

• General Info

DeCodingSpaces Components is a toolset with computational methods for analysing and generating architectural and urban space for Grasshopper(parametric modeling - PlugIn for Rhino).

This first release supports segment maps (as known from Space Syntax) as an analysis type and provides the calculation and visulisation of two graph-measures (Betweenness, aka Choice and Closeness aka Integration) plus small set of useful additional tool for parametric urban modelling (Modeling Components).

We would be glad if you share your experiences with this toolset. In case of any questions you can contact us:
info@decodingspaces.de

The toolset is still work in progress... further analysis types and modeling tools are coming soon

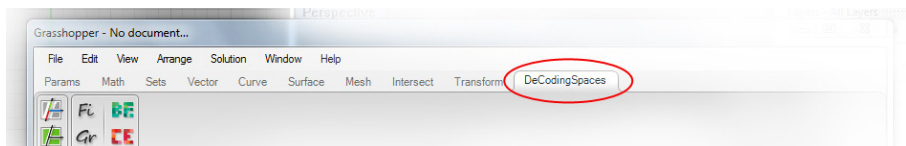
• Requirements

Rhino 5.0

Grasshopper for Rhino version 9.0006 or higher

• Installation

1. Extract *DeCodingSpacesComponents.zip* file.
2. Copy extracted files into ...\\Rhinceros 4.0\\Plug-ins\\Grasshopper\\Components\\
3. Start Grasshopper and the new components should appear under the tab called "DeCodingSpaces"



• General hints

This help file is not explaining how Grasshopper works or what does Spatial analysis mean!

1. If you are not sure about you geometry, use Filter Component
2. If you don't see colour results from your analysis into your Rhino viewport, probably some other component is drawing the same geometry. If it is the case, just turn off its preview.
3. Grasshopper is not optimised for making complex calculations. Graph Analysis are very resource-intensive, especially if the number of segments exceeds 1000, Grasshopper can crash. (depends on the performance of your computer). When analysing very complex geometry, we recommend to only calculate one graph-measure at the same time.
4. It is a beta-version, so please be patient :)

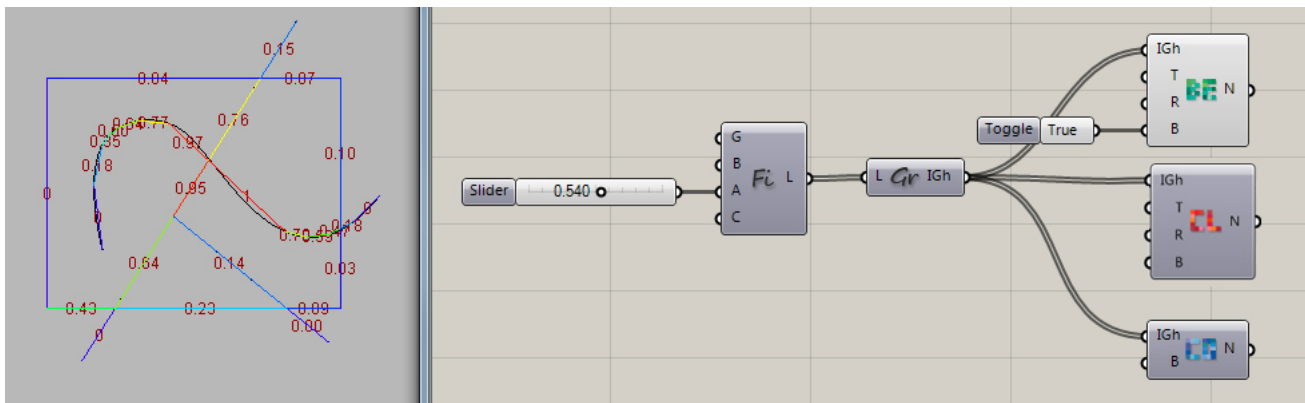
• Links

<http://infar.architektur.uni-weimar.de/service/drupal-cms/node/545>
http://www.slideboom.com/presentations/292558/Intro-to-Space-Syntax_Day-1
http://www.slideboom.com/presentations/293659/Intro-to-Space-Syntax_Day-2
<http://www.grasshopper3d.com>

- **Package content**

Analysis Components

Basic Example:



1. Filter - Component

This component creates a segment map from a given geometry. It converts curves to polylines, creates segments between intersections and eliminates identical elements. It ensures that any line-structure you input is ready for analysis.

