust, at least 97% detection rate on all datasets, why?

Conclusion:window classification

- method I: based on amount of Houghlines
  - performs quite good on the dataset we used
  - very sensitive to alignment errors
  - (errors in alignment propagate to classifications)
- method II:based on shape of Houghlines Histogram
  - strong increase/decrease of Houghlines
  - very robust, at least 97% detection rate on all datasets, why?
  - higher order interpretation of Histogram function

Conclusion:window classification

- method I: based on amount of Houghlines
  - performs quite good on the dataset we used
  - very sensitive to alignment errors
  - (errors in alignment propagate to classifications)
- method II:based on shape of Houghlines Histogram
  - strong increase/decrease of Houghlines
  - very robust, at least 97% detection rate on all datasets, why?
  - higher order interpretation of Histogram function
  - robust to alignment errors

Conclusion:window classification

- method I: based on amount of Houghlines
  - performs quite good on the dataset we used
  - very sensitive to alignment errors
  - (errors in alignment propagate to classifications)
- method II:based on shape of Houghlines Histogram
  - strong increase/decrease of Houghlines
  - very robust, at least 97% detection rate on all datasets, why?
  - higher order interpretation of Histogram function
  - robust to alignment errors
  - the increase/decrease is relative

Conclusion:window classification

- method I: based on amount of Houghlines
  - performs quite good on the dataset we used
  - very sensitive to alignment errors
  - (errors in alignment propagate to classifications)
- method II:based on shape of Houghlines Histogram
  - strong increase/decrease of Houghlines
  - very robust, at least 97% detection rate on all datasets, why?
  - higher order interpretation of Histogram function
  - robust to alignment errors
  - the increase/decrease is relative
  - robust to variation in window type and illumination conditions

Conclusion

- We retrieved semantics of urban scenes

Conclusion

- We retrieved semantics of urban scenes
  - a full 3D reconstruction of a building

Conclusion

- We retrieved semantics of urban scenes
  - a full 3D reconstruction of a building
  - extraction of windows

Questions

Questions?