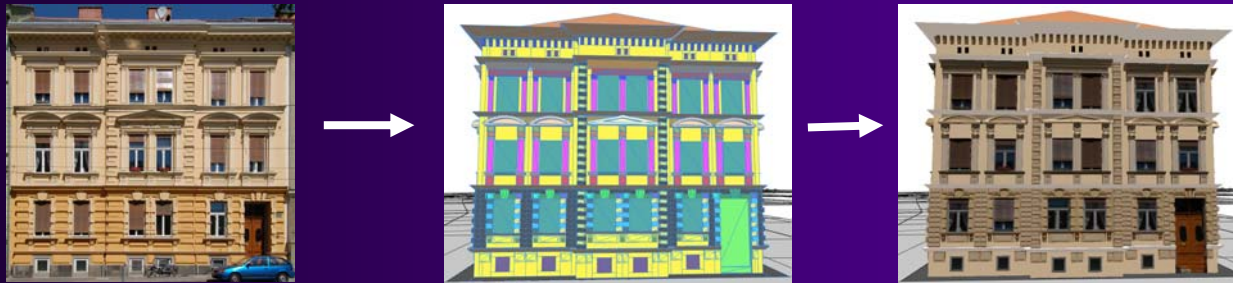
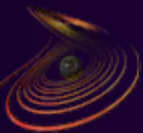


# CityFit

High-Quality Urban Reconstructions by  
Fitting Shape Grammars to Images and  
derived Textured Point Clouds



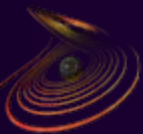
Bernhard Hohmann  
Institute of Computer Graphics & Knowledge Visualization  
Graz University of Technology  
Austria



# Motivation

## Demand for 3D city models

- Google Earth:
  - ❖ User-generated buildings
- Microsoft Virtual Earth:
  - ❖ Scanning of populated earth in 15 cm resolution (Vexcel Graz - now Microsoft Photogrammetry)



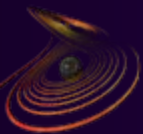
# Motivation

- State of the art: Extruded ground polygons, fully automatically from aerial images

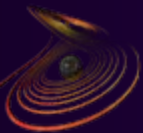
[ Zebedin et. al. ISPRS 2006 ]



- Solved problem: Detailed roof landscapes
- Remaining problem: Detailed facades



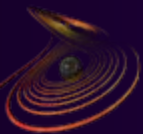
# The Challenge



# The Challenge

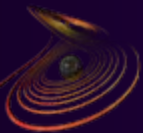
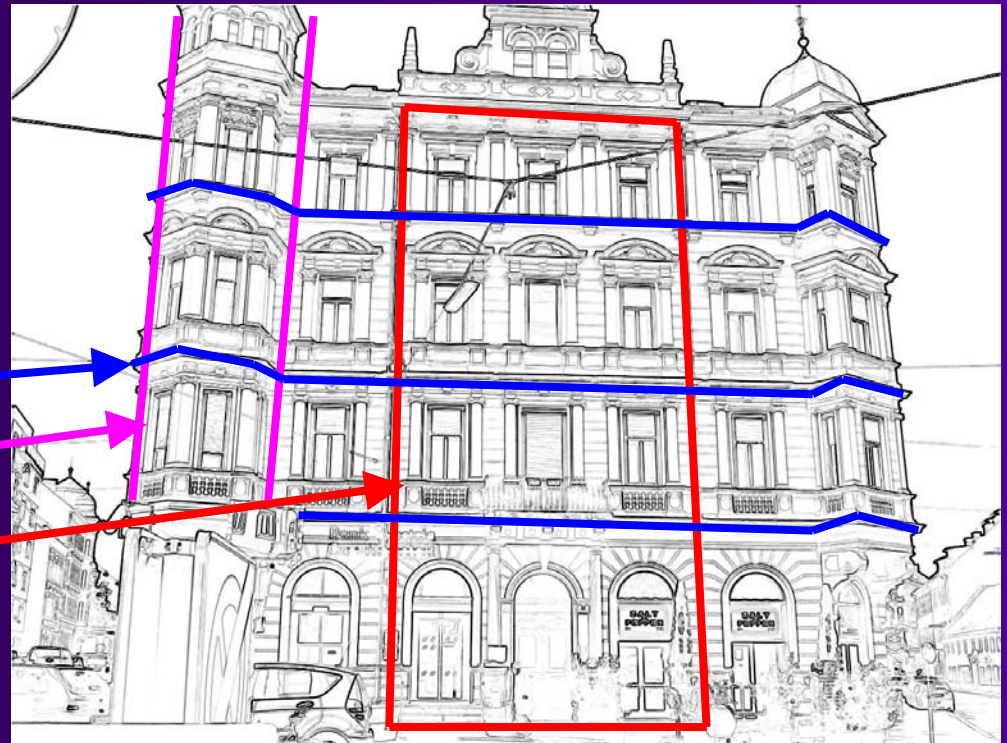


