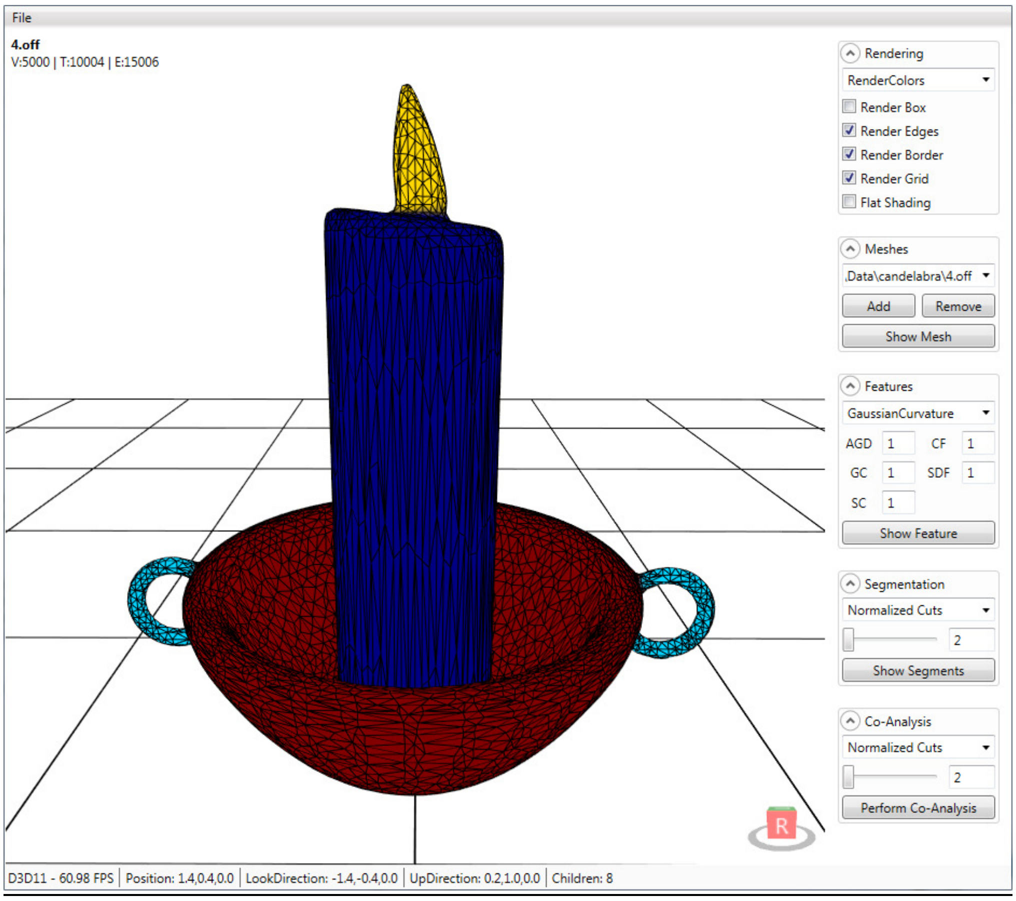
Co-Analysis Documentation

Name: Kurt Leimer
Matrikelnummer: 0825842
Kennzahl: 066 932
LVA: Praktikum aus Visual Computing
Technische Universität Wien, WS 2014/15



4.off
V:5000 | T:10004 | E:15006

RenderColors
Render Box [ ]
Render Edges [✓]
Render Border [✓]
Render Grid [✓]
Flat Shading [ ]

.Data\candelabra\4.off

GaussianCurvature
AGD 1 CF 1
GC 1 SDF 1
SC 1

Normalized Cuts 2

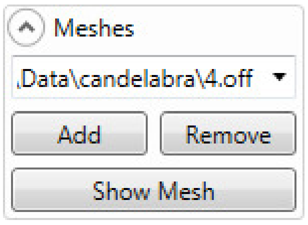
Normalized Cuts 2

D3D11 - 60.98 FPS | Position: 1.4,0.4,0.0 | LookDirection: -1.4,-0.4,0.0 | UpDirection: 0.2,1.0,0.0 | Children: 8

