

SEMANTIC ANNOTATION OF URBAN SCENES: SKYLINE AND WINDOW DETECTION

Speaker: Tjerk Kostelijk

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

July 6, 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 light

Did you see?

- ▶ building
- ▶ tree
- ▶ bicycle
- ▶ street light
- ▶ blue car

Did you see?

- ▶ building
- ▶ tree
- ▶ bicycle
- ▶ street light
- ▶ blue car
- ▶ red car

Did you see?

- ▶ building
- ▶ tree
- ▶ bicycle
- ▶ street light
- ▶ 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*

Outline

Introduction

Skylinedetection

Extracting the 3D model

Window detection

Outline

Introduction

Skylinedetection

Extracting the 3D model

Window detection

What is my research about?

- ▶ Semantic annotation of urban scenes

What is my research about?

- ▶ Semantic annotation of urban scenes
- ▶ Describe the scene on a higher level

What is my research about?

- ▶ Semantic annotation of urban scenes
- ▶ Describe the scene on a higher level
- ▶ Based on 2D input images of different views

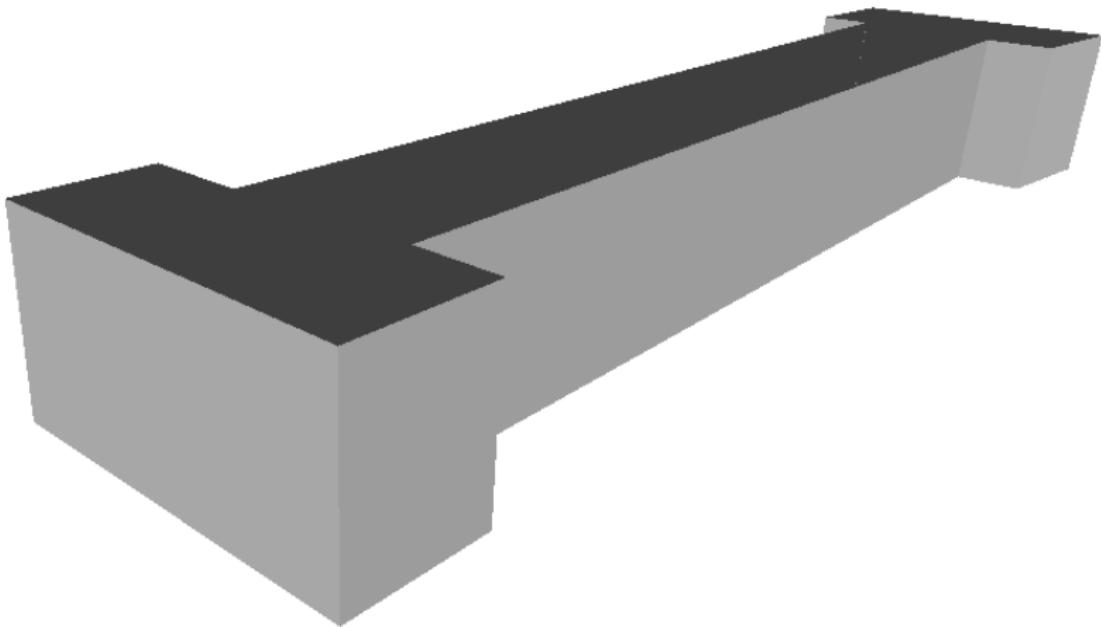
What is my research about?

- ▶ Semantic annotation of urban scenes
- ▶ Describe the scene on a higher level
- ▶ Based on 2D input images of different views
 - ▶ Skyline detection



Where is my research about?

- ▶ Semantic annotation of urban scenes
- ▶ Describe the scene on a higher level
- ▶ Based on 2D input images of different views
 - ▶ Skyline detection
 - ▶ **3D building reconstruction**

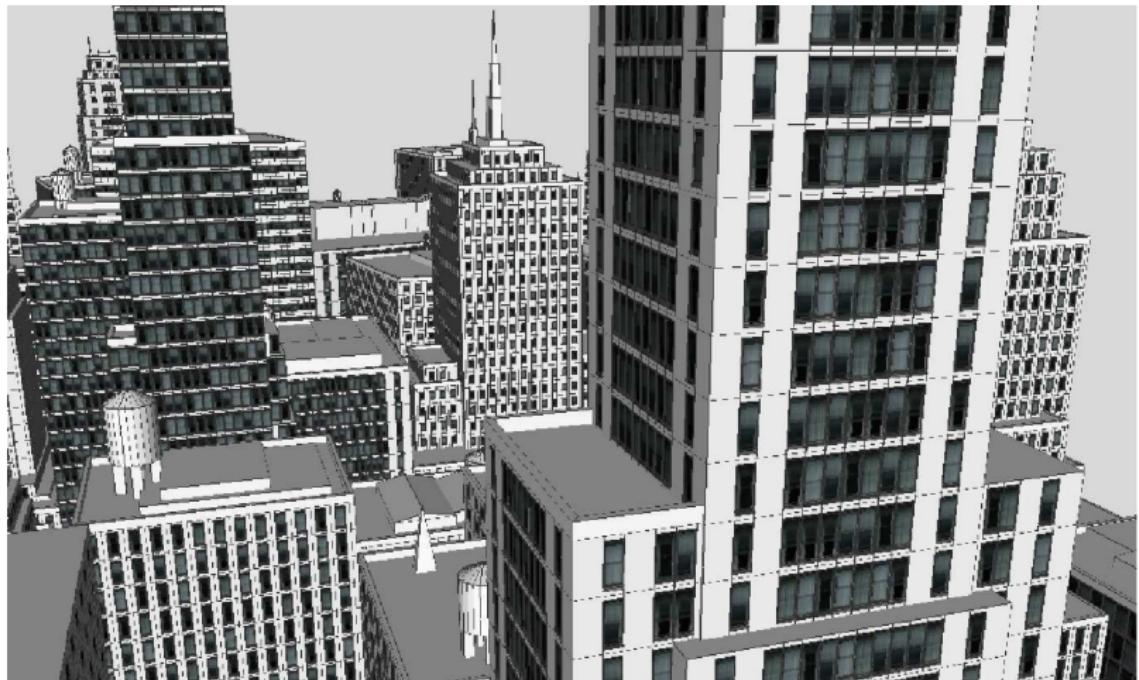


Where is my research about?

- ▶ Semantic annotation of urban scenes
- ▶ Describe the scene on a higher level
- ▶ Based on 2D input images of different views
 - ▶ Skyline detection
 - ▶ 3D building reconstruction
 - ▶ **Window detection**

Application examples of semantic annotation of urban scenes

- ▶ 3D city models









Application examples of annotation of urban scenes

- ▶ 3D city models
- ▶ Driving simulation

Application examples of annotation of urban scenes

- ▶ 3D city models
- ▶ Driving simulation
- ▶ **Building recognition**

Application examples of annotation of urban scenes

- ▶ 3D city models
- ▶ Driving simulation
- ▶ Building recognition
- ▶ Augmented reality





Application examples of annotation of urban scenes

- ▶ 3D city models
- ▶ Driving simulation
- ▶ Building recognition
- ▶ Augmented reality
- ▶ Analysis building deformation

Application examples of annotation of urban scenes

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

Outline

| Introduction

Skylinedetection

Extracting the 3D model

Window detection

Skyline detection application example

- ▶ Horizon detection for Unmanned Air Vehicles



Skyline detection in urban scenes

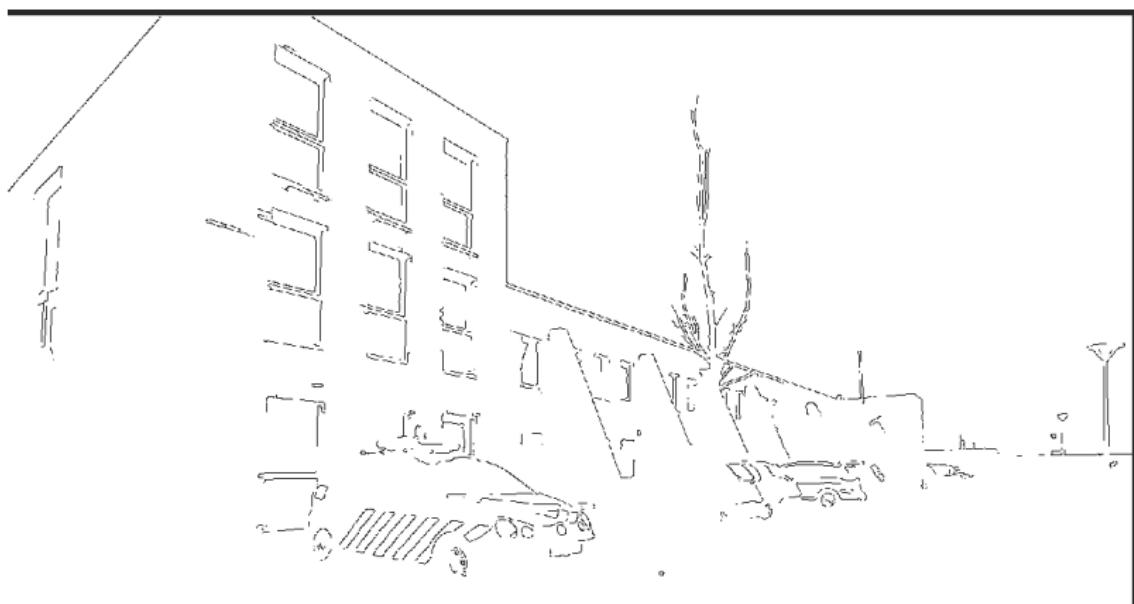


Skyline detection in urban scenes

- ▶ Canny edge detection (based on intensity change)

Skyline detection in urban scenes

- ▶ Canny edge detection (based on intensity change)
- ▶ Result: binary image (edge or no edge)



Skyline detection algorithm

- ▶ Which edge represents the building contour?

Skyline detection algorithm

- ▶ Which edge represents the building contour?
- ▶ Top sharp edge assumption
"The first sharp edge (seen from top to bottom) in the image represents the skyline."

Skyline detection algorithm

- ▶ Which edge represents the building contour?
- ▶ Top sharp edge assumption
 - "The first sharp edge (seen from top to bottom) in the image represents the skyline."*
- ▶ Algorithm:

Skyline detection algorithm

- ▶ Which edge represents the building contour?
- ▶ 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

- ▶ Which edge represents the building contour?
- ▶ 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 pixel columns

Skyline detection algorithm

- ▶ Which edge represents the building contour?
- ▶ 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 $\#w$ pixel columns
 - ▶ Each column present $\#h$ binary edge values (edge or no edge)

Skyline detection algorithm

- ▶ Which edge represents the building contour?
- ▶ 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 $\#w$ pixel columns
 - ▶ Each column present $\#h$ binary edge values (edge or no edge)
 - ▶ **y-location of the most upper edge value is stored**

Skyline detection result



Skyline detection result

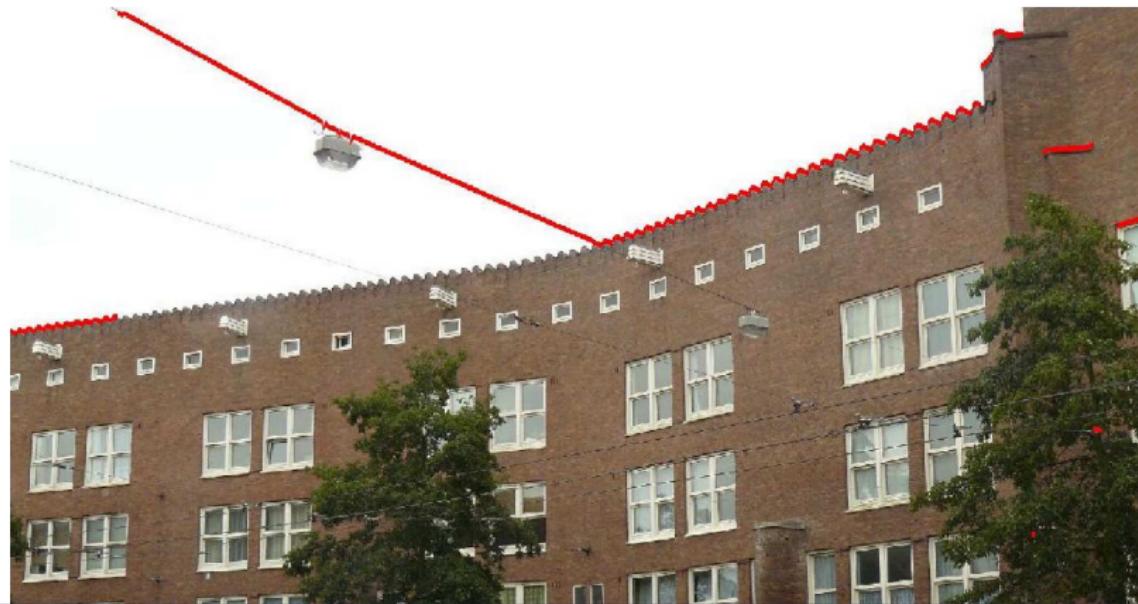


Future research

Hypothesis based skyline detection, assumption

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

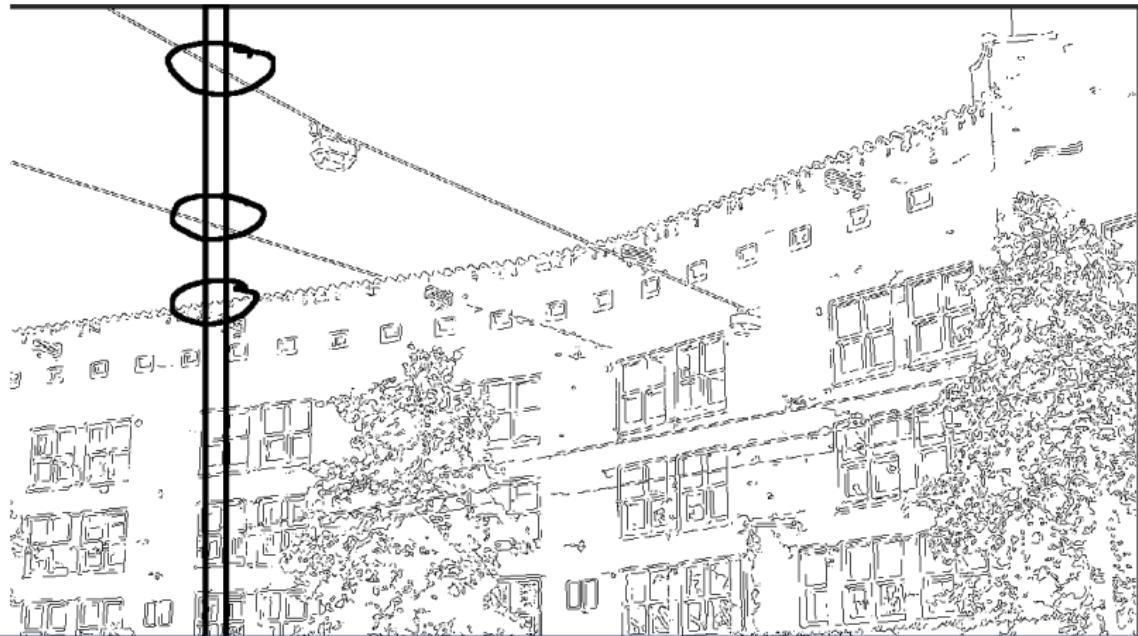
- ▶ Example of a scene where this assumption is violated