- Graz as complex use case
  - ❖ Many different styles coexist
- Ambitious Goal:  
Reconstruct 80% fully automatically



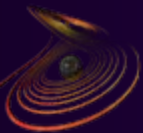
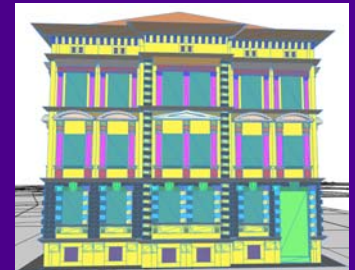
# Structuring Complex Facades

- Detect structural elements  $>50\text{cm}$  in the facades
- Create a shape vocabulary
  - ❖ Windows
  - ❖ Doors
  - ❖ Balconies
  - ❖ Columns
  - ❖ Cornices/Ledges
  - ❖ Oriels
  - ❖ Risalits

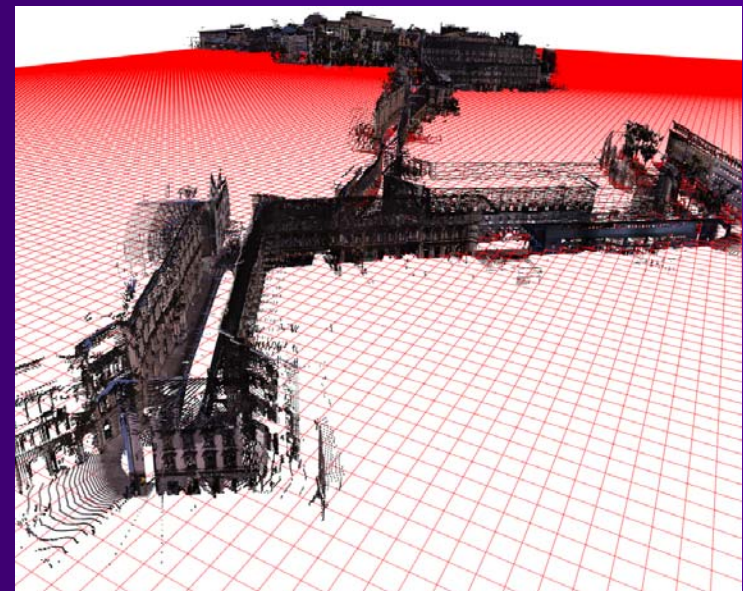


# CityFit Workflow

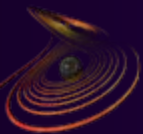
- Data Acquisition
  - ❖ Microsoft Photogrammetry (Graz)
- Detection and Recognition
  - ❖ ICG (also TU Graz)
- Representation by Shape Grammar
  - ❖ CGV
- Fitting of Terminal Symbols
  - ❖ CGV