This is the documentation for the *MeshSegmentationViewerDX* application (revision 1861, 10.12.2014) and the *CoAnalysis* library (revision 1922, 22.12.2014). The application serves as a framework for Co-Analysis, in which a set of meshes of a specific category (e.g. candelabra or chairs) is analysed in an attempt to group the parts constituting each mesh into semantic categories. This is achieved by first individually subdividing each mesh into smaller segments and then clustering the segments of all meshes based on their face-level features. In the example pictured above, the mesh parts are grouped into 4 distinct categories, each represented by a different color. Furthermore, the application allows the selection of different algorithms for each step of the Co-Analysis pipeline and can also be easily extended by adding more algorithms. The Co-Analysis pipeline is divided into 4 steps, each explained in the following paragraphs:

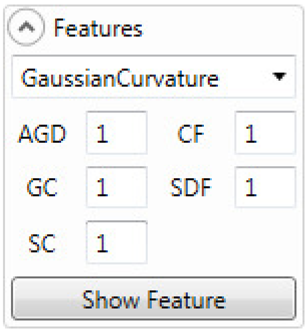
1. Mesh selection
2. Feature computation
3. Individual mesh segmentation
4. Segment clustering

1. Mesh selection



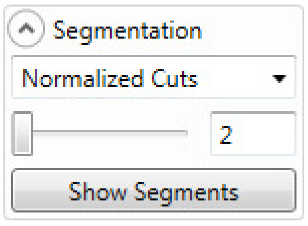
The application can load multiple meshes at once, although only one mesh is displayed at a time. The Add button allows the user to add more meshes to the list of meshes, while the Remove button removes the mesh that is currently selected from the list. The Show Mesh button will display the selected mesh with its segment colors if they are available or in the default coloring if they are not.

2. Feature computation



In this step various features are computed for each face of every mesh. The available features are Average Geodesic Distance (AGD) [3], Conformal Factor (CF) [2], Gaussian Curvature (GC) [2], Sphere Diameter Function (SDF) [7] and Shape Contexts (SC) [1,4]. The text boxes allow the user to specify different weights for the features. Furthermore, some features can be visualized as the coloring of the mesh by selecting the feature from the drop-down list and pressing the Show Feature button. All methods for calculating features are contained in the *FeatureDescriptor* class.

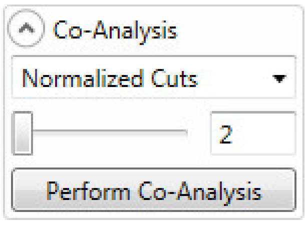
3. Individual mesh segmentation



In this step each mesh is individually divided into smaller segments. The desired number of segments can be specified by the user. Various methods are available for this step and more can be created by implementing the *ISegmentationAlgorithm* interface. The segmentation created by this step can be displayed by pressing the Show Segmentation button.

- Normalized Cuts
    - Based on the approach by Toony et al. [9]. The dihedral angle between adjacent faces is used to create a similarity matrix which is then used to cluster faces with the Normalized Cuts [8] algorithm. It is implemented in the *NcutSegmentation* class.
- Hierarchical Normalized Cuts
    - Like the Normalized Cuts method, except that each mesh segment is recursively subdivided into two segments until the desired number of segments is reached. It is implemented in the *NcutHierarchicalSegmentation* class.
- Load from SEG-file
    - Loads a SEG-file that contains the segment index for each face of the mesh. The number of segments is thus determined automatically. It is implemented in the *SegFileSegmentation* class.
- Load from SEG-file (split disconnected)
    - Loads a SEG-file that contains the segment index for each face of the mesh, but disconnected parts of the same segment are split into separate segments. The number of segments is thus determined automatically. It is implemented in the *SegFileSplitSegmentation* class.
- Split disconnected parts
    - This method treats every disconnected part of the mesh as a separate segment. The number of segments is thus determined automatically. It is implemented in the *SplitDisconnectedSegmentation* class.