Hypothesis based skyline detection, assumption

- ▶ Change of assumption

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

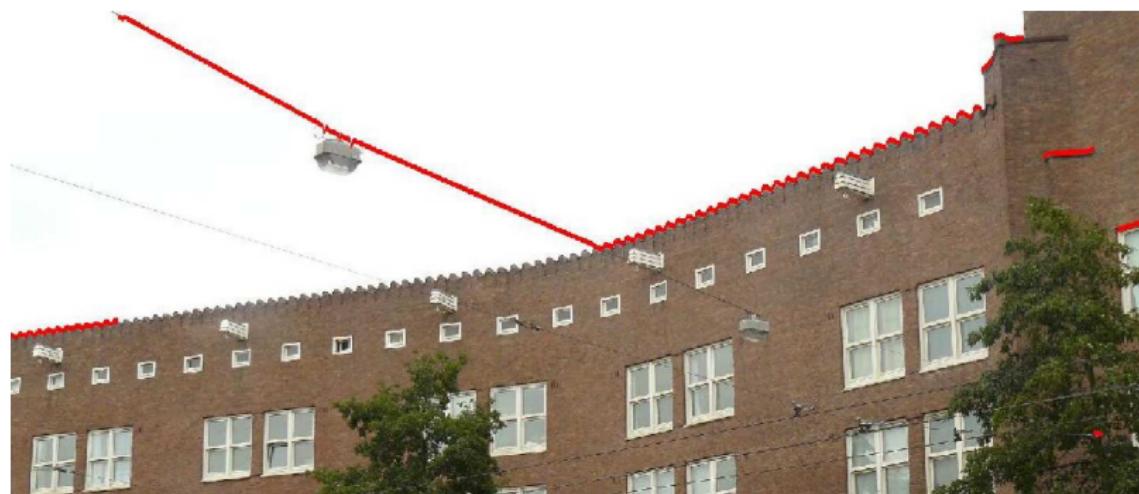


Hypothesis based skyline detection, algorithm

- ▶ Generate n hypothesis

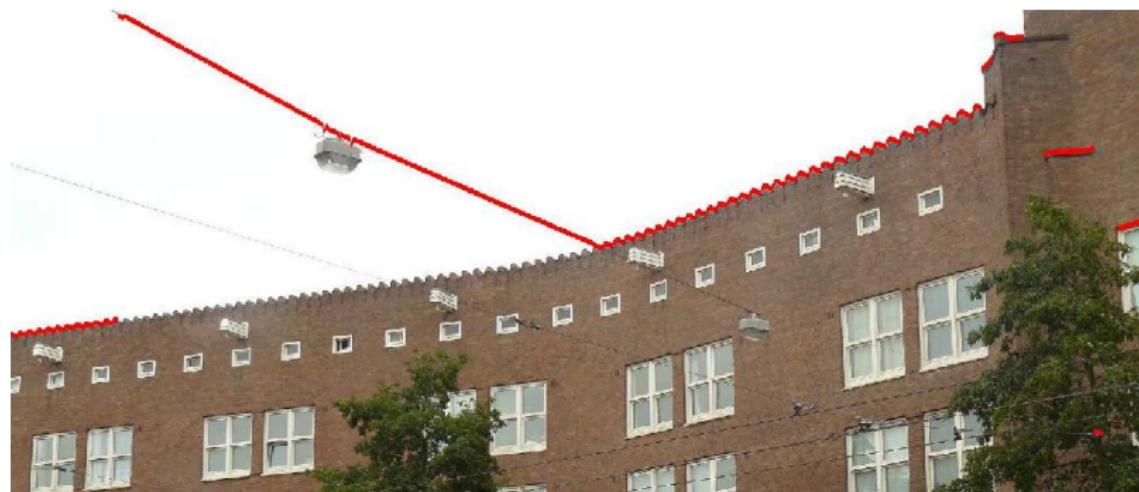
Hypothesis based skyline detection, algorithm

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



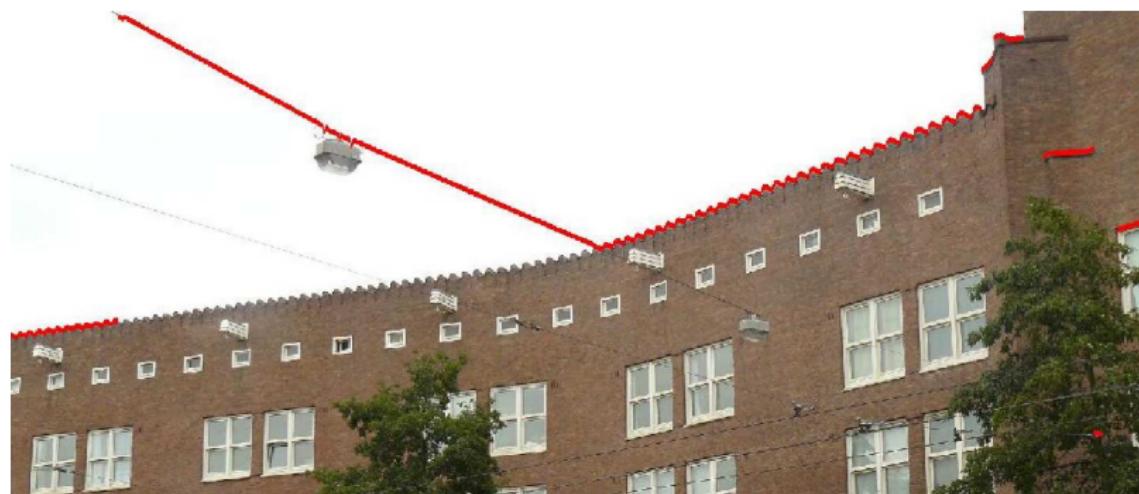
Hypothesis based skyline detection, algorithm

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



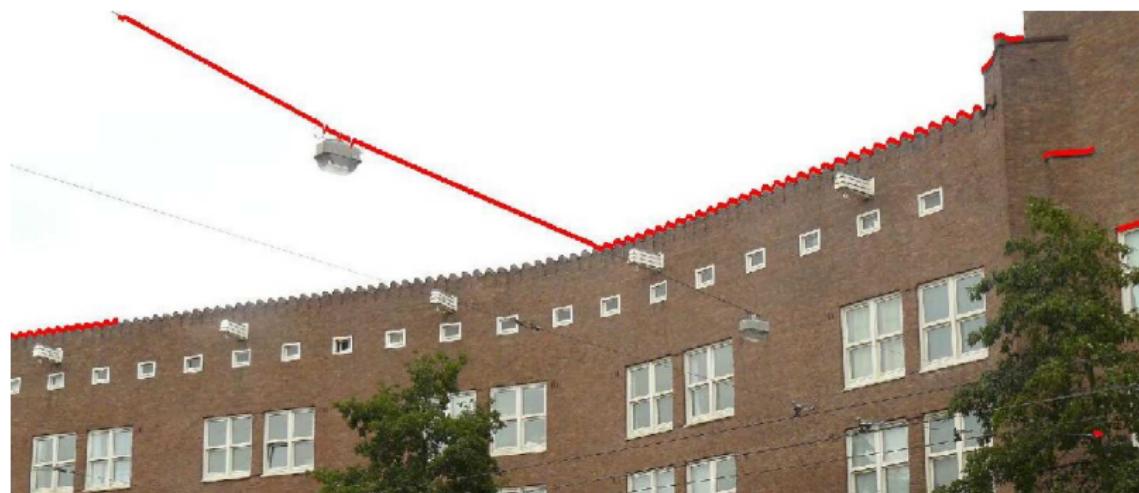
Hypothesis based skyline detection, algorithm

- ▶ 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



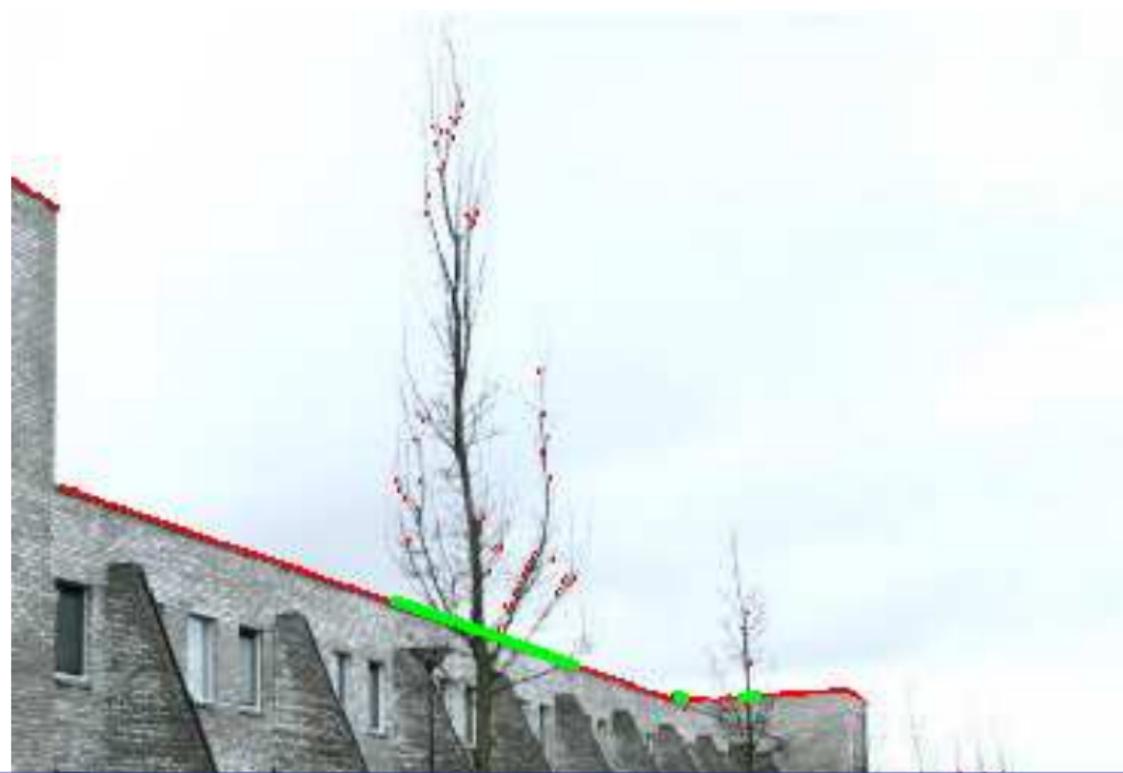
Expected result on hypothesis classification based on color



Expected result classification based on height variation



Expected result classification based on height variation



Conclusion

- ▶ The skyline detection algorithm

Conclusion

- ▶ The skyline detection algorithm
 - ▶ is simple and has a low complexity

Conclusion

- ▶ The skyline detection algorithm
 - ▶ is simple and has a low complexity
 - ▶ works well under the assumption skyline is upper edge

Conclusion

- ▶ The skyline detection algorithm
 - ▶ is simple and has a low complexity
 - ▶ works well under the assumption skyline is upper edge
 - ▶ **needs future research**

Conclusion

- ▶ The skyline detection algorithm
 - ▶ is simple and has a low complexity
 - ▶ works well 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

Introduction

Skylinedetection

Extracting the 3D model

Window detection

Overview

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

Overview

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

Overview

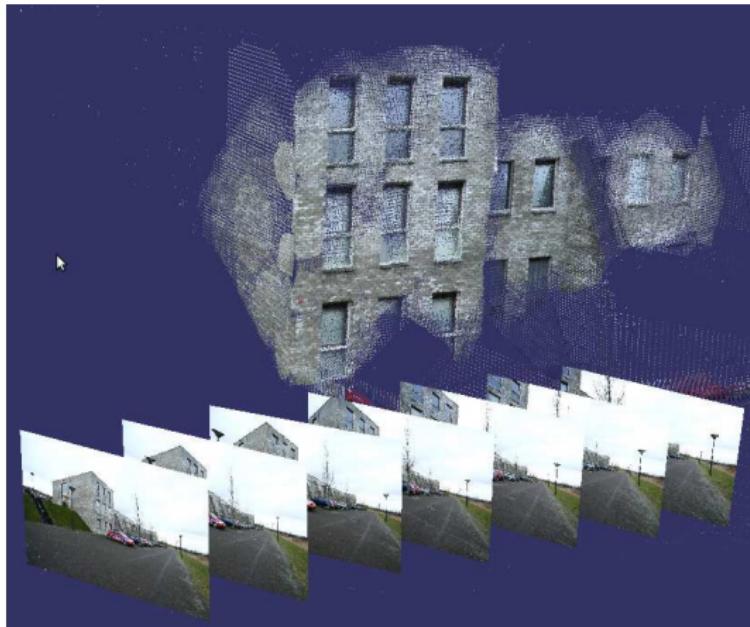
- ▶ Create (top view) 2D model of the scene using *Openstreetmap*
- ▶ Align 2D model with the scene
- ▶ Transform the 2D model to a 3D model by extending the walls

Extract 2D model



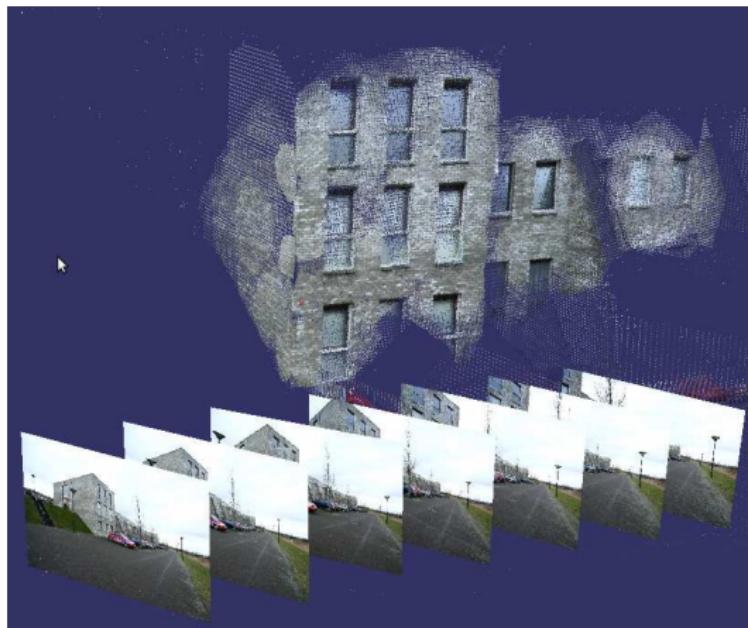
Align 2D model

- Isaac Esteban's *FIT3D toolbox*



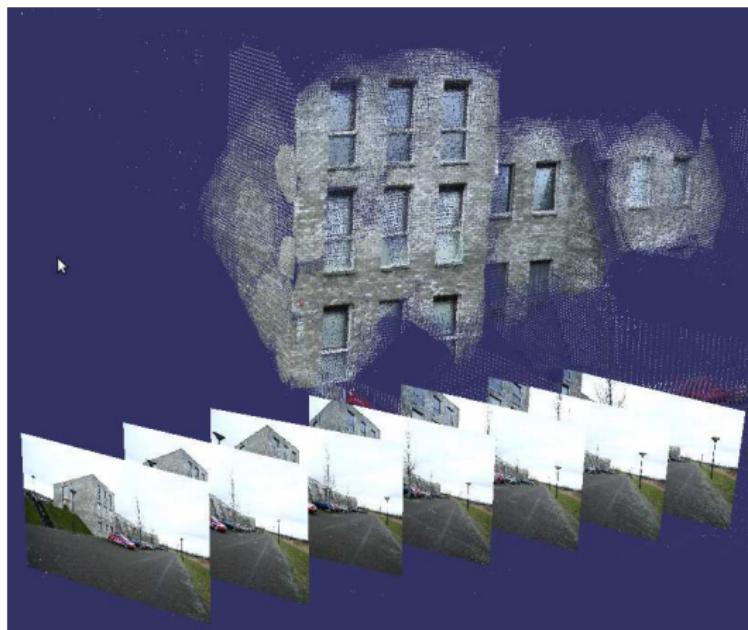
Align 2D model

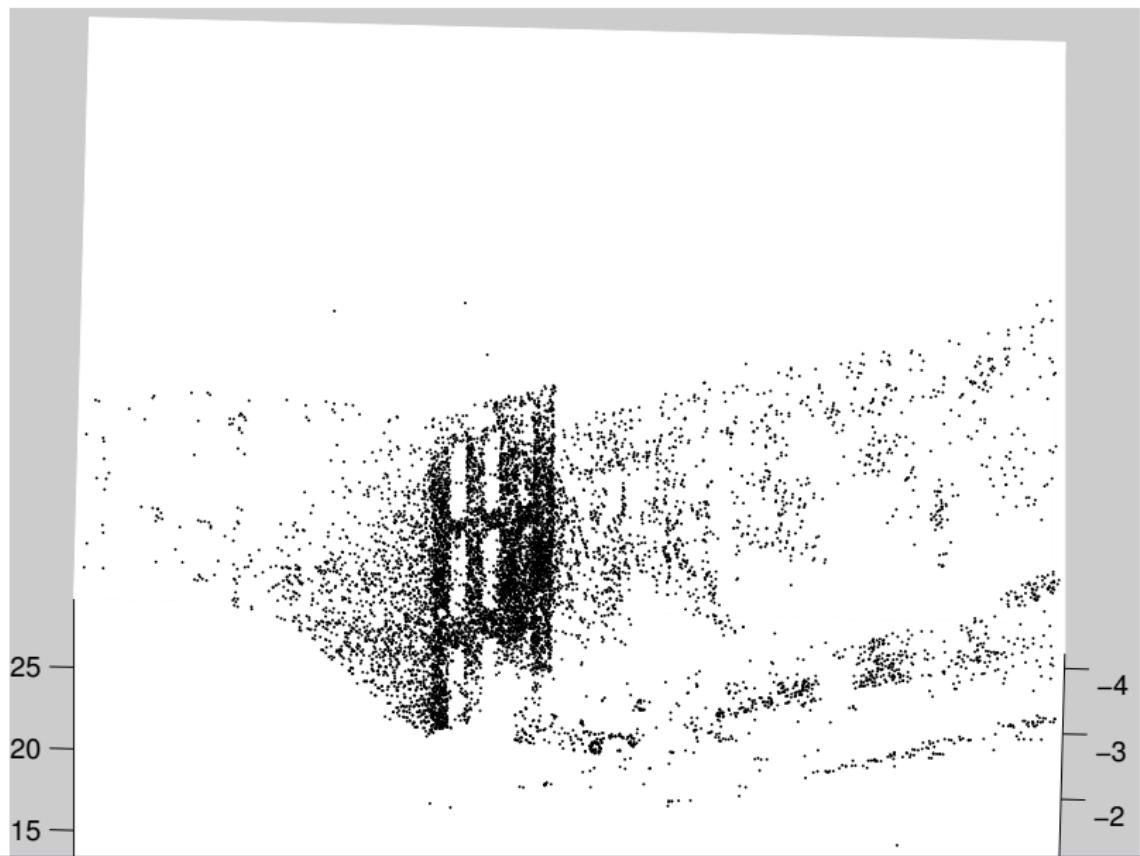
- ▶ Isaac Esteban's *FIT3D toolbox*
 - ▶ Input: sequence of images (different views of the building)



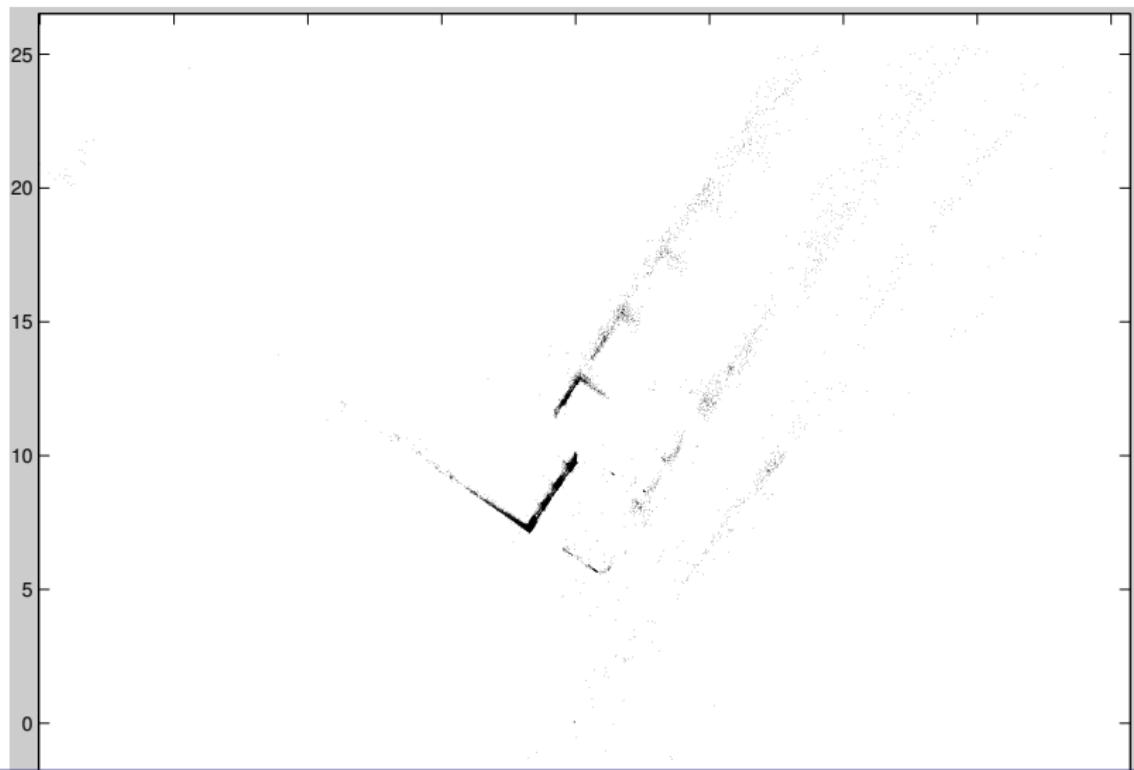
Align 2D model

- ▶ Isaac Esteban's *FIT3D toolbox*
 - ▶ Input: sequence of images (different views of the building)
 - ▶ Output: a 3D point cloud of the building

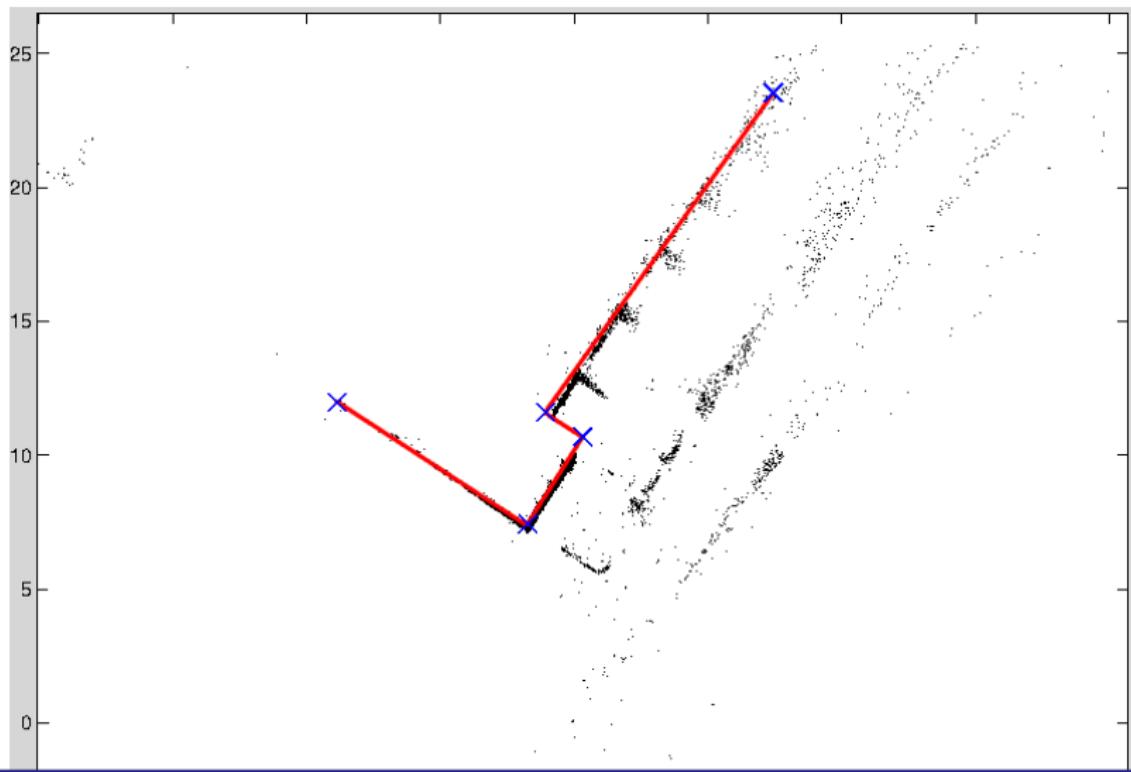




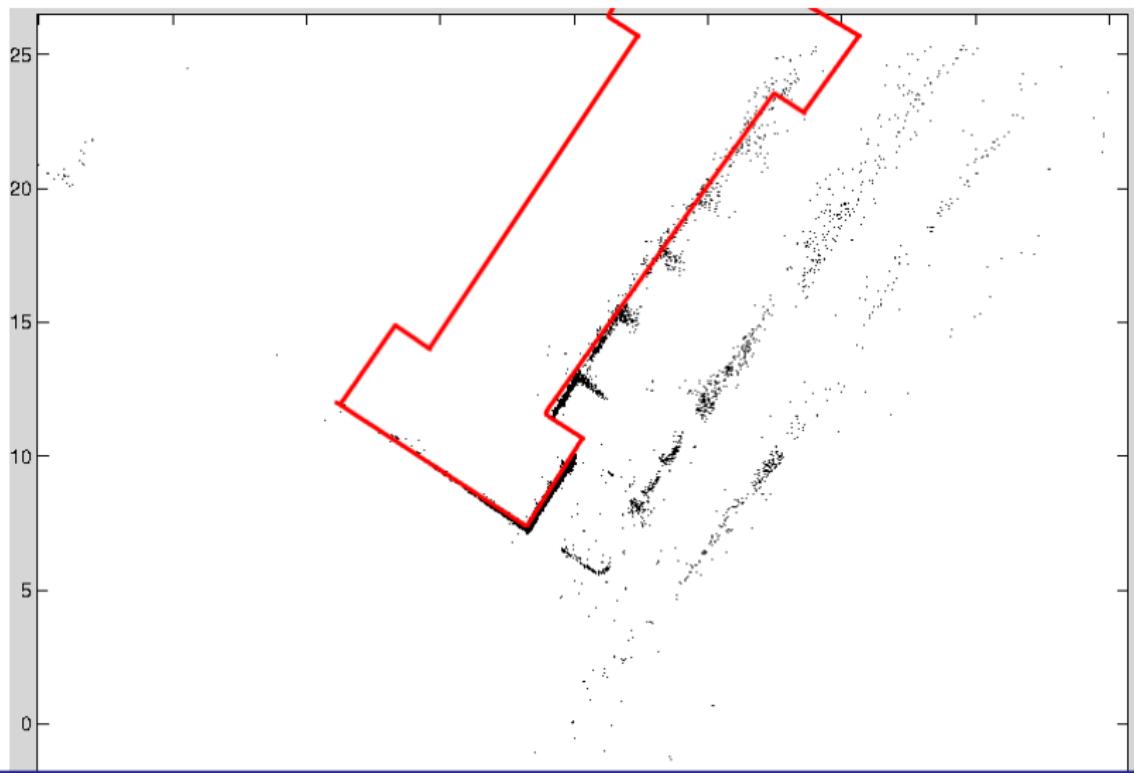
Align 2D model



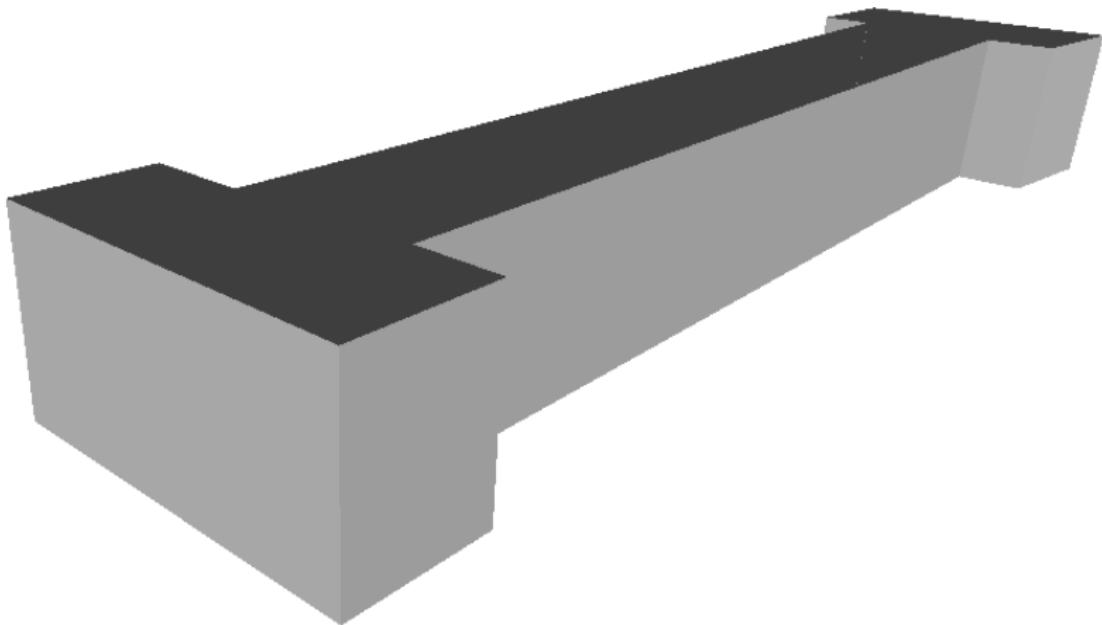
Align 2D model



Align 2D model



Results

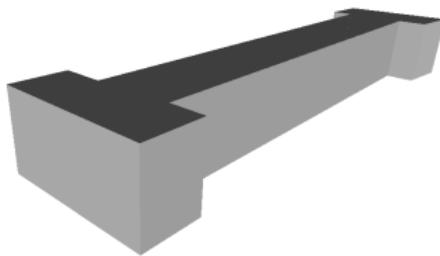


Wall height estimation

- ▶ Wall heights of 3D model are not accurate

Wall height estimation

- ▶ Wall heights of 3D model are not accurate
- ▶ Improve the 3D model by wall height estimation



(c)



(d)

Skyline detection for wall height estimation

- ▶ Straight lines assumption:

Skyline detection for wall height estimation

- ▶ Straight lines assumption:
 - ▶ *Straight lines in the skyline are likely to come from the building contour*

Skyline detection for wall height estimation

- ▶ Straight lines assumption:
 - ▶ *Straight lines in the skyline are likely to come from the building contour*
- ▶ Flat roof assumption:

Skyline detection for wall height estimation

- ▶ Straight lines assumption:
 - ▶ *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

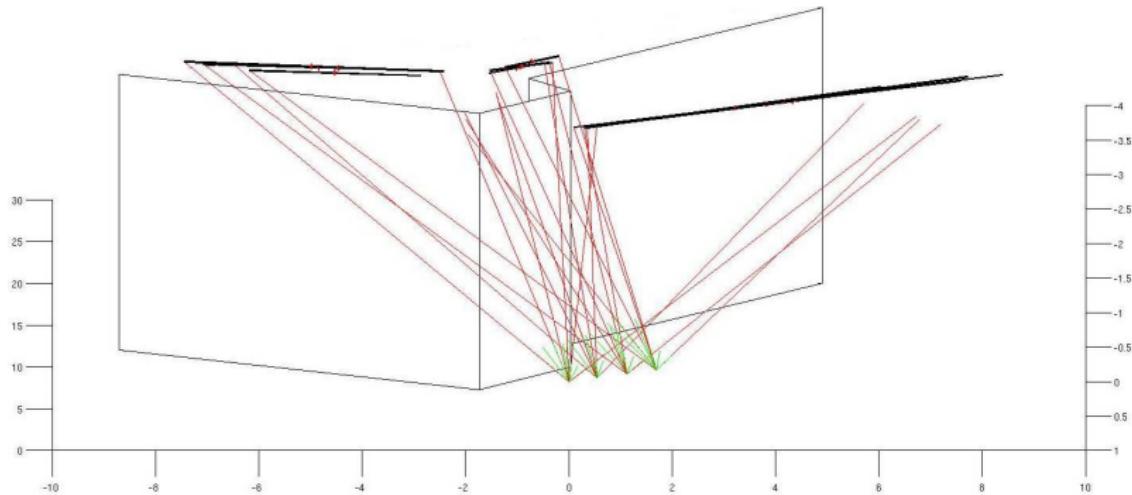


Project to the 3D model

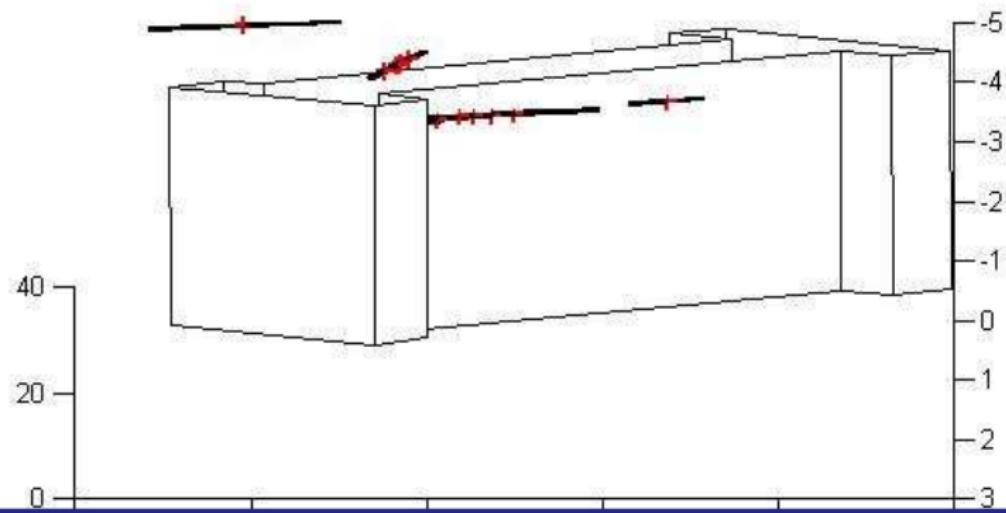
- ▶ Estimate wall heights

Project to the 3D model

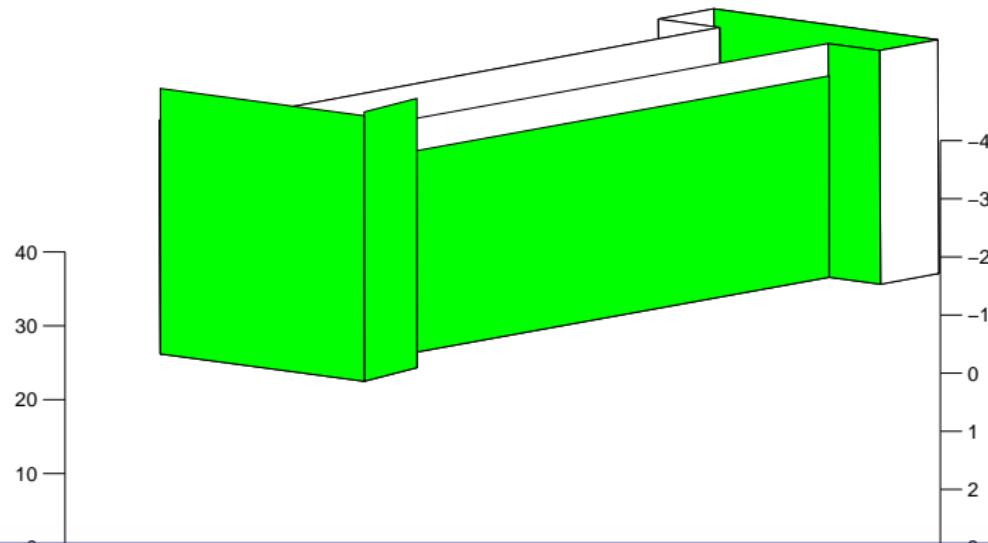
- ▶ Estimate wall heights
- ▶ Project line segments to specific walls of 3D model



Results

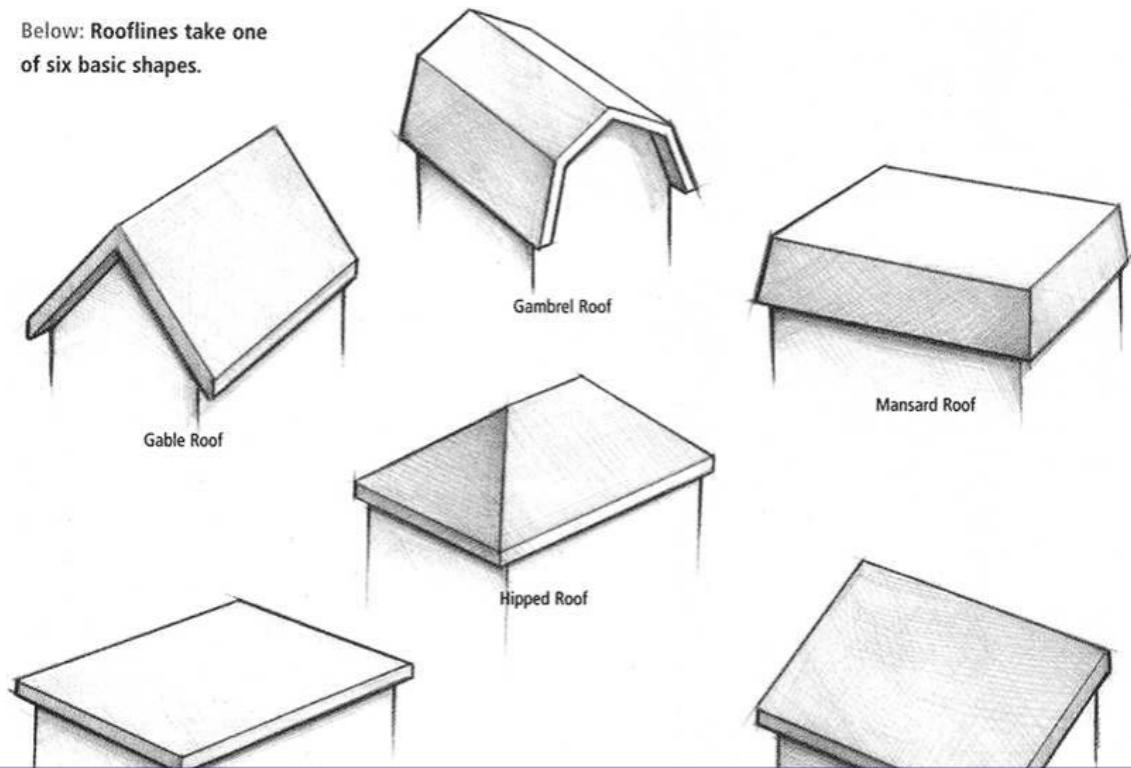


Results

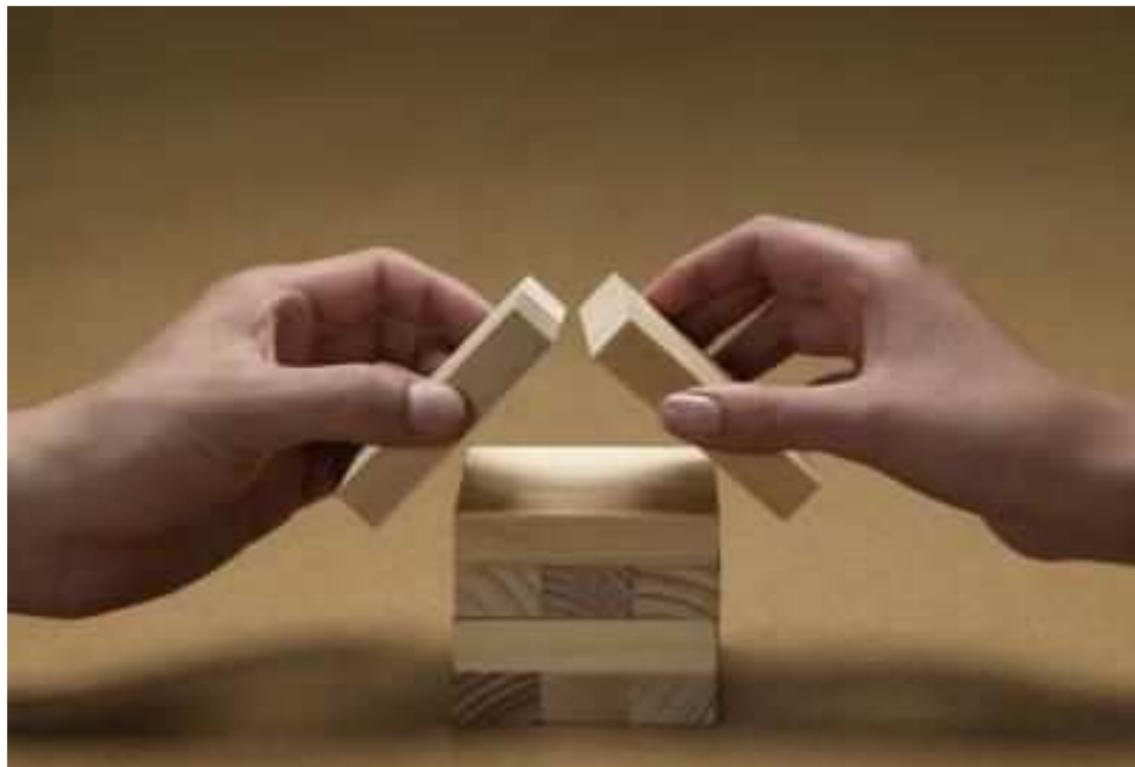


Future research

Below: Rooflines take one of six basic shapes.



Future research



Outline

Introduction

Skylinedetection

Extracting the 3D model

Window detection



(e)



(f)

Outline window detection

- ▶ Method I: Connected corner approach
 - ▶ invariant to viewing direction
- ▶ Facade rectification
- ▶ Method II: Histogram based approach
 - ▶ frontal view only
 - ▶ uses histograms of Houghlines

Method I: Connected corner approach

Situation

- ▶ Window = frame + glass (+ sub windows)

Situation

- ▶ Window = frame + glass (+ sub windows)
- ▶ Color difference, frame, glass

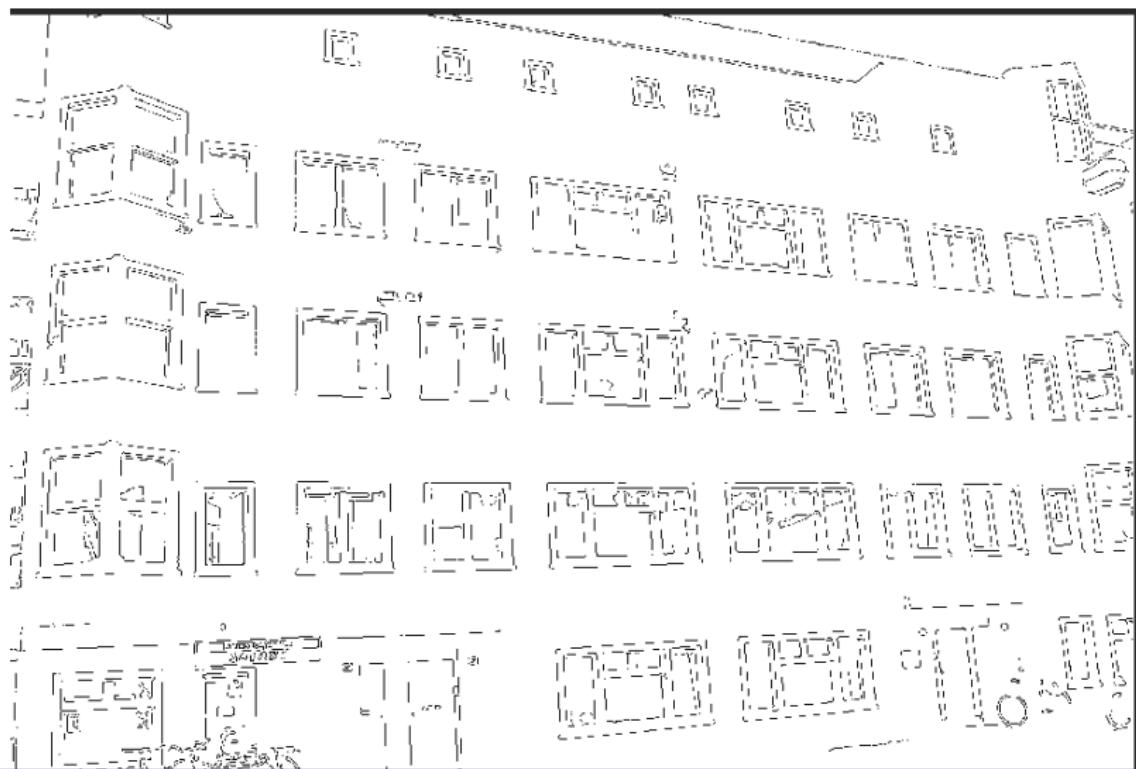
Situation

- ▶ Window = frame + glass (+ sub windows)
- ▶ Color difference, frame, glass
- ▶ Produce edges in horizontal and vertical direction

Original



Edge detection



The idea

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

Edge detection and Houghline extraction

- ▶ Two groups of straight lines (Houghlines)
 - ▶ θ -constraint

Edge detection and Houghline extraction

- ▶ Two groups of straight lines (Houghlines)
 - ▶ θ -constraint
 - ▶ Horizontal, $\theta = [-30..0..30)$ degrees

Edge detection and Houghline extraction

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

Edge detection and Houghline extraction

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

Edge detection and Houghline extraction

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

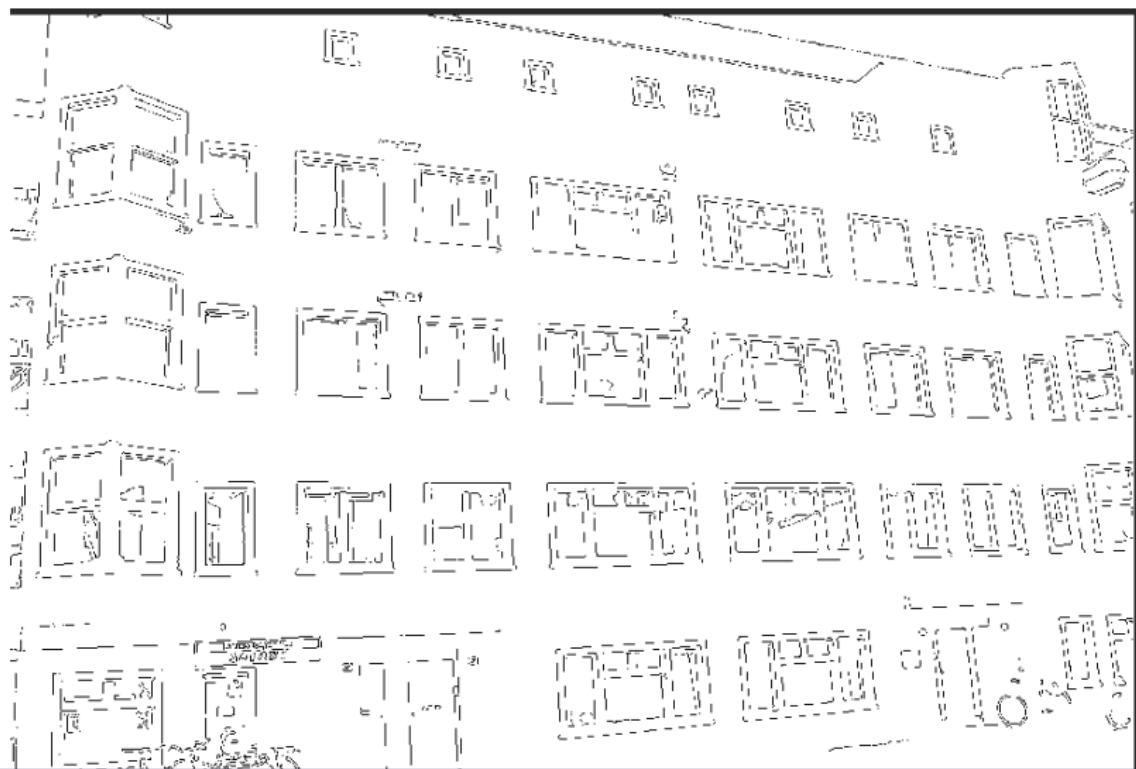
Edge detection and Houghline extraction

- ▶ Two groups of straight lines (Houghlines)
 - ▶ θ -constraint
 - ▶ 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

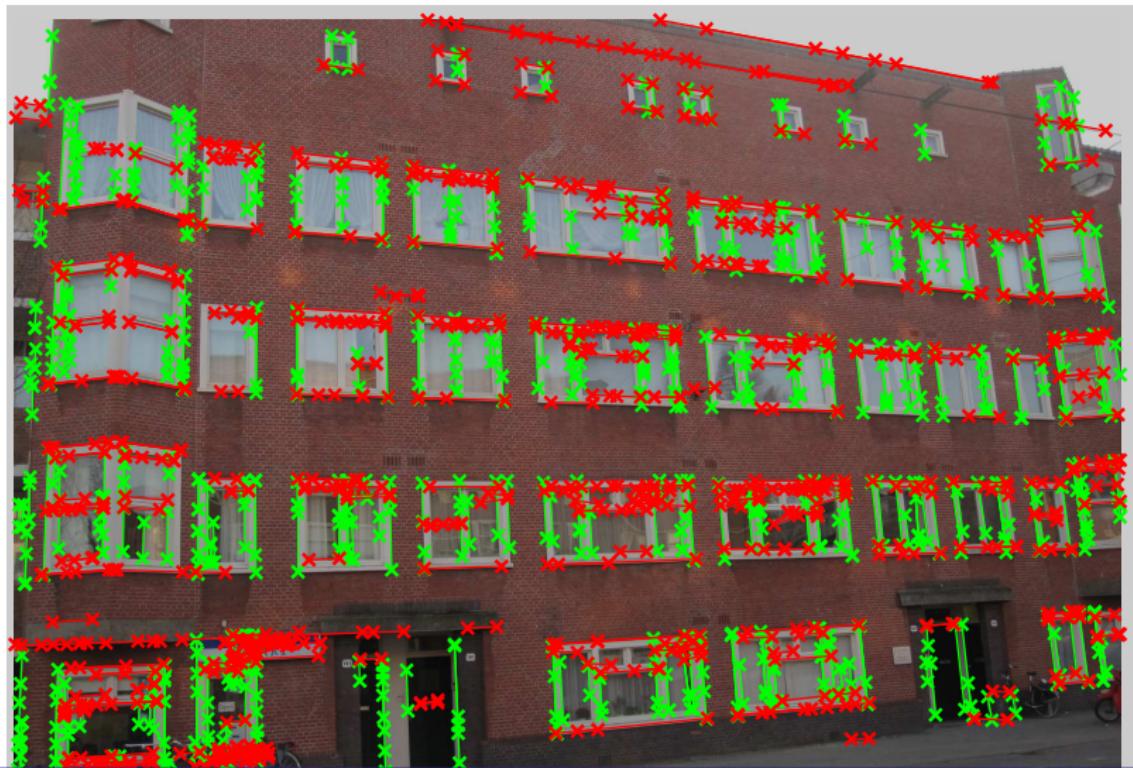
Original



Edge detection



Result of θ constrained Hough transform



Connected corner

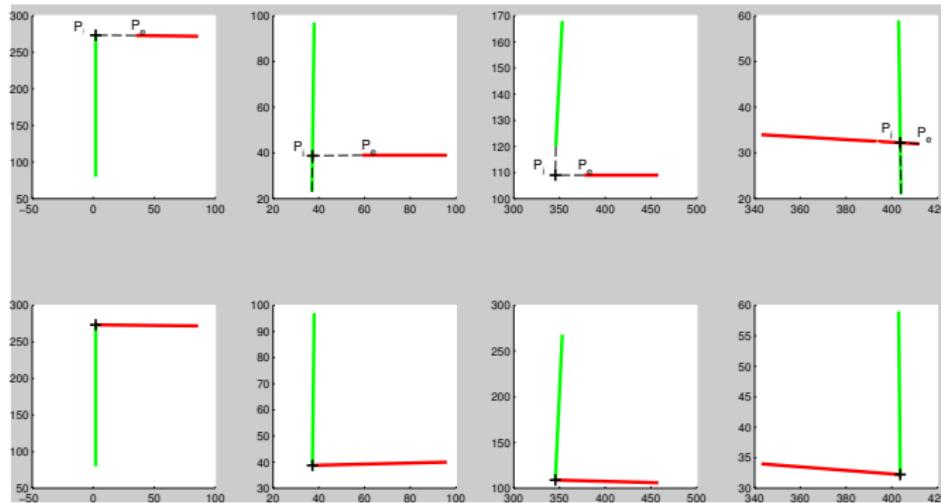
- ▶ horizontal and vertical edge share same corner e.g. L

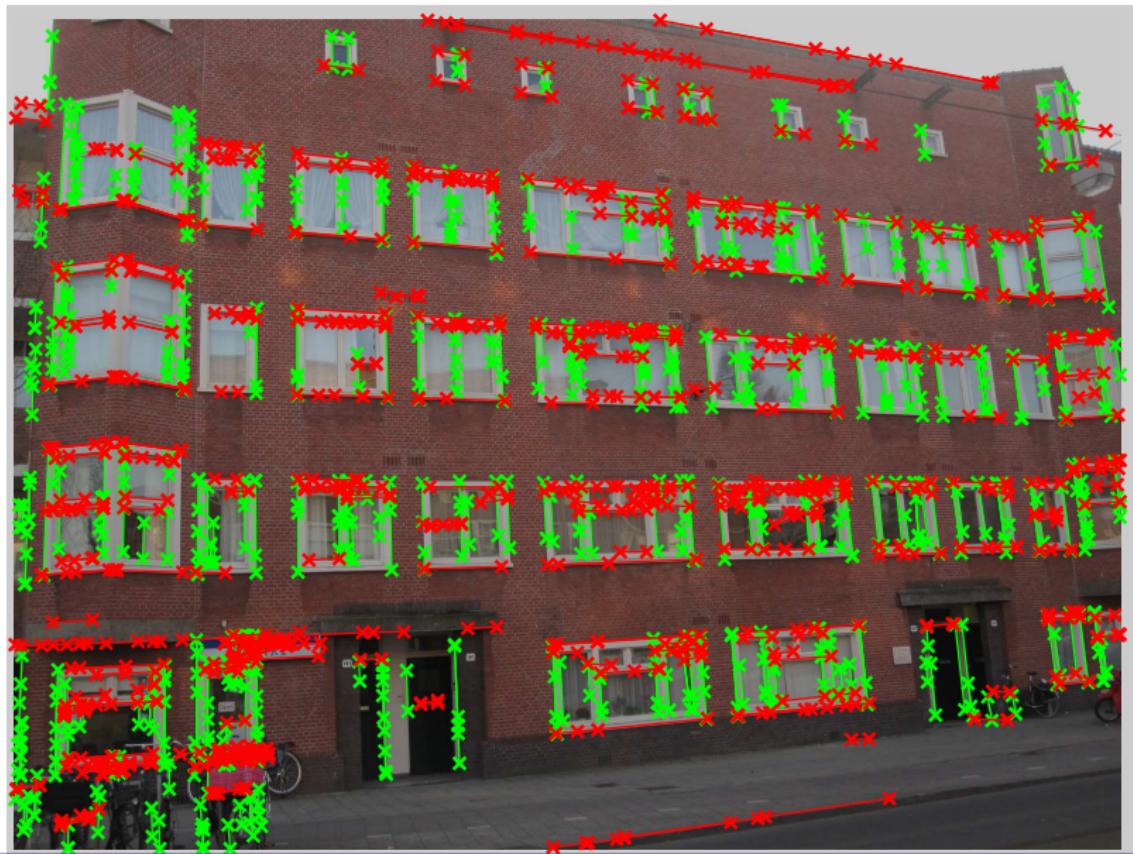
Connected corner

- ▶ horizontal and vertical edge share same corner e.g. L
- ▶ If P_i, P_e are close enough form connected corner

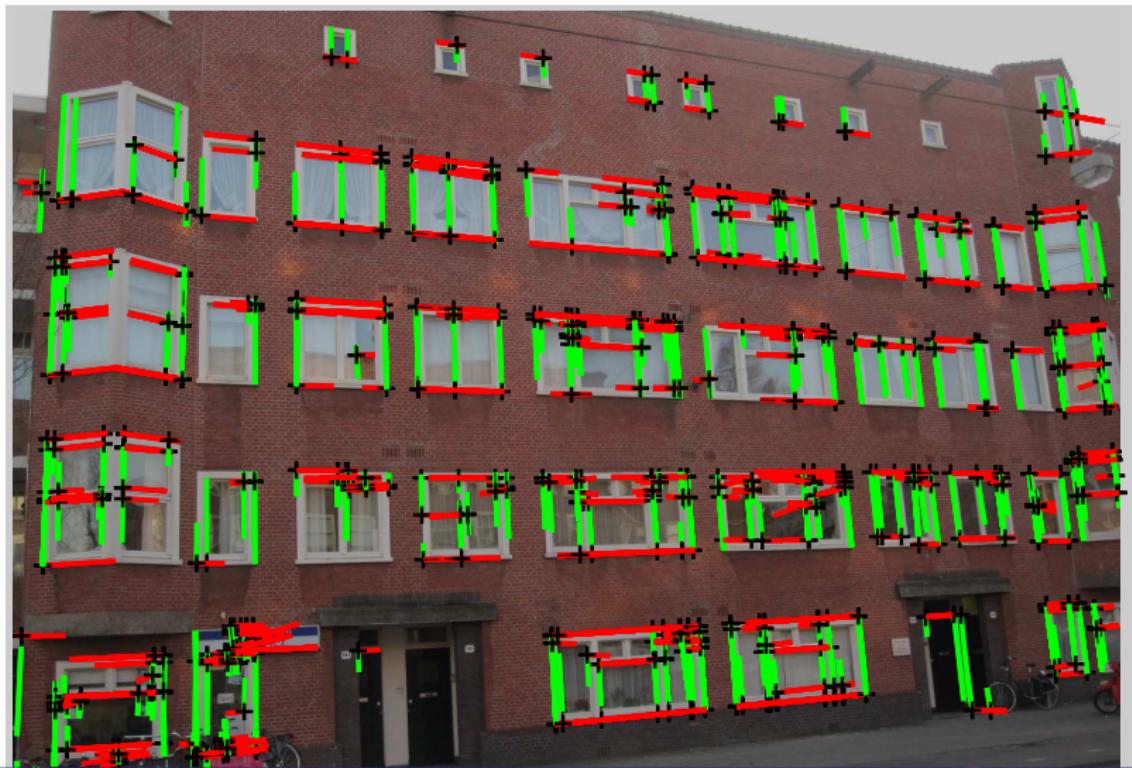
Connected corner

- ▶ horizontal and vertical edge share same corner e.g. L
- ▶ If P_i , P_e are close enough form connected corner
- ▶ Tolerate gap or extension





Connected Corners



Window area extraction

- ▶ How to extract the window areas?

Window area extraction

- ▶ How to extract the window areas?
- ▶ Mirror from diagonal through endpoints

Window area extraction

- ▶ How to extract the window areas?
- ▶ Mirror from diagonal through endpoints
- ▶ L-shapes becomes quadrangle shaped window areas

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
 - ▶ Cluster connected corner on location and length

Future research

- ▶ L-shapes, U-shapes
- ▶ Analysis of substructure of windows
 - ▶ Cluster connected corner on location and length
 - ▶ Close connected corners of same size will be grouped

Future research

- ▶ L-shapes, U-shapes
- ▶ Analysis of substructure of windows
 - ▶ Cluster connected corner on location and length
 - ▶ Close connected corners of same size will be grouped
 - ▶ **Maximum inter-cluster distance correlates with size sub window**

Future research

- ▶ L-shapes, U-shapes
- ▶ Analysis of substructure of windows
 - ▶ Cluster connected corner on location and length
 - ▶ Close connected corners of same size will be grouped
 - ▶ Maximum inter-cluster distance correlates with size sub window
 - ▶ Assume nr of sub windows to estimate
 - ▶ max and min size of window
 - ▶ number of clusters
 - ▶ max inter-cluster distance

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



Facade rectification

- ▶ 3D model of building gives plane that corresponds to wall

Facade rectification

- ▶ 3D model of building gives plane that corresponds to wall
- ▶ plane's normal vector: b

Facade rectification

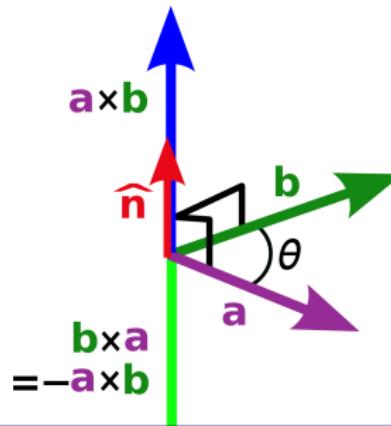
- ▶ 3D model of building gives plane that corresponds to wall
- ▶ plane's normal vector: b
- ▶ (FIT3D) Camera's heading: a

Facade rectification

- ▶ 3D model of building gives plane that corresponds to wall
- ▶ plane's normal vector: b
- ▶ (FIT3D) Camera's heading: a
- ▶ if $a \parallel b$, view is frontal

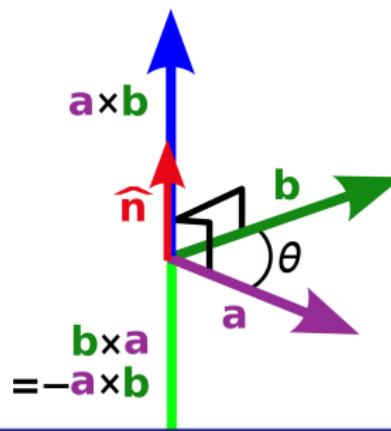
Facade rectification

- ▶ 3D model of building gives plane that corresponds to wall
- ▶ plane's normal vector: b
- ▶ (FIT3D) Camera's heading: a
- ▶ if $a \parallel b$, view is frontal
- ▶ Rotation matrix R is calculated and applied



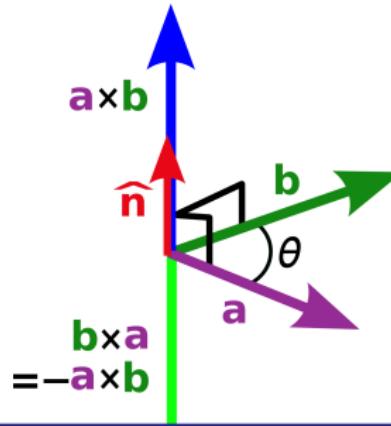
Facade rectification

- ▶ 3D model of building gives plane that corresponds to wall
- ▶ plane's normal vector: b
- ▶ (FIT3D) Camera's heading: a
- ▶ if $a \parallel b$, view is frontal
- ▶ Rotation matrix R is calculated and applied
 - ▶ orthogonal rotation vector $a \times b$



Facade rectification

- ▶ 3D model of building gives plane that corresponds to wall
- ▶ plane's normal vector: b
- ▶ (FIT3D) Camera's heading: a
- ▶ if $a \parallel b$, view is frontal
- ▶ Rotation matrix R is calculated and applied
 - ▶ orthogonal rotation vector $a \times b$
 - ▶ angle θ



Unrectified facade



Rectified facade







Extracting the window alignment

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

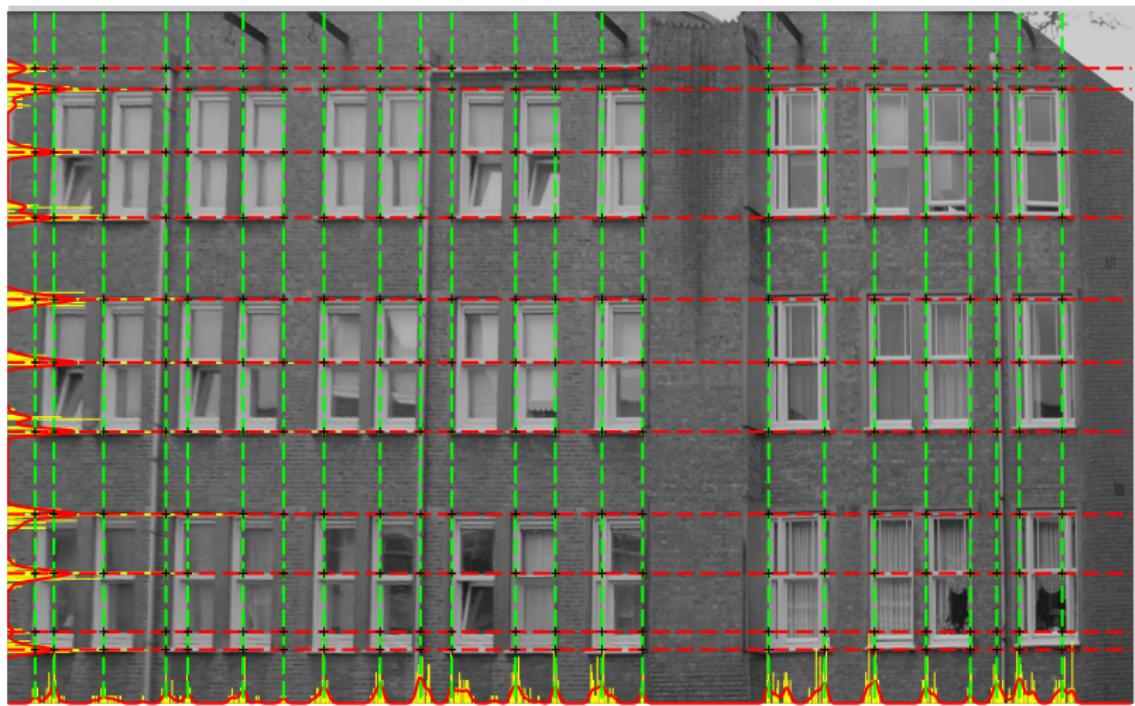
Extracting the window alignment

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

Extracting the window alignment

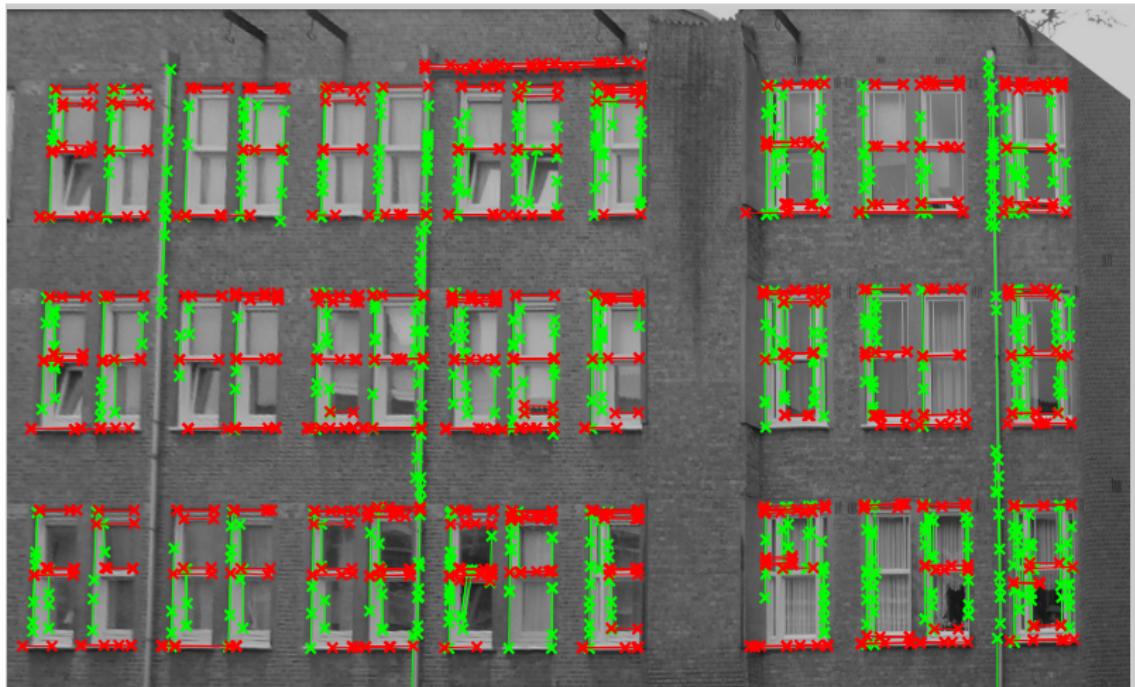
- ▶ Window alignment line
 - ▶ *"A horizontal or vertical line that aligns multiple windows"*
- ▶ Alignment lines provide grid of blocks
- ▶ Window, non-window areas

Window alignment lines example



The idea

- ▶ Amount of Houghlines high at window locations



The algorithm for vertical alignment (columns)

- ▶ Isolate *vertical* Hough lines

The algorithm for vertical alignment (columns)

- ▶ Isolate *vertical* Hough lines
- ▶ Extract pixel coordinates of endpoints

The algorithm for vertical alignment (columns)

- ▶ Isolate *vertical* Hough lines
- ▶ Extract pixel coordinates of endpoints
 - ▶ set of (2 dimensional) coordinates (x,y)

The algorithm for vertical alignment (columns)

- ▶ Isolate *vertical* Hough lines
- ▶ Extract pixel coordinates of endpoints
 - ▶ set of (2 dimensional) coordinates (x,y)
- ▶ Discard least informative dimension

The algorithm for vertical alignment (columns)

- ▶ Isolate *vertical* Hough lines
- ▶ Extract pixel coordinates of endpoints
 - ▶ set of (2 dimensional) coordinates (x,y)
- ▶ Discard least informative dimension
 - ▶ the height is irrelevant for column division

The algorithm for vertical alignment (columns)

- ▶ Isolate *vertical* Hough lines
- ▶ Extract pixel coordinates of endpoints
 - ▶ set of (2 dimensional) coordinates (x,y)
- ▶ Discard least informative dimension
 - ▶ the height is irrelevant for column division
 - ▶ project to x-axis (by discard y-value)

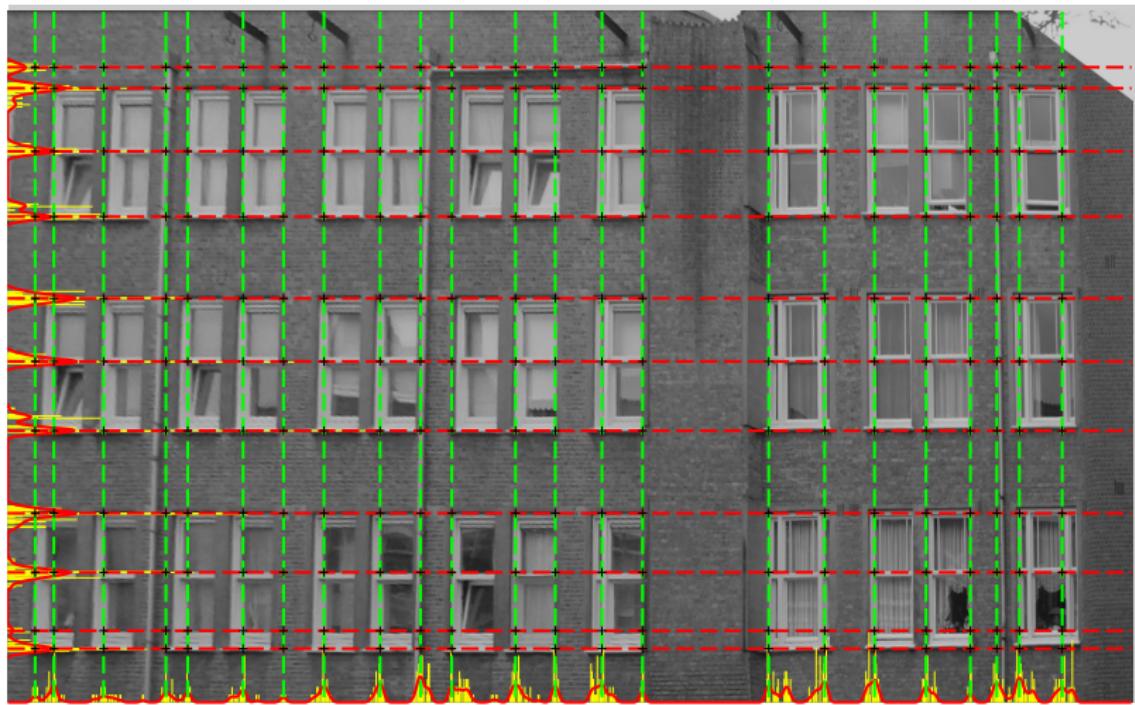
The algorithm for vertical alignment (columns)

- ▶ Isolate *vertical* Hough lines
- ▶ Extract pixel coordinates of endpoints
 - ▶ set of (2 dimensional) coordinates (x,y)
- ▶ Discard least informative dimension
 - ▶ the height is irrelevant for column division
 - ▶ project to x-axis (by discard y-value)
 - ▶ **gives a set of (1 dimensional) x-values**

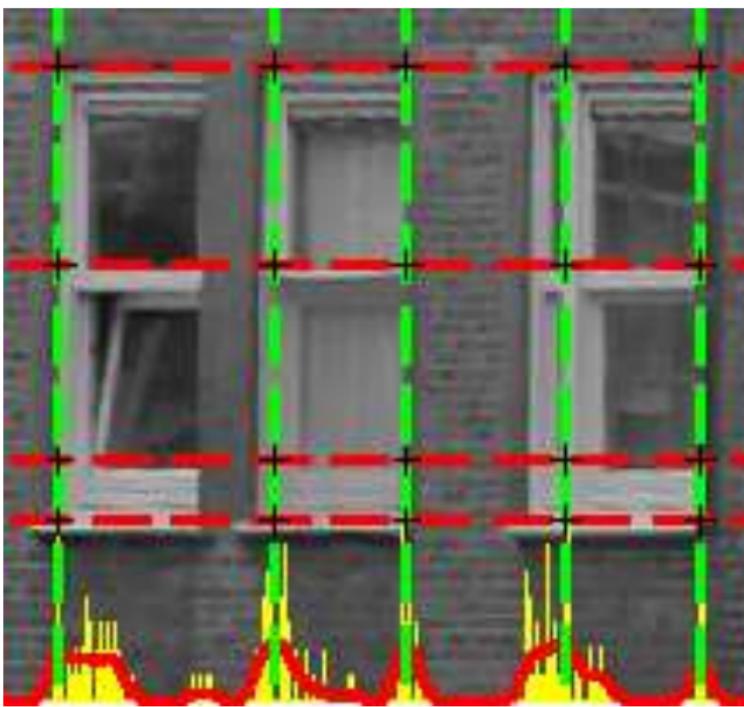
The algorithm for vertical alignment (columns)

- ▶ Isolate *vertical* Hough lines
- ▶ Extract pixel coordinates of endpoints
 - ▶ set of (2 dimensional) coordinates (x,y)
- ▶ Discard least informative dimension
 - ▶ the height is irrelevant for column division
 - ▶ project to x-axis (by discard y-value)
 - ▶ gives a set of (1 dimensional) x-values
- ▶ Create histogram: count number of values on each x-position

Window alignment lines

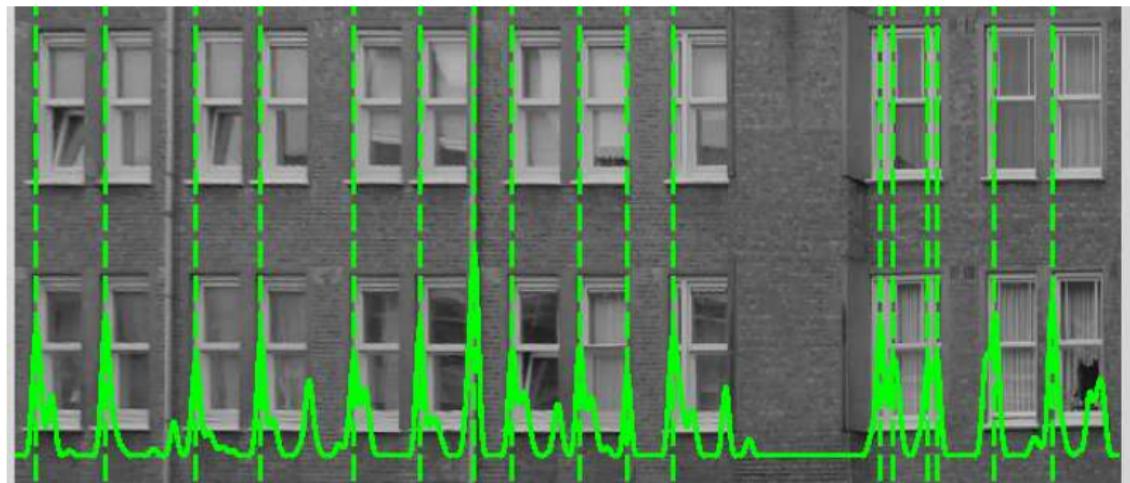


Window alignment lines



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





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
 - ▶ occlusion doesn't affect horizontal edges

Alternative window alignment

- ▶ Idea
 - ▶ occlusion doesn't affect horizontal edges
 - ▶ on vertical alignment positions appears a big increase/decrease of horizontal lines

Alternative window alignment

- ▶ Idea
 - ▶ occlusion doesn't affect horizontal edges
 - ▶ on vertical alignment positions appears a big increase/decrease of horizontal lines
 - ▶ take this increase/decrease into account

Alternative window alignment

- ▶ Idea
 - ▶ occlusion doesn't affect horizontal edges
 - ▶ on vertical alignment positions appears a big increase/decrease of horizontal lines
 - ▶ take this increase/decrease into account
 - ▶ instead of amount of Houghlines

Alternative window alignment

- ▶ Idea
 - ▶ occlusion doesn't affect horizontal edges
 - ▶ on vertical alignment positions appears a big increase/decrease of horizontal lines
 - ▶ take this increase/decrease into account
 - ▶ instead of amount of Houghlines
 - ▶ use entire line

Alternative window alignment

- ▶ Idea
 - ▶ occlusion doesn't affect horizontal edges
 - ▶ on vertical alignment positions appears a big increase/decrease of horizontal lines
 - ▶ take this increase/decrease into account
 - ▶ instead of amount of Houghlines
 - ▶ use entire line
 - ▶ instead of only the endpoints

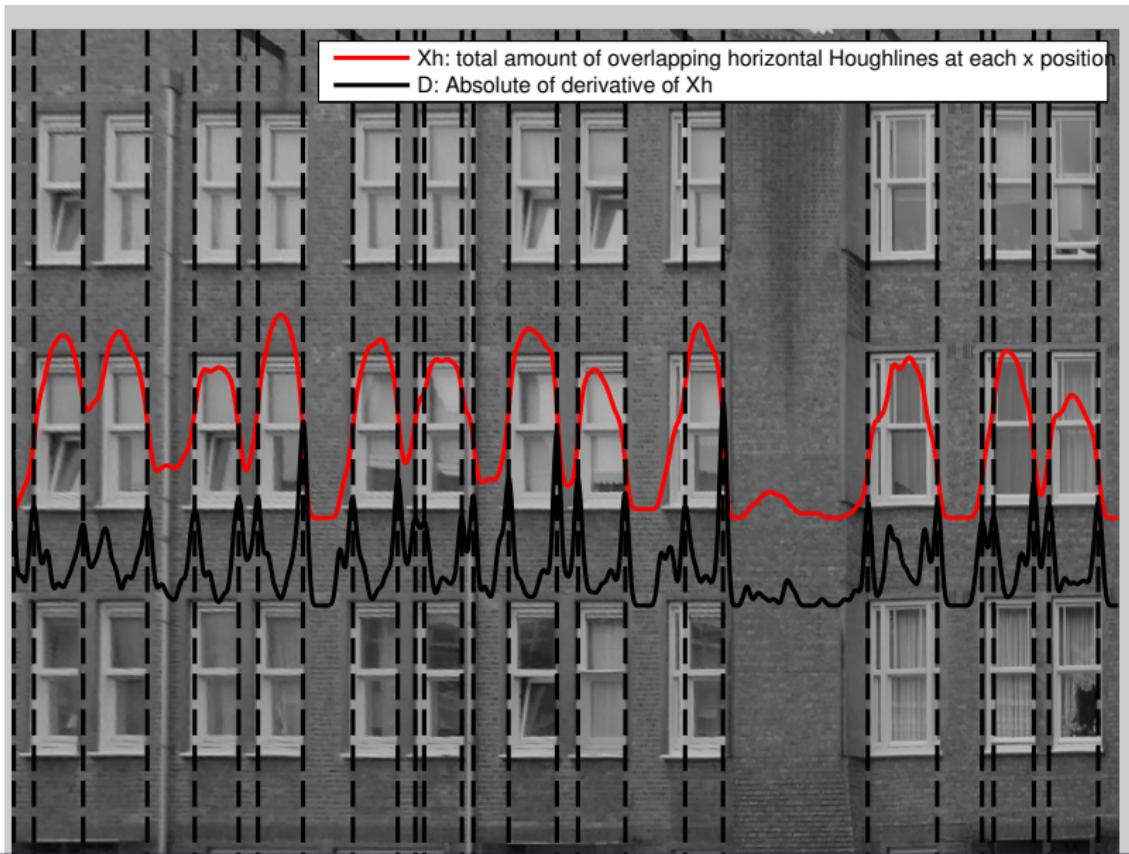
Alternative window alignment

- ▶ Idea
 - ▶ occlusion doesn't affect horizontal edges
 - ▶ on vertical alignment positions appears a big increase/decrease of horizontal lines
 - ▶ take this increase/decrease into account
 - ▶ instead of amount of Houghlines
 - ▶ use entire line
 - ▶ instead of only the endpoints
- ▶ X_h defines number of overlapping horizontal lines on each x position

Alternative window alignment

- ▶ Idea
 - ▶ occlusion doesn't affect horizontal edges
 - ▶ on vertical alignment positions appears a big increase/decrease of horizontal lines
 - ▶ take this increase/decrease into account
 - ▶ instead of amount of Houghlines
 - ▶ use entire line
 - ▶ instead of only the endpoints
- ▶ X_h defines number of overlapping horizontal lines on each x position
- ▶ peak function

$$D = \text{abs}(X'_h)$$



Fusing the window alignment methods

- ▶ Plot both methods

Fusing the window alignment methods

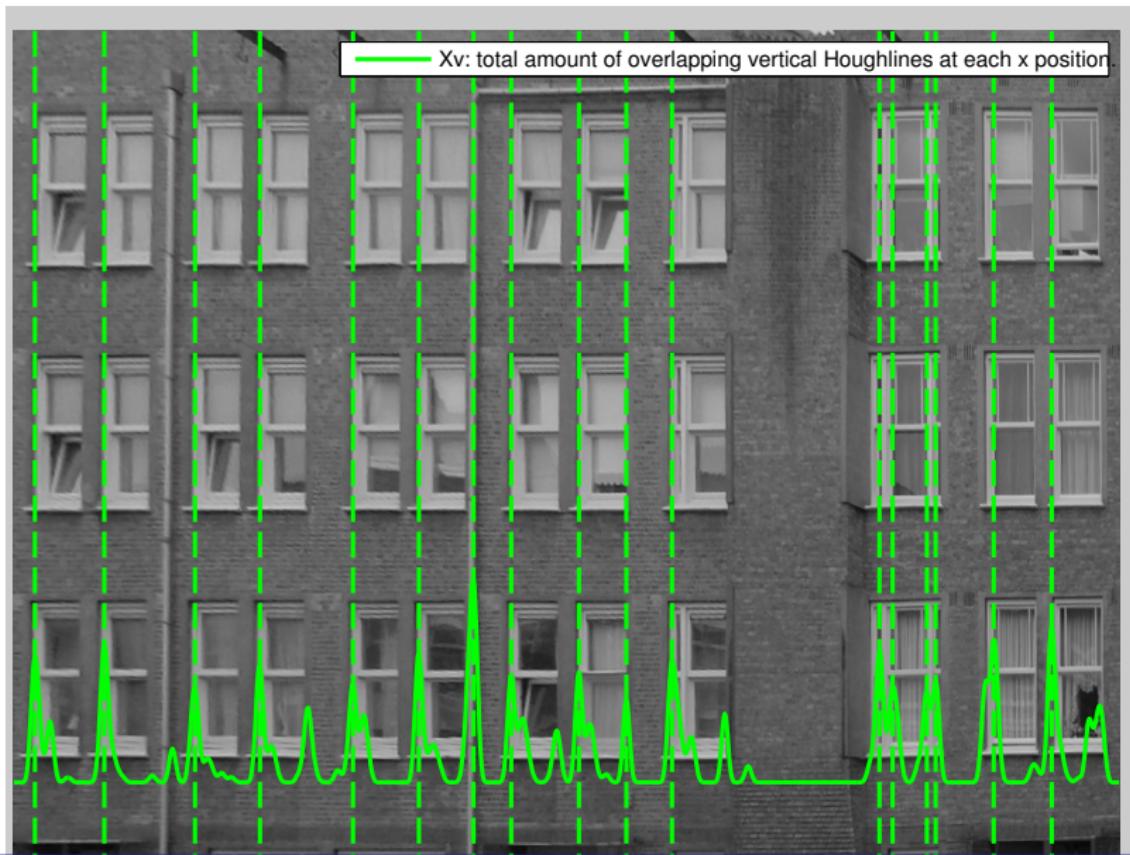
- ▶ Plot both methods
- ▶ Increase threshold (less but more certain alignment lines)

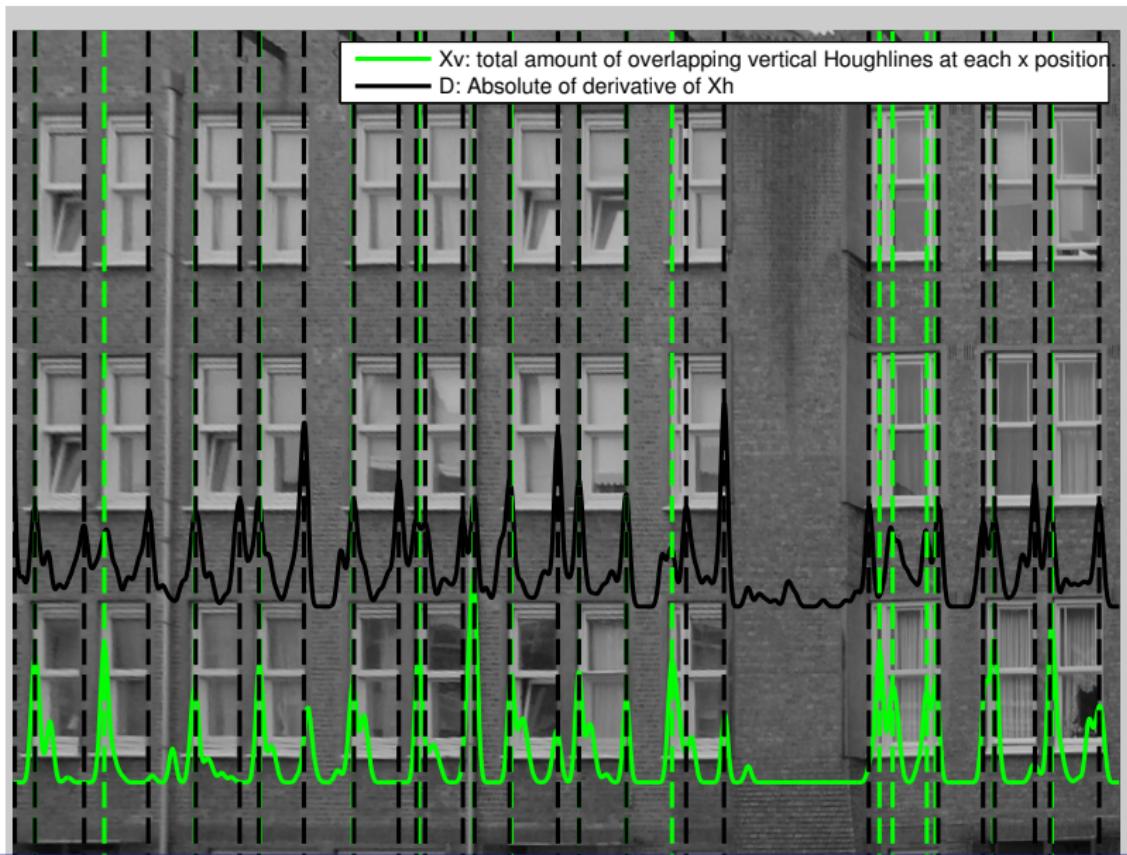
Fusing the window alignment methods

- ▶ Plot both methods
- ▶ Increase threshold (less but more certain alignment lines)
- ▶ Peak found by both methods?

Fusing the window alignment methods

- ▶ Plot both methods
- ▶ Increase threshold (less but more certain alignment lines)
- ▶ Peak found by both methods?
 - ▶ Merge peaks that are close





Other dataset







Window classification

- ▶ Alignment lines give grid of blocks

Window classification

- ▶ Alignment lines give grid of blocks
- ▶ Classify blocks

Window classification

- ▶ Alignment lines give grid of blocks
- ▶ Classify blocks
 - ▶ window or non-window block

Window classification

- ▶ Alignment lines give grid of blocks
- ▶ Classify blocks
 - ▶ window or non-window block
- ▶ Two methods:

Window classification

- ▶ Alignment lines give grid of blocks
- ▶ Classify blocks
 - ▶ window or non-window block
- ▶ Two methods:
 - ▶ Basic classification

Window classification

- ▶ Alignment lines give grid of blocks
- ▶ Classify blocks
 - ▶ window or non-window block
- ▶ Two methods:
 - ▶ Basic classification
 - ▶ based on amount of Houghline endpoints

Window classification

- ▶ Alignment lines give grid of blocks
- ▶ Classify blocks
 - ▶ window or non-window block
- ▶ Two methods:
 - ▶ Basic classification
 - ▶ based on amount of Houghline endpoints
 - ▶ Improved classification

Window classification

- ▶ Alignment lines give grid of blocks
- ▶ Classify blocks
 - ▶ window or non-window block
- ▶ Two methods:
 - ▶ Basic classification
 - ▶ based on amount of Houghline endpoints
 - ▶ Improved classification
 - ▶ based on shape of histogram function

Basic window classification assumptions

- ▶ Assumptions:

Basic window classification assumptions

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

Basic window classification assumptions

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

Basic window classification assumptions

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

Basic window classification assumptions

- ▶ Assumptions:
 - ▶ a block contains a high amount of Houghlines: window
 - ▶ a block contains low amount of Houghlines: non-window
- ▶ 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 Houghline endpoints in each blockrow

Basic window classification algorithm

- ▶ Exploit window alignment:
 - ▶ classify full block rows and block columns
- ▶ Count the nr of Houghline endpoints in each blockrow
- ▶ **discard blockrow size influence (normalize):**

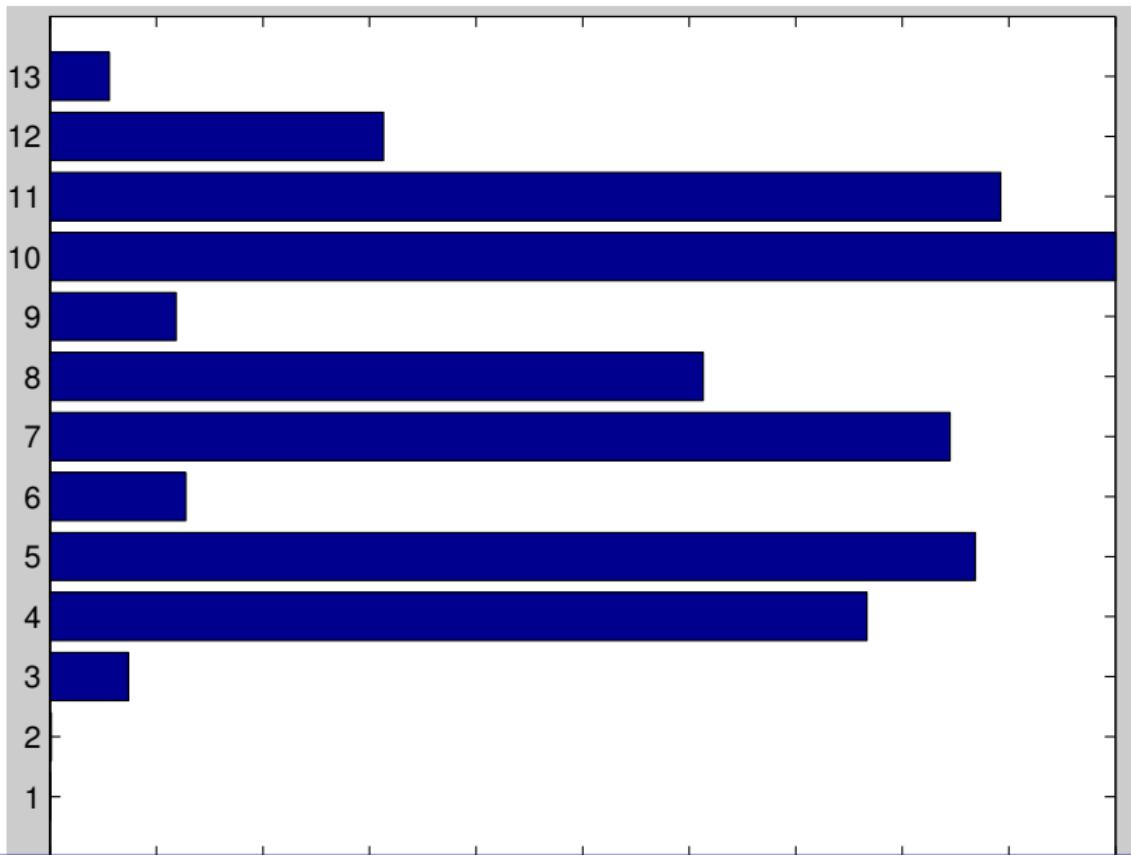
Basic window classification algorithm

- ▶ Exploit window alignment:
 - ▶ classify full block rows and block columns
- ▶ Count the nr of Houghline endpoints in each blockrow
- ▶ discard blockrow size influence (normalize):
- ▶

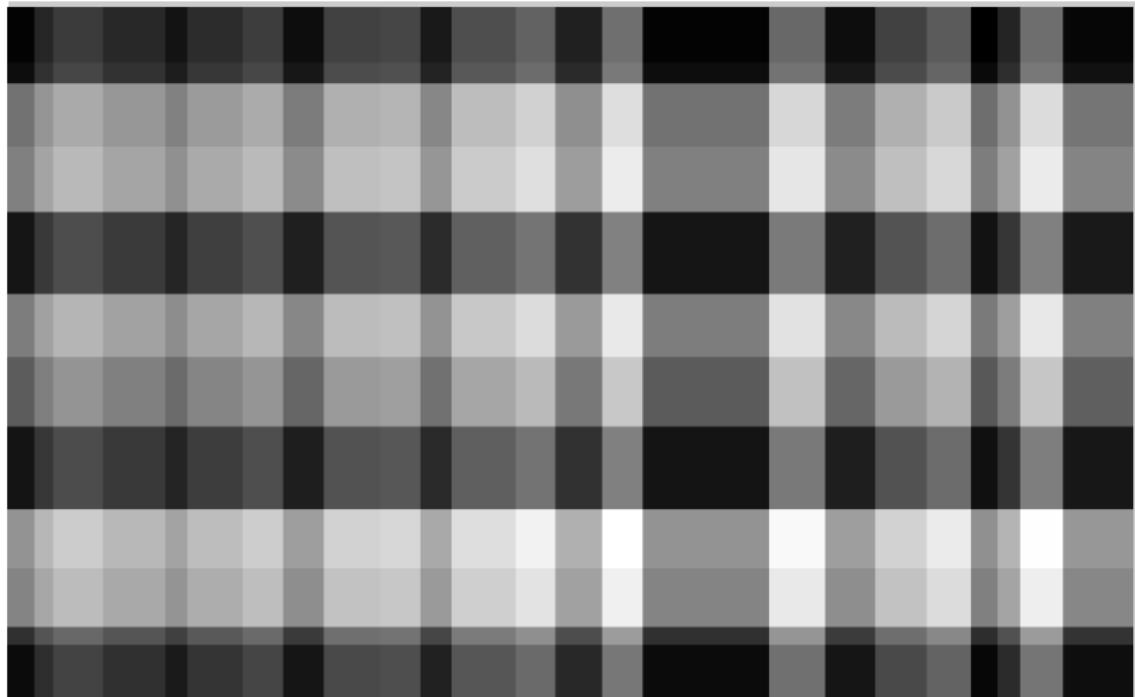
$$\forall R_i \in \{1..numRows\} : R_i = \frac{HoughlinePxCount}{R_i^{width} \cdot R_i^{height}}$$

Basic window classification algorithm

- ▶ Exploit window alignment:
 - ▶ classify full block rows and block columns
- ▶ Count the nr of Houghline endpoints in each blockrow
- ▶ discard blockrow size influence (normalize):
 - ▶
$$\forall R_i \in \{1..numRows\} : R_i = \frac{HoughlinePxCount}{R_i^{width} \cdot R_i^{height}}$$
- ▶ Output: $\|R\|$ scalar values that ranks a row being window area or not

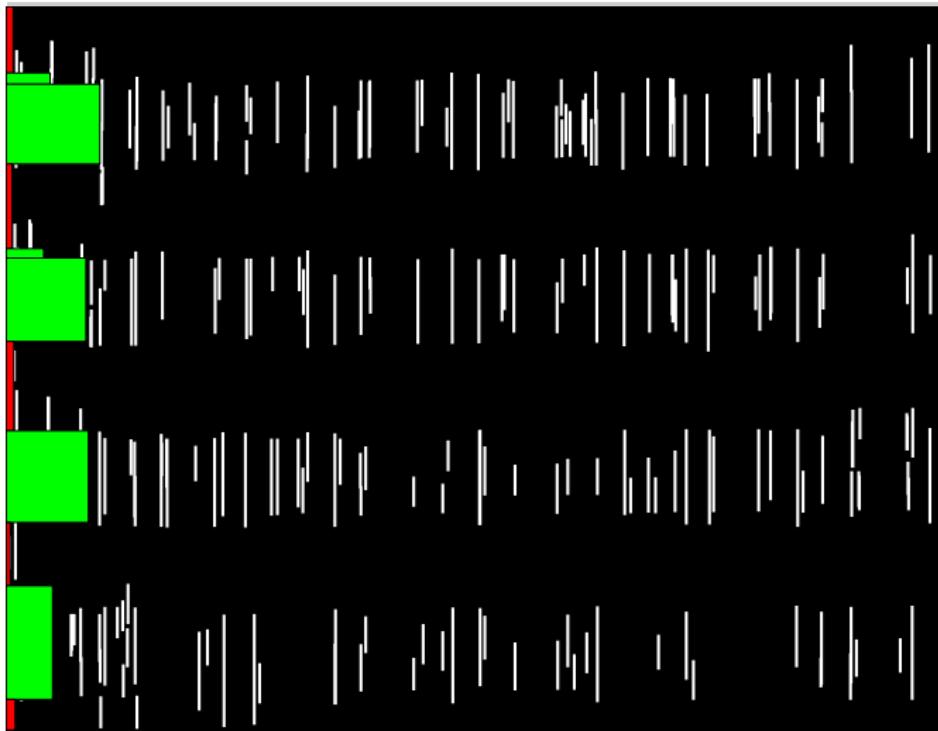


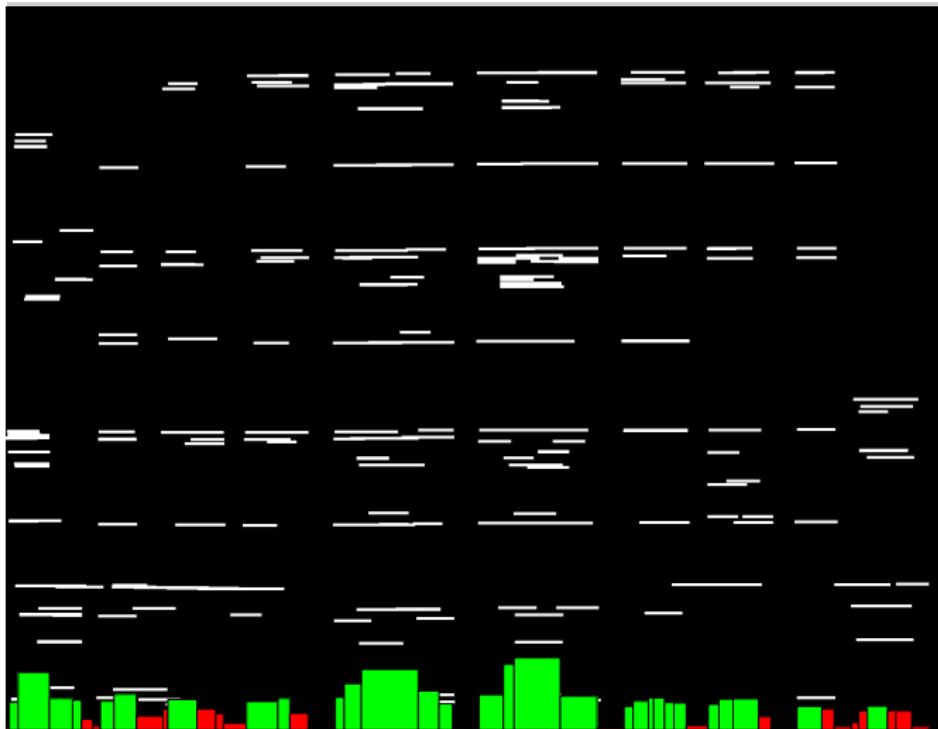


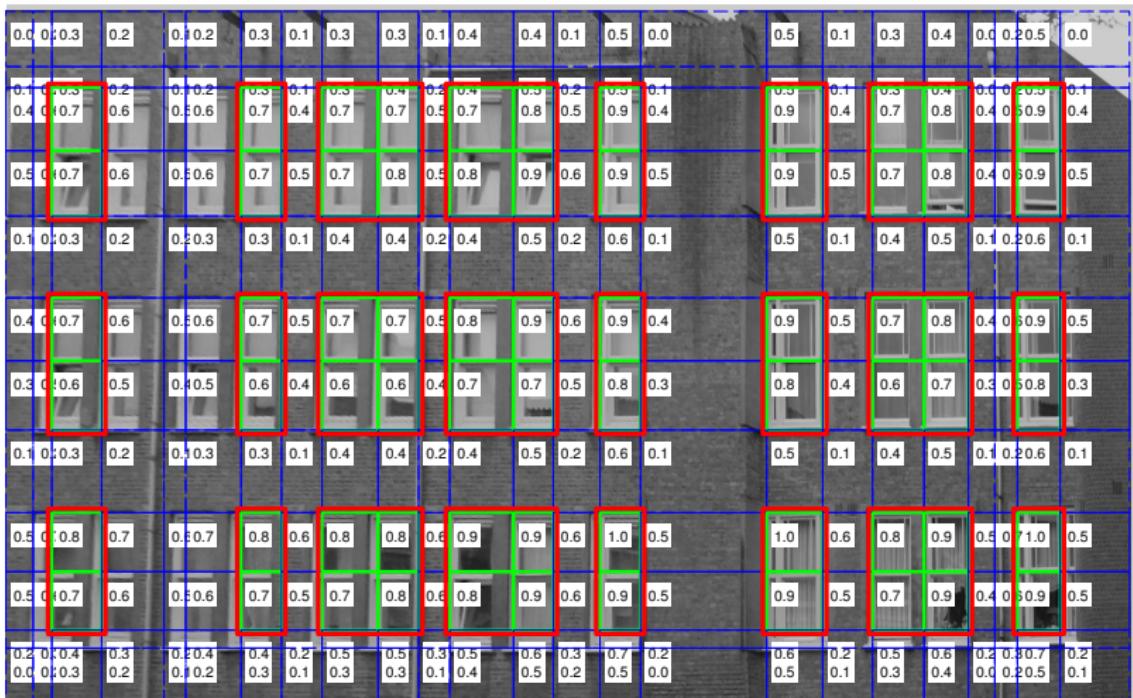


From certainty to classification

- ▶ *k*-means classification on R and C values







Improved window classification

- ▶ Use idea of alternative alignment

Improved window classification

- ▶ Use idea of alternative alignment
- ▶ Amount of Houghlines

Improved window classification

- ▶ Use idea of alternative alignment
- ▶ Amount of Houghlines
 - ▶ increases towards center of window area

Improved window classification

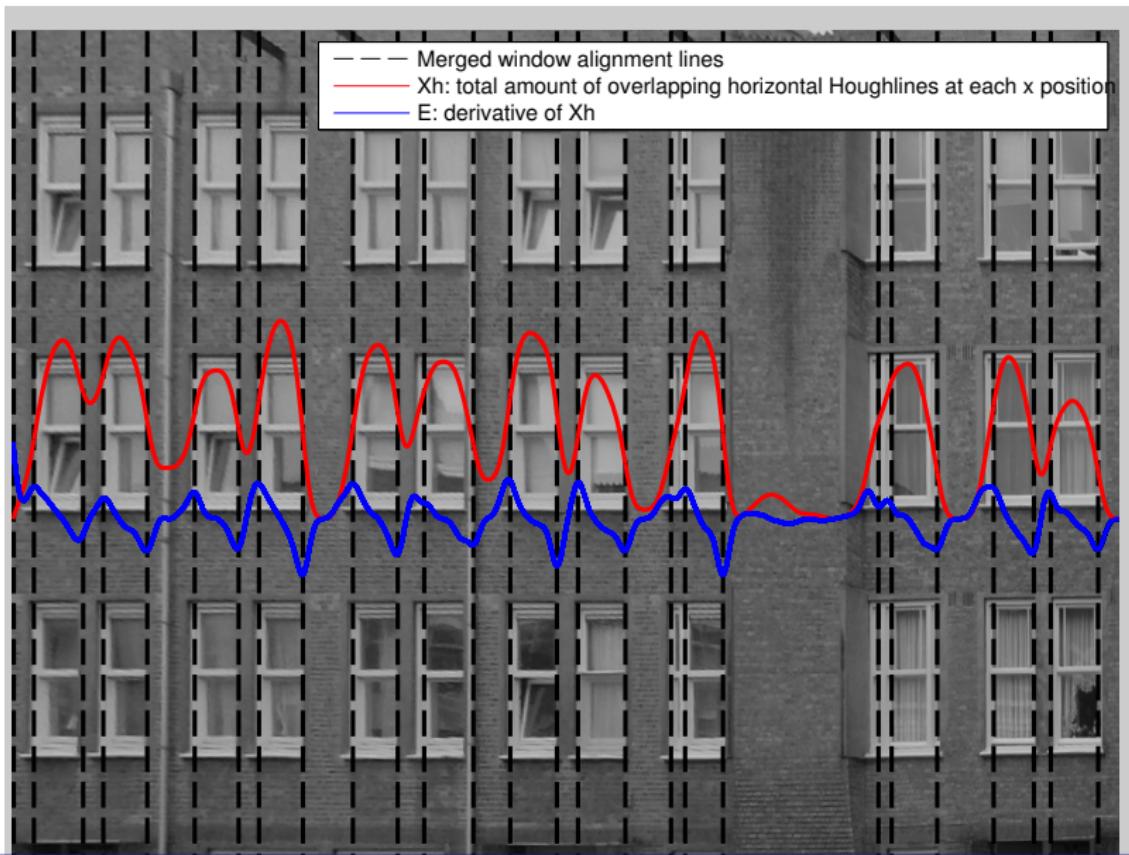
- ▶ Use idea of alternative alignment
- ▶ Amount of Houghlines
 - ▶ increases towards center of window area
 - ▶ decreases towards center of non-window area

Improved window classification

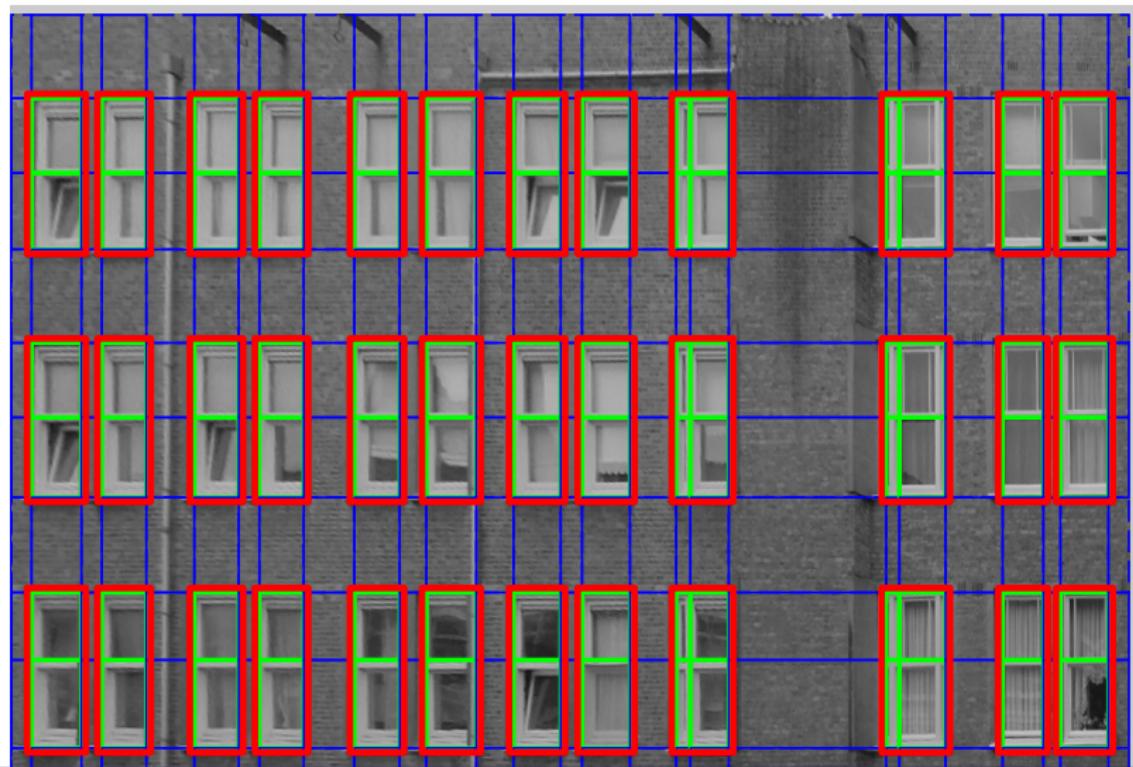
- ▶ Use idea of alternative alignment
- ▶ Amount of Houghlines
 - ▶ increases towards center of window area
 - ▶ decreases towards center of non-window area
- ▶ Shape concave - window area

Improved window classification

- ▶ Use idea of alternative alignment
- ▶ Amount of Houghlines
 - ▶ increases towards center of window area
 - ▶ decreases towards center of non-window area
- ▶ Shape concave - window area
- ▶ **Shape convex - non-window area**



Result



Conclusion: Connected corner

Conclusion: Connected corner



Conclusion: Connected corner



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 the building

Conclusion: Connected corner

- ▶ Detection rate 97%
- ▶ Suitable for scenes with variation in window size/type
- ▶ Small requirement on input data
 - ▶ neither 3D information about the 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 the 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 the 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 the building
 - ▶ nor rectification is needed
- ▶ Future research
 - ▶ U shapes
 - ▶ **Analysis of sub window structure**

Conclusion: histogram based window alignment

- ▶ Interpretation amount of Houghlines = strong approach towards window detection

Conclusion: histogram based window alignment

- ▶ Interpretation amount of Houghlines = strong approach towards window detection
- ▶ Two window detection methods

Conclusion: histogram based window alignment

- ▶ Interpretation amount of Houghlines = strong approach towards window detection
- ▶ Two window detection methods
- ▶ Alternative window alignment performs better, used:

Conclusion: histogram based window alignment

- ▶ Interpretation amount of Houghlines = strong approach towards window detection
- ▶ Two window detection methods
- ▶ Alternative window alignment performs better, used:
 - ▶ a strong combination horizontal and vertical Houghlines histograms

Conclusion: histogram based window alignment

- ▶ Interpretation amount of Houghlines = strong approach towards window detection
- ▶ Two window detection methods
- ▶ Alternative window alignment performs better, used:
 - ▶ a strong combination horizontal and vertical Houghlines histograms
 - ▶ **high order (derivative) shape interpretation of the 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 classification)

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 classification)
- ▶ 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 classification)
- ▶ Method II: based on shape of Houghlines histogram
 - ▶ based on 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 classification)
- ▶ Method II: based on shape of Houghlines histogram
 - ▶ based on 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 classification)
- ▶ Method II: based on shape of Houghlines histogram
 - ▶ based on strong increase/decrease of Houghlines
 - ▶ very robust, at least 97% detection rate on all datasets, why?
 - ▶ **robust to alignment errors (works on center area)**

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 classification)
- ▶ Method II: based on shape of Houghlines histogram
 - ▶ based on strong increase/decrease of Houghlines
 - ▶ very robust, at least 97% detection rate on all datasets, why?
 - ▶ robust to alignment errors (works on center area)
 - ▶ 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 classification)
- ▶ Method II: based on shape of Houghlines histogram
 - ▶ based on strong increase/decrease of Houghlines
 - ▶ very robust, at least 97% detection rate on all datasets, why?
 - ▶ robust to alignment errors (works on center area)
 - ▶ higher order interpretation of histogram function
 - ▶ **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 classification)
- ▶ Method II: based on shape of Houghlines histogram
 - ▶ based on strong increase/decrease of Houghlines
 - ▶ very robust, at least 97% detection rate on all datasets, why?
 - ▶ robust to alignment errors (works on center area)
 - ▶ higher order interpretation of histogram function
 - ▶ the increase/decrease is relative
 - ▶ **robust to variation in window type and illumination conditions**

Conclusion

- We retrieved different semantics of urban scenes

Conclusion

- ▶ We retrieved different semantics of urban scenes
- ▶ Similar to humans

Conclusion

- ▶ We retrieved different semantics of urban scenes
- ▶ Similar to humans
 - ▶ using 3D information, straight lines, right angles, etc.

Conclusion

- ▶ We retrieved different semantics of urban scenes
- ▶ Similar to humans
 - ▶ using 3D information, straight lines, right angles, etc.
- ▶ We achieved:

Conclusion

- ▶ We retrieved different semantics of urban scenes
- ▶ Similar to humans
 - ▶ using 3D information, straight lines, right angles, etc.
- ▶ We achieved:
 - ▶ The extraction of a skyline in different urban scenes

Conclusion

- ▶ We retrieved different semantics of urban scenes
- ▶ Similar to humans
 - ▶ using 3D information, straight lines, right angles, etc.
- ▶ We achieved:
 - ▶ The extraction of a skyline in different urban scenes
 - ▶ **The 3D reconstruction of a building**

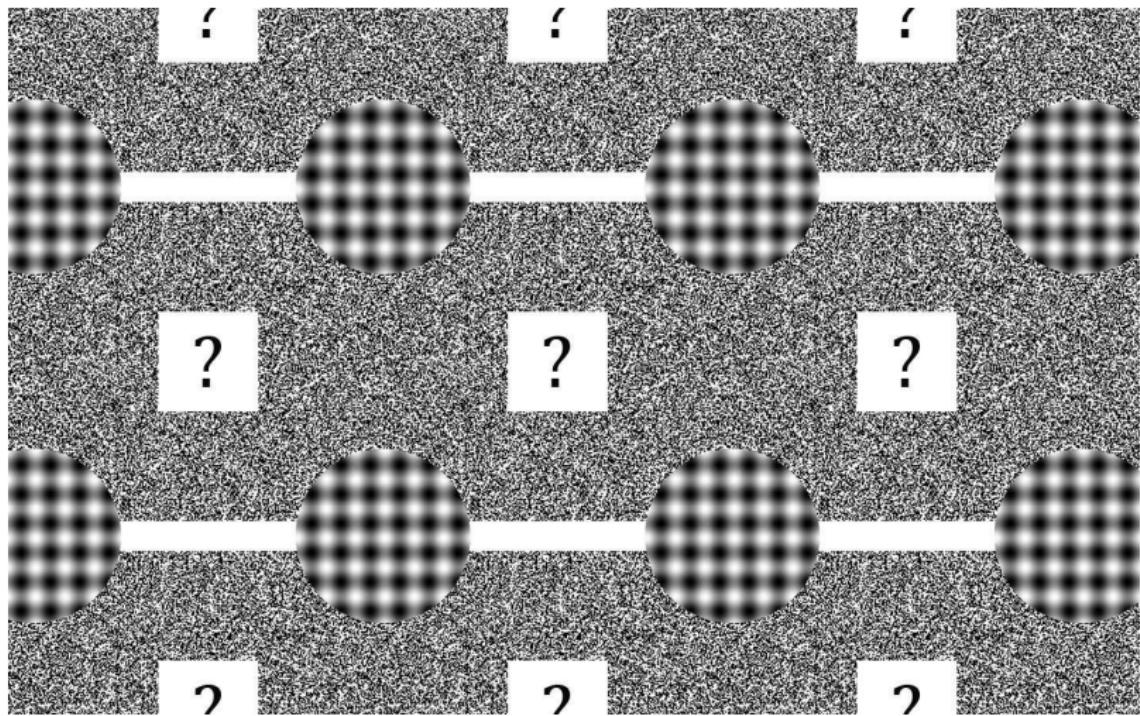
Conclusion

- ▶ We retrieved different semantics of urban scenes
- ▶ Similar to humans
 - ▶ using 3D information, straight lines, right angles, etc.
- ▶ We achieved:
 - ▶ The extraction of a skyline in different urban scenes
 - ▶ The 3D reconstruction of a building
 - ▶ **The extraction of windows**

Conclusion

- ▶ We retrieved different semantics of urban scenes
- ▶ Similar to humans
 - ▶ using 3D information, straight lines, right angles, etc.
- ▶ We achieved:
 - ▶ The extraction of a skyline in different urban scenes
 - ▶ The 3D reconstruction of a building
 - ▶ The extraction of windows
 - ▶ For a set of urban scenes with different properties

Questions



Questions

Questions?