# Input Data

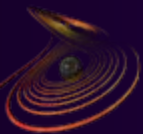
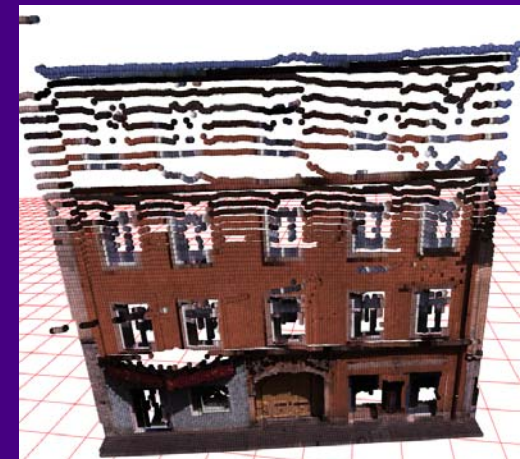
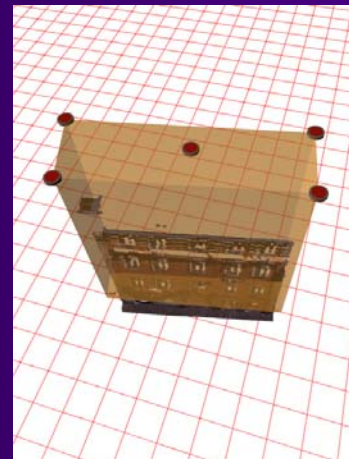
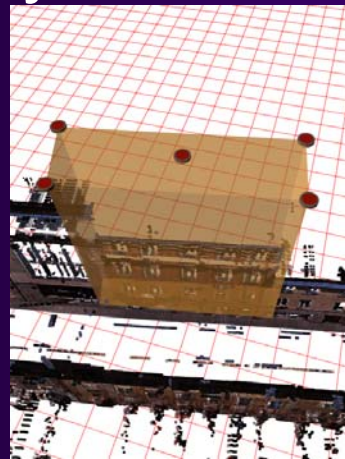
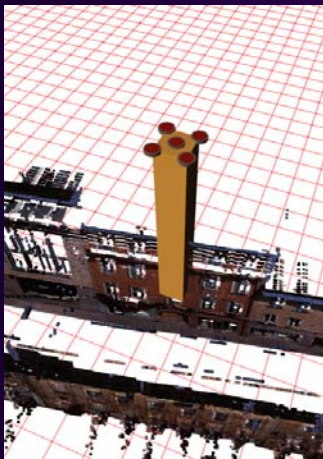
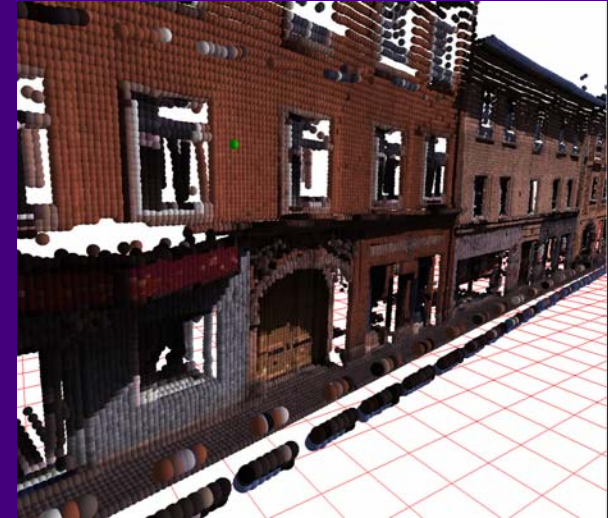


- Road-Side Photographs
- LIDAR Scans



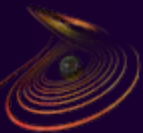
# Preprocessing

- Point Cloud Viewer
- Textured LIDAR Points
  - ❖ Spheres
- Segmentation of Facades
  - ❖ Currently Manual



# Data Representation

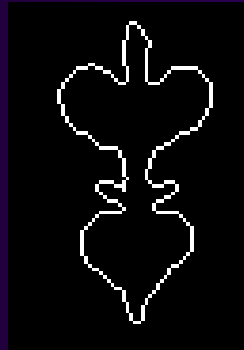
- Fixed point 16.16 bit integer coordinate system
- ~1/65 mm resolution over ~65 km
- Uniform accuracy (in contrast to float)
- Allows to test points for equality



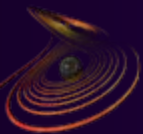
# Detection (ICG)

- Partial Shape Fragment Matcher
- Performs well for Decorative Elements

➤ Prior

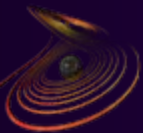


Hayko Riemenschneider (ICG)



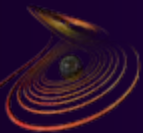
# Shape Grammar

- Conceptually simple
  - ❖ Non-Terminal Symbols (e.g. Floor)
  - ❖ Terminal Symbols (e.g. Window)
  - ❖ Replacement Rules (e.g. Floor  $\rightarrow$  5 x Window)
  - ❖ Hierarchical