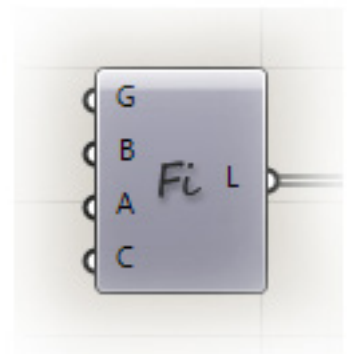
Important hints: This component is optional, if your geometry has been drawn correctly as a segment map, you don't have to use it. But, since it enables one to draw freely without checking for segment-consistency, we highly recommend its use. All geometrical elements should be in the same branch. If you are not getting strange results, try to flatten the input parameter G.

Inputs:

- G:** Geometry (Lines, Polylines, Nurbs Curves)
- B:** ConvertCurve (Boolean). If True, Curves will be converted to polylines according to Angle between each segment. If False, Curves will be filtered out (default = true).
- A:** Angle for curve conversion in radians (Number, default = 0.2).
- C:** Segments (Boolean). If True, geometry will be converted to segments. (default = true)

Output:

- L:** Segment Map (Lines).



2. Graph - Component

Converts a Segment Map to a Graph representation. Segments are interpreted as nodes and the connections between segments as edges. This component is central to all further steps, since all analysis-measures are derived from this graph.

Inputs:

- L:** Segment Map (Lines).

Output:

- IGh:** Graph representation of the segment map (This output is not usable in other grasshopper components)



3. Betweenness - Component

Calculates the the graph measure “betweenness centrality” for each segment. In Space Syntax terminology this measure is known as “choice”.

Important hint: Calculation time depends on the number of segments and its growing exponential, so it is recommended to reduce the number of analysed segments to a necessary minimum. And: **Save your work before plug in or enabling this component!**

Inputs:

IGh: Graph representation of the segment map (from Graph Component)

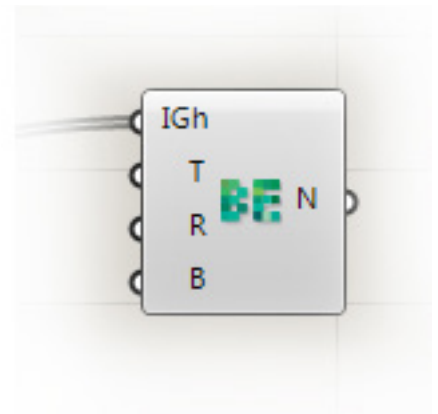
T: Type (Number (def. = 0). Control how the shortest paths are calculated. If Type = 0, angle between segments is considered. If Type = 1, distance is considered.

R: Radius (Number (def. = 0). If Radius > 0, given Radius will be used for local Analysis. If Radius = 0, global analysis will be computed.

B: Show (Boolean (def. = False). Shows or hide numerical results of analysis in Rhino viewport.

Output:

N: Normalized analysis results in range from 0 to 1.



4. Closeness - Component

Calculates the the graph measure “closeness centrality” for each segment. In Space Syntax terminology this measure is known as “integration”.

Important hint: Calculation time depends on the number of segments and its growing exponential, so it is recommended to reduce the number of analysed segments to a necessary minimum. And: **Save your work before plug in or enabling this component!**

Inputs:

IGh: Graph representation of analysed geometry

T: Type (Number (def. = 0). Control how the shortest paths are calculated (later used for the final Closeness computation). If Type = 0, angle between segments is considered. If Type = 1, distance is considered.