4. Segment clustering


Co-Analysis

Normalized Cuts

2

In this step the segments of all meshes are clustered to determine segments that are geometrically or structurally similar. The number of clusters can be specified by the user. The available methods of this step are implementing the *IClusteringAlgorithm* interface. By pressing the Perform Co-Analysis button, the entire Co-Analysis algorithm is executed using the settings specified in the control elements of each step of the algorithm.

- Normalized Cuts
    - Based on the approach by Meng et al. [6]. First, the distribution of features in each segment is captured by a histogram for each type of feature. A similarity matrix is created by computing the distances between the histograms of each pair of segments. The matrix is then used to cluster the segments with the Normalized Cuts [8] algorithm. This method is implemented in the *NcutClustering* class.
- Laga Method
    - Based on the approach by Laga et al. [5]. Each mesh is represented by a graph where the nodes correspond to segments or unions of segments and edges correspond to a type of structural relationship (adjacency, symmetry or enclosure). The similarity between two segments is obtained by computing both the geometrical and the structural similarity of two nodes. The

geometrical similarity is computed in the same way as in the previous method. The structural similarity is computed by comparing graph walks of a specific length starting at the node corresponding to the segment. The similarities are then used to create a similarity matrix and to compute a clustering with the Normalized Cuts [8] algorithm. This method is implemented in the *LagaClustering* class.

5. Miscellaneous

Some additional utility classes are also used in the algorithm. The *Histogram1D* class represents a one-dimensional histogram used to capture the feature distribution in the clustering stage. The *HistogramShell* class represents a shell histogram and is used to calculate the Shape Contexts feature. The *MeshGraphNode* and *MeshGraphEdge* classes are used in the Laga Method to represent each mesh as a graph. More detailed documentation for each class and function can be found in the source code. The example meshes provided are taken from the COSEG (http://web.siat.ac.cn/~yunhai/ssl/ssd.htm) and Princeton (http://segeval.cs.princeton.edu/) datasets.

6. References

[1] S. Belongie, J. Malik and J. Puzicha. *Shape matching and object recognition using shape contexts*. IEEE Transactions on Pattern Analysis and Machine Intelligence 24, no. 4, pages 509-522, 2002.

[2] M. Ben-Chen and C. Gotsman. *Characterizing Shape Using Conformal Factors*. 3DOR, pages 1-8, 2008.

[3] M. Hilaga, Y. Shinagawa, T. Kohmura and T. L. Kunii. *Topology matching for fully automatic similarity estimation of 3D shapes*. Proceedings of the 28th annual conference on Computer graphics and interactive techniques, pages 203-212, ACM, 2001.

[4] M. Körtgen, G.-J. Park, M. Novotni and R. Klein. *3D shape matching with 3D shape contexts*. The 7th central European seminar on computer graphics, Volume 3, pages 5-17, 2003.

[5] H. Laga, M. Mortara and M. Spagnuolo. *Geometry and context for semantic correspondences and functionality recognition in man-made 3d shapes*. ACM Transactions on Graphics (TOG) 32, Volume 5, page 150, 2013.

[6] M. Meng, J. Xia, J. Luo and Y. He. *Unsupervised co-segmentation for 3D shapes using iterative multi-label optimization*. Computer-Aided Design 45, no. 2, pages 312-320, 2013.

[7] L. Shapira, A. Shamir and D. Cohen-Or. *Consistent mesh partitioning and skeletonisation using the shape diameter function*. The Visual Computer 24, no. 4, pages 249-259, 2008.

[8] J. Shi and J. Malik. *Normalized cuts and image segmentation*. IEEE Transactions on Pattern Analysis and Machine Intelligence 22, Volume 8, pages 888-905, 2000.

[9] Z. Toony, D. Laurendeau, P. Giguère and C. Gagné. *3D-NCuts: Adapting Normalized Cuts to 3D Triangulated Surface Segmentation*. Proceedings of the 9th International Conference on Computer Graphics Theory and Applications (GRAPP), 2004.