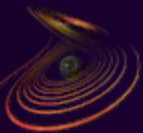
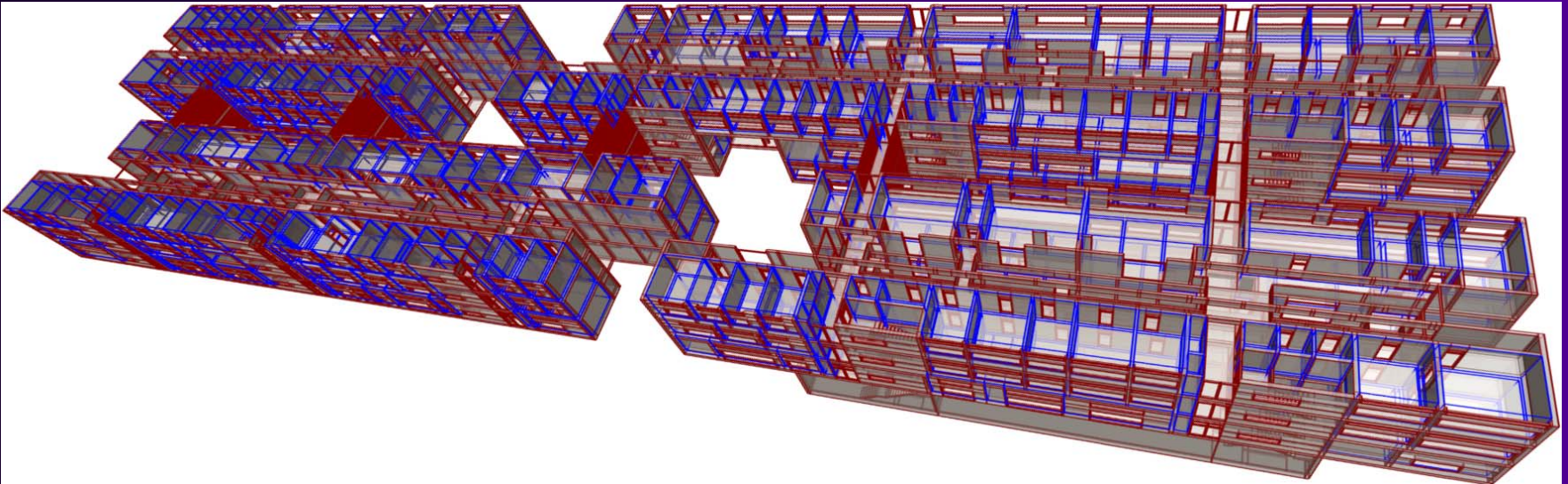
# Shape Grammar

- Based on concepts from *CGA Shape*  
[Müller et. al. SIGGRAPH 2006], [CityEngine, Procedural Inc.]
  - ❖ CGA Shape: Grammar description language
  - ❖ „Scope“: Rectangular Box
- Convex Polyhedra: Generalization of boxes
- Grammar Description Language:  
GML (*Generative Modeling Language*)
  - ❖ Stack-based
  - ❖ Easy to generate code automatically (PostScript)



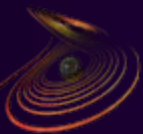
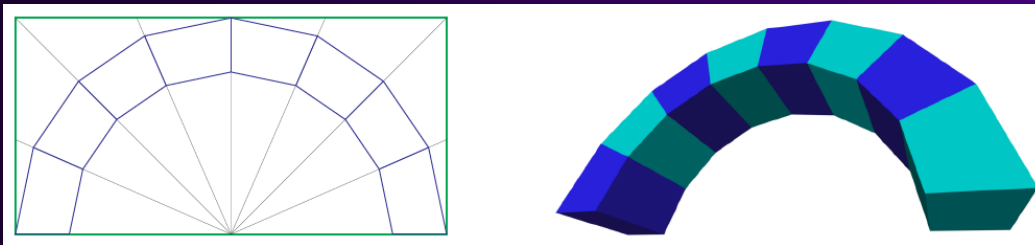
# Geometric Representation

## ➤ Box Based Version



# Geometric Representation

- Convex Polyhedra (CP)
  - ❖ Connecting any two points of a CP does not leave the CP
  
- Split Operation

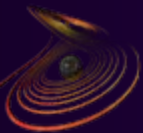
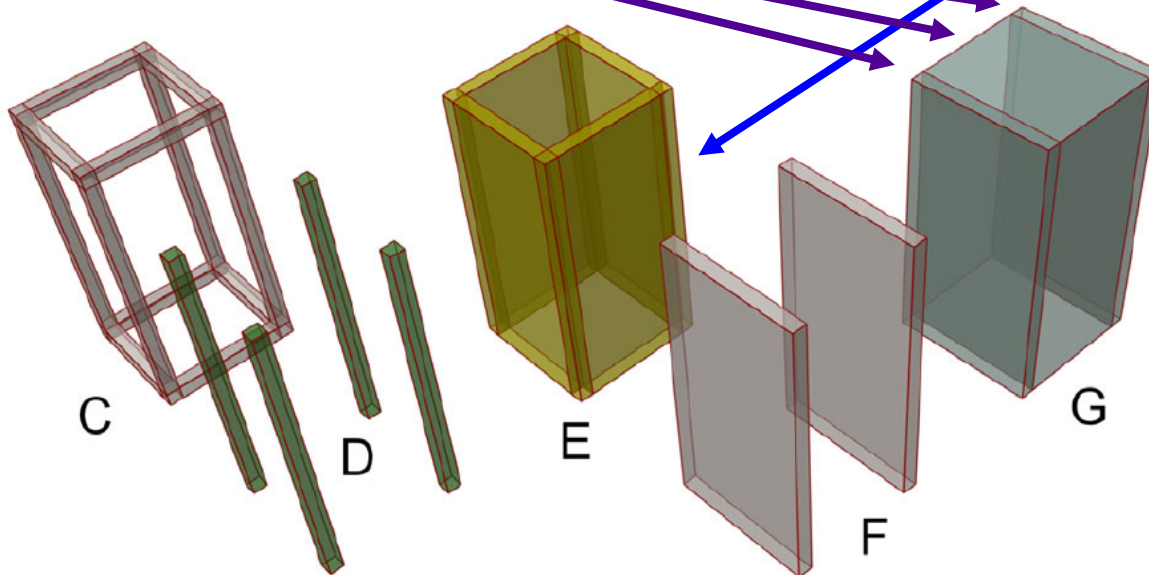


# Shape Grammar in GML

ShapeGrammar.Tools.init

```
/A { terminal-box } def  
/B { terminal-void } def  
/C { [ 0.1 -1 0.1 ] /X split E D E } def  
/D { [ 0.1 -1 0.1 ] /Y split F B F } def  
/E { [ 0.1 -1 0.1 ] /Y split G F G } def  
/F { [ 0.1 -1 0.1 ] /Z split A B A } def  
/G { [ 0.1 -1 0.1 ] /Z split A A A } def
```

```
scope (0,0,-2) move (2,1,1) scale C  
finish
```

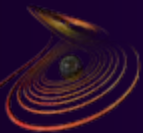
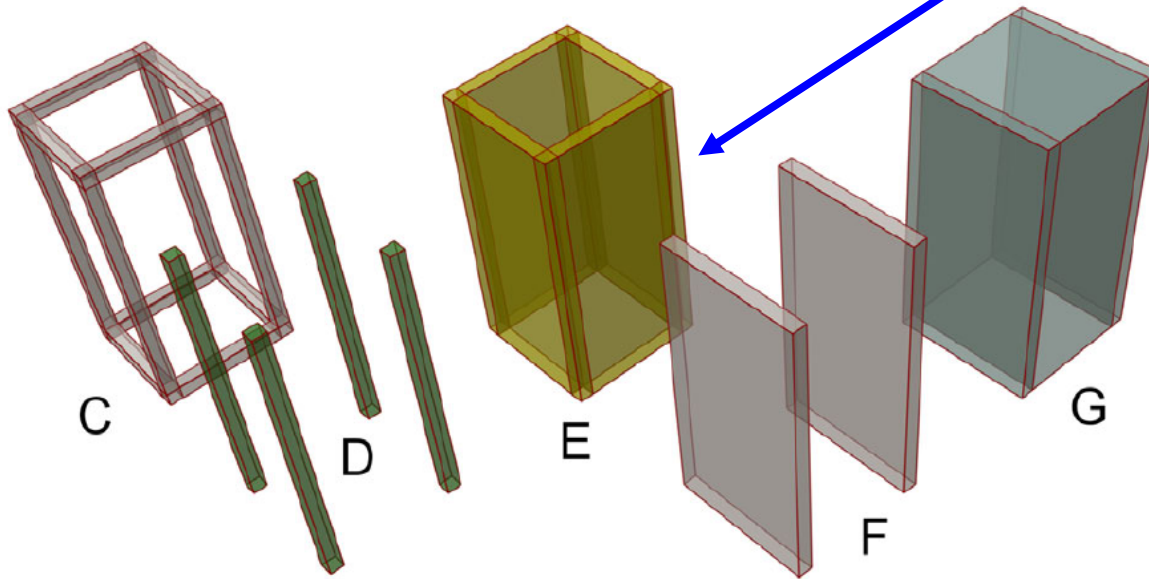


# Shape Grammar in GML

ShapeGrammar.Tools.init

```
/A { terminal-box } def  
/B { terminal-void } def  
/C { [ 0.1 -1 0.1 ] /X split E D E } def  
/D { [ 0.1 -1 0.1 ] /Y split F B F } def  
/E { [ 0.1 -1 0.1 ] /Y split G F G } def  
/F { [ 0.1 -1 0.1 ] /Z split A B A } def  
/G { [ 0.1 -1 0.1 ] /Z split A A A } def
```

scope (0,0,-2) move (2,1,1) scale C  
finish



# Shape Grammar in GML

- Parameters determined by recognition

```
1.2 !window-width
16 !wall-colour

/RightFacade (
    [2.85 0.15 3.0 3.0] /Y split-r
    FirstFloorRight
    0.15 3 Ledge
    2 :wall-colour 0 0.5 Floor ( :window-width :wall-colour -0.625 -0.125 0.04 WindowTileSill Window WindowSill ) repeat
    2 :wall-colour 0 0.5 Floor ( :window-width :wall-colour -0.625 -0.125 0.04 WindowTileSill Window WindowSill ) repeat
) def

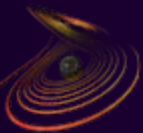
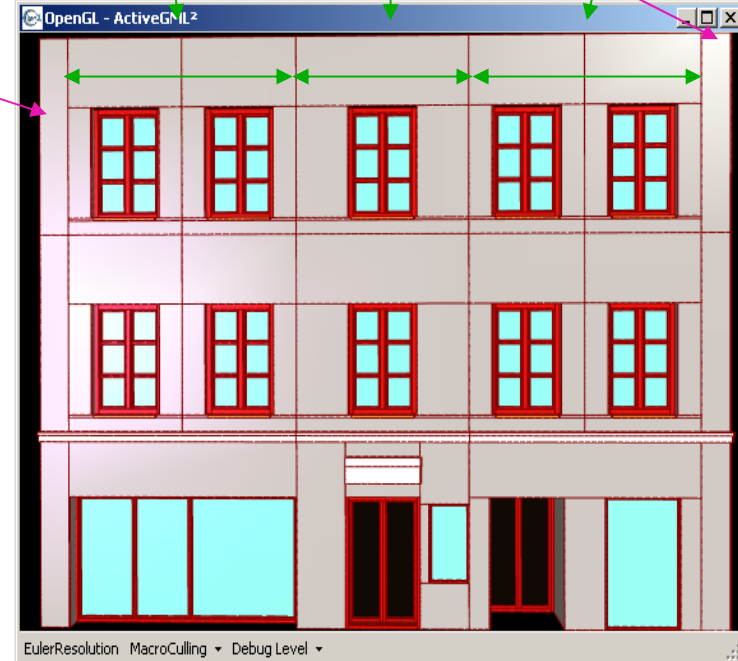
/CenterFacade (
    [2.85 0.15 3.0 3.0] /Y split-r
    FirstFloorCenter
    0.15 3 Ledge
    1 :wall-colour 0 0 Floor ( :window-width :wall-colour -0.625 -0.125 0.04 WindowTileSill Window WindowSill ) repeat
    1 :wall-colour 0 0 Floor ( :window-width :wall-colour -0.625 -0.125 0.04 WindowTileSill Window WindowSill ) repeat
) def

/LeftFacade (
    [2.85 0.15 3.0 3.0] /Y split-r
    FirstFloorLeft
    0.15 3 Ledge
    2 :wall-colour 0.5 0 Floor ( :window-width :wall-colour -0.625 -0.125 0.04 WindowTileSill Window WindowSill ) repeat
    2 :wall-colour 0.5 0 Floor ( :window-width :wall-colour -0.625 -0.125 0.04 WindowTileSill Window WindowSill ) repeat
) def

/WindowSill (
    [-1 1.08 -1] /X split-r
    W
    0.04 Jut 7 WM
    W
) def

/Window (
    0.08 Deepening
    0.08 4 Frame
    [-1 0.08 -1] /X split-r
    -3 /Y subdivide pop
    WindowElement
    WindowElement
    WindowElement
    4 WM
    -3 /Y subdivide pop
    WindowElement
    WindowElement
    WindowElement
) def

/WindowElement (
    0.02 Deepening 0.03 4 Frame 0.02 Deepening 5 WM
) def
```

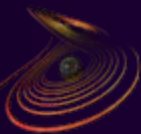


# Shape Grammar Templates

$F_3 \rightarrow A_3 B_3^* A_3'$   
 $F_2 \rightarrow A_2 B_2^* A_2'$   
 $F_1 \rightarrow A_1 B_1^* A_1'$

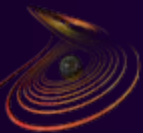
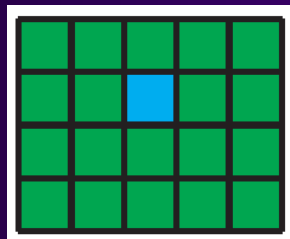
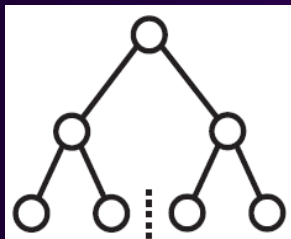
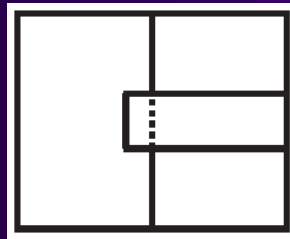
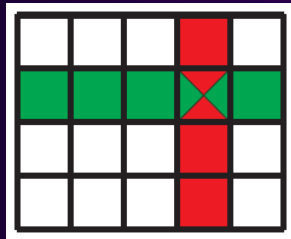
\* = Several Times

' = Mirror



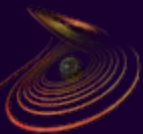
# Facade Case Study

- First Floor Different
- Center Structures
- Symmetries
- Result of Analysis:  
4 common problems



# Summary & Future Work

- CityFit Workflow
- 2D and 3D Input Data
- Shape Grammar Based on GML
- Convex Polyhedra
- Facade Analysis → Grammar Templates
- Future Work
  - ❖ Idea: Grammar Templates to guide Recognition
  - ❖ Automatization
  - ❖ Fitting

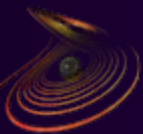


# References

- [1] L. Zebedin, A. Klaus, B. Gruber Geymayer and K. Karner. *Towards 3d map generation from digital aerial images*. ISPRS 2006
- [2] P. Müller, P. Wonka, P. Haegler and L. Van Gool. *Procedural Modeling of Buildings*. SIGGRAPH 2006
- [3] CityEngine, Procedural Inc.

## Thank you for your attention!

[www.cg.v.tugraz.at/cityfit](http://www.cg.v.tugraz.at/cityfit)  
[www.generative-modeling.org](http://www.generative-modeling.org)



# Shape Grammar Templates

$F_n$	$\rightarrow$	$A_n$	$B_n^*$	$A_n'$
$(F_i$	$\rightarrow$	$A_i$	$B_i^*$	$A_i')$
$F_1$	$\rightarrow$	$A_1$	$B_1^*$	$A_1'$
$F_0$	$\rightarrow$	$A_0$	$B_0^*$	$A_0'$