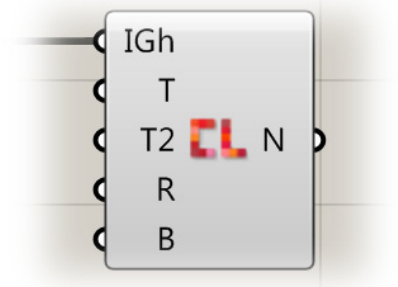
T2: Type (Number (def. = 0). Control Closeness is calculated (based on the shortest paths calculation). If Type = 0, metric is considered (According to v“T” settings, is it angle or distance). If Type = 1, steps are considered. segments is considered. If Type = 1, distance is considered.

R: Radius (Number (def. = 0). If Radius > 0, given Radius will be used for local Analysis. If Radius = 0, global analysis will be computed.

B: Show (Boolean (def. = False). Shows or hide numerical results of analysis in Rhino viewport.

Output:

N: Normalized analysis results in range from 0 to 1.



5. Connectivity - Component

Calculates the Connectivity of each segment map

Inputs:

IGh: Graph representation of the geometry

B: Show values (Boolean (def. False). Shows or hide numerical results of analysis in Rhino viewport.

Output:

N: Normalized analysis results in range from 0 to 1.



1. Extract Polygons - Component

Creates closed polygons from the Graph Representation of the analysed geometry.

Inputs:

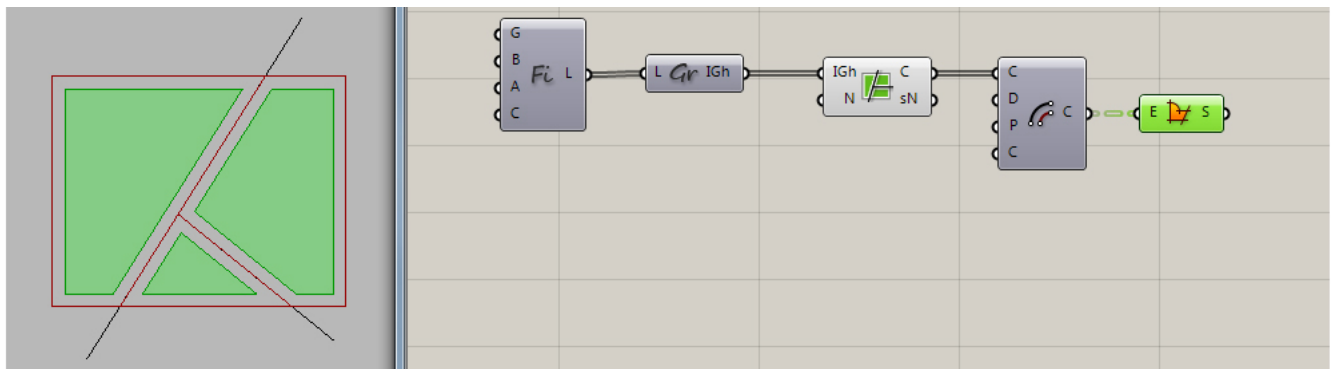
IGH: Graph representation of the analysed geometry N: (Optional field):
Numbers. Synchronize the results of analysis(Betweenness, Closeness or
Conectivity component output) with edges of Closed Polygons(output C:)

Output:

C: Closed Polygons

sN: Synchronised analysis results.(Use it as input for custom offset)

Basic Example:



2. Custom Offset - Component

Offset each edge a polygon () according to its own Sn-Value (eg. Closeness, Betweenness).

Important hint: Use flattened lists for both inputs. You should get 1:1 lists if you compare flattened list of exploded polygons and synchronised Values list.

Inputs:

C: Closed Polygons (e.g. from Extract Polygons - Component)

sN: List of numbers witch represent offset distance of each polygon edge.

Output:

C: Polygons (offsetted)

Basic Example:

