Variational Multilevel Mesh Clustering
Iurie Chiosoa, Andreas Kolb
Institute for Vision and Graphics
University of Siegen, Germany

ABSTRACT
In this paper a novel clustering algorithm is proposed, namely Variational Multilevel Mesh Clustering (VMLC). The algorithm incorporates the advantages of both hierarchical and variational (Lloyd) algorithms, i.e. the initial number of seeds is not predefined and on each level the obtained clustering configuration is quasi-optimal. The algorithm performs a complete mesh analysis regarding the underlying energy functional. Thus, an optimized multilevel clustering is built.

The first benefit of this approach is that it resolves the inherent problems of variational algorithms, for which the result and the convergence is strictly related to the initial number and selection of seeds. On the other hand, the greedy nature of hierarchical approaches, i.e. the non-optimal shape of the clusters in the hierarchy, is solved. We present an optimized implementation based on an incremental data structure. We demonstrate the generic nature of our approach by applying it for the generation of optimized multilevel Centroidal Voronoi Diagrams and planar mesh approximation.

Index Terms: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Geometric algorithms; I.5.3 [Pattern Recognition]: Clustering—Algorithms

1 INTRODUCTION
Due to technological advances, 3D model acquisition or generation systems become faster and more precise. Thus tools for fast and reliable geometry processing are required. In most cases the models are represented by a polygonal surface mesh, i.e. not only geometry information but also connectivity (topology) information is available. For clustering this kind of shape representation, there are two major classes of methods which are frequently employed: Hierarchical and Variational.

Hierarchical methods produce a hierarchy, i.e. a binary tree, of clusters. Each cluster represents a set of faces that aggregates a specific criteria. The main advantage of such an approach is that it does not require any additional parameters or any intervention from the user. Despite the simplicity and wide range of applications, hierarchical clustering is yet a greedy approach, i.e. an assigned face can no further be reassigned to other clusters although that may result in a more appropriate configuration. This drawback limits the applicability of hierarchical clustering in many situations, e.g. generating Centroidal Voronoi Diagram (CVD) for mesh coarsening. Applying a greedy hierarchical approach on base of a CVD does not yield a valid solution, i.e. the result is not a CVD.

In contrast, variational methods do provide an optimal clustering (in the sense that at least a local minimum is reached) for an a-priori specified number of clusters. The determination of an appropriate number of clusters involves in general manual user intervention. Additionally, the choice of initial seeds, i.e. starting positions and starting representatives, affects the convergence and the final result. Usually a random initialization is used, thus two consecutive executions of the algorithm produce in general different results. In [2] some heuristics like incremental insertion and deletion of regions and region teleportation were proposed. Despite their efficiency, it is still an open question when and where to apply them.

In this paper we propose a novel algorithm which incorporates the advantages of both methods, i.e. hierarchical and variational, and consequently overcomes the drawbacks of both. The result of the algorithm is an optimized multilevel clustering. The algorithm neither uses any heuristics to obtain the final result nor a-priori user specified parameters. Since a brute-force combination of variational and hierarchical methods would be rather expensive regarding computational effort and storage requirements, we provide a very efficient implementation and an optimized data structure which allow a fast and complete mesh analysis.

We also show the generic nature of this algorithm by applying it to different tasks, i.e. planar fitting and the generation of a multilevel Centroidal Voronoi Diagram (MLCVD).

The paper is organized as follows: After discussing related work (Sec. 2), some notations and definitions are provided in Sec. 3, followed by the algorithm overview (Sec. 4). Sec. 5 presents the energy minimization algorithm. In Sec. 6 a detailed description of the algorithm is provided. Sections 7 and 8 present results of the proposed algorithm and draw some final conclusions.

2 RELATED WORK
In this section we give an overview of approaches which constitute building blocks for our algorithm. For a more detailed description of the first two algorithms see for example [8].

2.1 Hierarchical Face Clustering (HFC)
HFC was introduced by Garland et al. [3] for planar approximation of polygonal meshes. The algorithm starts by creating a dual graph of the mesh, i.e. each face is assigned to a node in the dual graph. A dual edge (DE) between two nodes is created if the corresponding mesh faces are adjacent. Using a priority queue (PQ) all created DE are sorted according to their contraction cost. At each step, a DE with highest priority is popped from the queue and collapsed. Additionally, the PQ is updated for all DEs incident to the newly created node. In the contraction operation corresponding nodes of the DE are merged into a new node, i.e. two clusters are merged into one representative cluster. As a result one obtains a hierarchy (a binary tree) of clusters.

In [1] Attene et al. employed a HFC technique for fitting a family of primitives such as planes, spheres and cylinders. In that case the cost of collapsing a DE is the minimum of the approximation errors computed against all primitives.

2.2 Variational Clustering (VC)
Cohen-Steiner et al. [2] presented VC as an extension of Lloyd’s algorithm for piecewise approximation of the input polygonal surface. The algorithm works in two phases: partitioning and fitting, both repeated alternately to minimize the total energy, i.e. the approximation error. A notion of a shape proxy is introduced in order to describe the region’s local representative. In the partitioning step
a global PQ is used to assign each triangle to the best fitting proxy. As a result, a new clustering of the surface is obtained. In the fitting step for each cluster the proxy is re-computed in order to find the best representative for their associated faces. The algorithm is executed until some convergence criteria is met or a specified number of iterations is executed. For a user specified number of clusters \( k \), the algorithm provides a \( k \) partitioning of the input mesh.

In [2] the algorithm was applied for the extraction of planar regions. Different works like [11] and [12] extended this approach to other proxy shapes like spheres, cylinders, rolling-balls and general quadric. In [5] the algorithm was adapted for identifying quasi-developable surfaces. Though an extension of Lloyd’s algorithm (sometimes referred as \( k\)-means) and requires a user specified number of clusters and, in general, only a local optimum is reached. To obtain a better clustering results, i.e. a further reduction of the energy, heuristics such as incremental insertion and deletion of proxies, or region teleportation can be used [2]. In the context of data clustering, there were attempts to overcome the problem of choosing the number of clusters for \( k\)-means [7] or to obtain better clustering results (see [4] for an overview and comparison).

2.3 Approximated Centroidal Voronoi Diagram (CVD)  
Vallette and Chassery [9] introduced a new algorithm for creating an approximated CVD for uniform mesh coarsening. In [10] this algorithm was extended to adaptive mesh coarsening. The main idea is to minimize the CVD energy functional \( E \) by iteratively updating the clusters on the boundaries between clusters, which leads to a very efficient algorithm. For each boundary edge between two clusters \( C_i \) and \( C_j \), a local test is performed, i.e. the energies for three cases are computed: \( E_{ini} \) (initial configuration), \( E_1 \) \((C_i \text{ grows and } C_j \text{ shrinks})\), \( E_2 \) \((C_i \text{ shrinks and } C_j \text{ grows})\). The case with the smallest energy is chosen and the cluster configuration is updated accordingly. Thus, the energy functional is iteratively decreased and the final clustering is obtained when no further energy reduction is achieved.

The main advantage of this approach over the classical variational (Lloyd) method is that it solves the same problem, i.e. constructing an approximated CVD, in a more efficient manner. It does not require any global priority queue and there is no explicit proxy fitting step in the algorithm. We will use this algorithmic concept in our algorithm (see Sec. 5.1) and we will show in Sec. 6.5 that this kind of algorithm can be extended to other energy functionals.

3 NOTATIONS AND DEFINITIONS

Notations: Suppose that an input polygonal mesh \( M \) with \( m \) faces \( F \) is provided. A halfedge data structure [6] is used for mesh representation. Let \( F \) denote the set of all faces of \( M \). Clustering \( M \) into \( k \) non-overlapping clusters \( C_i \) means that each cluster \( C_i \) consists of a union of \( n_i \) mesh faces \( F_l \).

Definition 1. A boundary loop (BL) is a closed sequence of all boundary half edges of a 1-connected set of faces.

In Fig. 2 the BL is represented by a dashed line. If a cluster \( C_i \) has more than one component, then each component will have its own boundary loop. Any change in the cluster configuration changes the BLs of the affected clusters (see Fig. 2 (a)-(c)).

Definition 2. A dual edge (DE) between two clusters \( C_i \) and \( C_j \) can be created if and only if BLs of both clusters share at least one common edge.

A collapse of a DE can be seen as a merging of two corresponding clusters into one representative cluster (see Fig. 1 (a)-(b)).

4 OVERVIEW OF THE ALGORITHM

We aim at building a multilevel clustering where for each level \( l \) the obtained clustering is optimized w.r.t. an underlying energy functional.

The proposed algorithm can be summarized by the following three steps:

Initialization: Each mesh face is assigned to an individual cluster, i.e. the total number of clusters \( k \) is equal to \( m \).

Merging: Identify and collapse a dual edge with the smallest cost (see Sec. 6.2). Here the total number of clusters is decreased by one.

Optimization: The newly obtained configuration is optimized (see Fig. 1 (b)-(c)), i.e. the total energy is minimized. The Energy Minimization by Local Queries (EMLQ) algorithm (see Sec. 5.1) is applied in this case.

After the initialization, the algorithm iteratively executes the merging and optimization steps until the final number of clusters is equal to one. This way an optimized multilevel clustering is built which fulfills the proposed objective. Fig. 1 shows an example of the two main stages of our algorithm.

This approach can be seen as a complete mesh analysis. For each level, i.e. each specific number of clusters, an optimized instead of a greedy clustering is provided. The initialization problem of variational clustering is also solved, because the merging of two clusters is always done according to the least energy cost. The following optimization step will provide a better initial configuration for the next level.

In general, the VMLC algorithm does not generate a nested hierarchy, i.e. the lower level clusters are not completely contained in a single upper level cluster as in the case of the HFC algorithm. The multilevel construction is a nested hierarchy only when no optimization takes place. Thus, in general navigating between different levels of the multilevel representation is a non-trivial task. We show in Sec. 6.4 that our proposed data structure provides an easy way of navigating between levels.

Here we should emphasize the versatility of our algorithm. If the optimization step is deactivated the algorithm will perform exactly as a hierarchical face clustering (HFC). Vice versa if the level \( l \) is fixed, i.e. the number of clusters is fixed, one performs a variational optimization (VC).

The approach of combining hierarchical and variational methods has several challenges, which will be addressed in Sec. 6.

Efficiency: The variational clustering is an iterative approach where partitioning and fitting steps are alternated until an optimal partitioning regarding an energy functional is obtained. Using this approach in the optimization step is computationally too exhaustive.

Using variational clustering in conjunction with hierarchical clustering will require, in general, two priority queues: one for all DEs and the second one for the priority of faces in the partitioning step.

Validity of dual edges: some DEs may get invalid (see Fig. 1(b)-(c)) or new DEs appear after optimization (see Fig. 1(c)). This is in contrast to the standard HFC algorithm, where only DEs incident to a newly created node need to be updated. Hence, special care must be taken for a correct update.

Tracking of cluster faces: applying the optimization step after each dual edge collapse will reassign some of the faces to different clusters. Thus, all such changes must be tracked and represented in an appropriate data structure.

5 CLUSTER OPTIMIZATION AND DUAL EDGE COST

For cluster optimization we require a more efficient optimization approach compared to variational clustering. Thus we propose to...
Figure 1: A single step of the algorithm. (a) Initial configuration, the solid line represents the Dual Edge (DE) to be collapsed. (b) The new configuration after the DE was collapsed. The solid lines represent DEs which will not be valid after the optimization step. (c) The resulting configuration after the optimization step. The solid lines represent newly created DEs.

use a technique that is more suitable for that, i.e. Energy Minimization by Local Queries (EMLQ). We also describe how to compute the merging cost of a dual edge for a given energy functional.

5.1 Energy Minimization by Local Queries (EMLQ)

This approach is a generalization of the approximative CVD algorithm proposed by Vallette and Chassery [9].

Suppose that an energy functional $E$ is provided:

$$ E = \sum_{i \in [0, \ldots, k-1]} E_i = \sum_{i \in [0, \ldots, k-1]} \sum_{F_j \in C_i} E^i_j. $$

$E^i_j$ is the positive semi-definite cost of assigning the face $F_j$ to the cluster $C_i$. Note, that in general $E^i_j \neq E^i_m$ if $i \neq m$, e.g. see the CVD energy functional in Sec. 6.5.

The value of $E$ in Eq. (1) depends only on a given clustering of the mesh $M$ into $k$ clusters $C_i$, where each cluster $C_i$ contains a set of faces $\{F_j\}$. Thus the process of minimizing the energy $E$, i.e. the cluster optimization, can be seen as a clustering problem in which the mesh faces are reassigned to different clusters in such a way that the energy functional $E$ is minimized.

Given an initial configuration, the algorithm iteratively reduces the energy functional $E$ as follows:

- Loop until no configuration changes
  - For all clusters $C_i$
    - For all boundary edges $e \in b$
      - Compute energies $E^0, E^1, E^2$
      - Choose case with smallest energy
      - Update cluster configuration

Where the energies $E^0, E^1$ and $E^2$ (see Fig. 2(a)-(c)) are defined as follows:

$$ E^0 = \sum_{F_j \in C_q} E^q_j + \sum_{F_j \in C_p} E^p_j, $$

$$ E^1 = \sum_{i} E_i + \sum_{F_j \in C_q \cup \{F_i\}} E^q_j + \sum_{F_j \in C_p \setminus \{F_i\}} E^p_j, $$

$$ E^2 = \sum_{i} E_i + \sum_{F_j \in C_q \setminus \{F_i\}} E^q_j + \sum_{F_j \in C_p \cup \{F_i\}} E^p_j. $$

with $i \in \{0, \ldots, k-1\} \setminus \{q, p\}$

Because the energy functional $E$ is supposed to be positive semi-definite and any modification in the cluster’s configuration reduces $E$, the algorithm converges. As a result an optimized clustering is obtained for which the functional $E$ is minimal. In general, there is no guarantee that the global minimum will be reached.

Observe that the first term is irrelevant when comparing $E^0, E^1$ and $E^2$. Thus the three energies to be compared can be simplified:

$$ D^0 = \sum_{F_j \in C_q} E^q_j + \sum_{F_j \in C_p} E^p_j, $$

$$ D^1 = \sum_{F_j \in C_q \setminus \{F_i\}} E^q_j + \sum_{F_j \in C_p \setminus \{F_i\}} E^p_j, $$

$$ D^2 = \sum_{F_j \in C_q \setminus \{F_i\}} E^q_j + \sum_{F_j \in C_p \setminus \{F_i\}} E^p_j. $$

The energy minimization is done only by applying local queries on the $BL$. For each edge $e \in BL$ the query is done only for two adjacent clusters and based only on this information the cluster configuration is changed, i.e. functional $E$ is minimized. That makes the algorithm very fast and efficient compared to the standard VC optimization.

Taking this algorithm by itself, it requires the initial number of seeds to be specified, just as variational clustering. The convergence and the final result is also strictly related to the starting configuration, i.e. the initial cluster configuration.

5.2 Dual Edge Cost

The cost of merging the clusters $C_q$ and $C_p$ into one representative cluster $C_r$, i.e. the collapse cost of the DE which connects $C_q$ and $C_p$, are defined as:

$$ Cost = \sum_{F_j \in (C_q, C_p)} E^q_j - \left( \sum_{F_j \in C_q} E^q_j + \sum_{F_j \in C_p} E^p_j \right). $$

Observe that, this cost definition is different from the one used in [1], where only the cost for the merged cluster, i.e. the first term, is used. However, our cost definition ensures that the collapse operation is done only for clusters which will give the smallest increase in the total energy $E$, i.e. the selected collapse has the least negative impact on the overall energy. Additionally, this cost definition leads to a very compact energy representation and makes our algorithm even more efficient (see Sec.6.5).

6 Variational Multilevel Clustering Algorithm

In this section we present a more detailed description of the algorithm outlined in Sec. 4. The standard HFC approach (see Sec. 2.1) starts with the creation of the dual graph (DG) of the mesh
In the initialization step each face 6.1 Initialization

6.2 Dual Edge Collapse

At this step an ODE with minimal cost is identified in the CA and to the opposite cluster

Optimal Dual Edge (ODE) (see Sec. 6.2) c.ODE=invalid, i.e. the ODE of this cluster is no longer valid due to some cluster configuration change or not determined, as in the initialization.

6.3 Optimization

The optimization is executed as repeated processing of all changed clusters. If the cluster c is marked as being changed, i.e. c.changed=true, then the cluster configuration needs to be optimized (see Sec. 5.1). After optimization we set c.changed=false and c.ODE=invalid for this cluster. Any change in the cluster configuration during energy minimization flags the opposite cluster as changed, thus in the next optimization loop these clusters are also handled. This process repeats until no cluster configuration change occurs, i.e. all clusters have c.changed=false. Remember, that the optimization always converges since the overall energy always decreases.

Observe that the energy minimization always starts with the cluster most affected by the collapse operation, i.e. cluster optimization is always applied in the merged cluster first, followed by a propagation of that change to the neighboring clusters.

After the optimization step the neighbor clusters of each cluster with c.ODE=invalid are also set as having non-valid ODE. This is required for a correct identification of an ODE with minimal cost for the neighboring clusters. Additionally, all changes in the clusters configurations are saved for multilevel representation (see Sec. 6.4).
After each optimization step, the information of faces which have moved from one cluster to another, i.e. the difference between two consecutive clusterings, is stored. For each such a face \( F \) we store a triplet \((a, b, c)\), where \( b \) is the ID of the cluster to which the face has moved and \( c \) is the ID of the initial cluster from which this face has been removed. Even though a face can be moved to different clusters during a full optimization step, we store only the final cluster ID. Fig. 3 shows an example of the differential data storage; here only the final situation after the complete optimization steps is stated.

To reconstruct level \( l \) from level \( l-1 \) using the stored information at level \( l \) and the differential data \( \{(a_1, b_1,c_1),(a_2, b_2,c_2),\ldots\} \), all faces with \( ID = d \) the cluster ID needs to be changed to \( b \). Taking the example in Fig. 3, going from step1 to step2 will assign the face with \( ID = 5 \) to the cluster with \( ID = 4 \). Vice versa, if going from \( l+1 \) to \( l \) all faces with \( ID = d^{l+1} \) the cluster ID is changed to \( ID = c^{l+1} \). Thus going from step4 to step3 in Fig. 3 requires to change the cluster ID to the \( ID = 1 \) of the face with \( ID = 1 \).

The core algorithm does actually not store any intermediate information like energy for any level in the multilevel representation. However, for some applications it may be important to have this kind of information to justify or compare individual cluster levels or to compute the cluster energy.

The energy of each level (see Sec. 6.5) can be computed using the stored triplets \((a, b, c)\). Suppose that each cluster \( C_j \) in the initialization step, i.e. the cluster consists of one face only, has some initial values \( C_j(\{m\}) \) for different measures \( m \) (see Tab. 1). The measures \( m \) may represent any cluster data such as the centroid \( \gamma \) for a CVD, the normal \( n \) for planar fitting or the area \( \rho \) (see Sec. 6.5). In this case, index \( a \) refers to the measure \( m_{ka} \) of the cluster \( C_j \). Index \( b \) indicates that the measure \( m_{ka} \) should be “inserted” into the cluster \( C_b \) and at the same time index \( c \) indicates that \( m_{ka} \) should be “removed” from the cluster \( C_c \). “Insertion” and “removal” may include various mathematical operations depending on the measure. For example for the cluster area the insertion operation is a simple addition and the removal operation a subtraction of the terms. A detailed example which refers to Fig. 3 is presented in Tab. 1.

It should be noted, that based on this principle arbitrary measure can be tracked throughout the multilevel construction.

### 6.4 VMLC Data Structure

In this section we describe a very fast and efficient differential data structure to store the multilevel representation. In general, this provides an easy way for reconstructing any level and for computing any additionally required cluster information using differential information.

6.4 VMLC Data Structure

After each optimization step, the information of faces which have moved from one cluster to another, i.e. the difference between two consecutive clusterings, is stored. For each such a face \( F^{moved} \) with \( ID = a \) we store a triplet \((a, b, c)\), where \( b \) is the ID of the cluster to which the face has moved and \( c \) is the ID of the initial cluster from which this face has been removed. Even though a face can be moved to different clusters during a full optimization step, we store only the final cluster ID.

For each step of the algorithm applied to a simple mesh, \( step_0 \) is the initial configuration, here the cluster ID corresponds to the face ID. \( step_7 \) is the final clustering.

Figure 3: An example of stored information for each step of the algorithm applied to a simple mesh. \( step_0 \) is the initial configuration, here the cluster ID corresponds to the face ID. \( step_7 \) is the final clustering.

### 6.5 Sample Energy Functionals

**Approximated CVD:** In [9] it has been shown that the energy functional for building an approximated Centroidal Voronoi Diagram (CVD) can be expressed as:

\[
E_{CVD} = \sum_{i=0}^{k-1} \left( \sum_{j \in C_i} \rho_j \| \gamma_j \|^2 - \frac{\| \sum_{j \in C_i} \rho_j \gamma_j \|^2}{\sum_{j \in C_i} \rho_j} \right). 
\]

where \( \gamma_j \) and \( \rho_j \) is the centroid and the weighted area of the face \( F_j \), respectively.

For the EMLQ approach (see Sec. 5.1) the initial configuration is represented by

\[
D_{CVD}^0 = -\frac{\| \sum_{j \in C_i} \rho_j \gamma_j \|^2}{\sum_{j \in C_i} \rho_j} - D_{CVD}^0. 
\]

where \( D_{CVD}^0 \) and \( D_{CVD}^0 \) are computed in a similar way (see Eq. (3), (4)).

The cost of a dual edge collapse is then computed as follows:

\[
Cost_{CVD} = -\frac{\| \sum_{j \in C_i} \rho_j \gamma_j \|^2}{\sum_{j \in C_i} \rho_j} - D_{CVD}^0. 
\]

Observe that this form of energy functional leads to a very fast and efficient energy computation. For each cluster only the values \( \sum_j \rho_j \gamma_j \) and \( \sum_j \rho_j \) need to be stored and a fast cluster update is possible in this case. For each face \( F_j \) the values of \( \rho_j \gamma_j \) and \( \rho_j \) are computed only once at the beginning, thus making the overall computational cost very low.

**Planar Approximation:** In [2] a new shape metric \( L^2 \) was introduced which in the discrete case can be written as:

\[
L^2 = \sum_{F_i \in C} \rho_i \| n_i - \bar{n} \|^2. 
\]

where \( n_i \) is the normal of the faces \( F_i \) and \( \bar{n} \) is the optimal proxy normal, i.e. the cluster normal in our case: \( \bar{n} = \sum_{i \in C} \rho_i \frac{n_i}{\| \sum_{j \in C} \rho_j \|} \).

In the following, we derive a very efficient formulation from Eq. (9) which can be used not only for the EMLQ algorithm but also for the classical VC approach. The overall energy is given by

\[
E_{Planar} = \sum_{i=0}^{k-1} \sum_{j \in C_i} \rho_j \| n_j - \bar{n} \| = \sum_{j \in C_i} \rho_j \| n_j \|. 
\]

Applying similar arguments as for Eq. (7), the energy of the initial configuration for planar cluster approximation is:

\[
D_{Planar}^0 = -\frac{\sum_{j \in C_i} \rho_j \| n_j \|}{\sum_{j \in C_i} \rho_j}. 
\]
The cost of a dual edge is then calculated as follows:

\[ \text{Cost}_{\text{planar}} = -\sum_{\mathbf{r}_j \in (C_i \cup C_j)} \rho_j \mathbf{n}_j - D^0_{\text{planar}}. \]  

The functional \( E_{\text{planar}} \) provides the same advantages as the CVD energy functional. For each cluster only the value \( \sum \rho_j \mathbf{n}_j \) needs to be stored. Again, these values can easily be updated and thus a fast energy computation is possible.

**Remark:** The EMLQ algorithm is more preferably to be used with energy functionals which can be represented in an incremental energy formulation as in Eq. (6) and Eq. (10), i.e. only local queries are required to compute the energy. This kind of representation results in an efficient simplification as in Eq. (7), (8) and Eq. (11), (12). Other energy functionals like the ones used in [1] require to store the cluster’s faces explicitly and to solve an eigen-system to compute the energies for each step. This is, in general, possible but very inefficient. In our further work we will address this limitation, e.g. by defining alternative energy functionals or by adapting the algorithm so that it supports existing energy functionals at acceptable computational cost.

### 6.6 Variations of VMLC algorithm

We already mentioned in Sec. 4 that our algorithm is very flexible, i.e. the merging and optimization steps can be altered in an arbitrary sequence or even deactivated if required. Thus, different variations of the algorithm are possible. As a result the user can choose between faster execution of the algorithm or higher quality or accuracy of the results.

At the beginning, when the clusters are composed of only one face the algorithm most likely merges only two by two of these single neighboring clusters (see in Fig. 3 step1 to step4). Thus, depending on the addressed problem, it may be adequate to skip the optimization step for the first \( p \% \) of the merging steps.

Based on this observation the following variation of VMLC has been implemented:

1. **Initialization step.** The initialization is done like in VMLC algorithm.
2. **Merging of \( p \% \) of clusters.** Apply only merging step, i.e. optimization deactivated, for \( p \% \) of clusters.

### 3. Apply VMLC. Continue with the standard VMLC algorithm.

Thus, after initialization in the merging step, sequentially for each valid cluster \( c \) in the CA an ODE is identified and directly collapsed if the c.ODE.opposite cluster was not already involved in another merging operation. This step is repeated until \( p \% \) of clusters are merged. Doing so allow us a further reduction of the computational cost (see Tab.2) without any substantial influence on the obtained results, e.g. in case of the CVD energy functional (see Fig. 8).

### 7 Results and discussion

Fig. 4 shows a comparison between the results of the hierarchical face clustering and of the proposed VMLC algorithm, applied to the Bunny model using the CVD energy functional. The VMLC algorithm provides a valid CVD construction at each level with well shaped clusters, HFC algorithm, in general, gives an invalid CVD, i.e. the centroids are not the cluster generators.

Fig. 5 shows the results of the planar clustering, i.e. using Eqs. (11), (12), applied to the Fandisk model. Observe, that for this model the results of both algorithms are approximatively similar. This is due to the fact that in regions with zero Gaussian curvature the hierarchical approach adequately merges regions in principal directions of zero curvature. In the case of nonzero Gaussian curvature this is no longer true; see for example Fig. 6 and Fig. 7. Observe that the VMLC algorithm provides a better planar approximation in these cases. For shapes with non-zero Gaussian curvature the VMLC algorithm will in general provide better results compared to the hierarchical clustering.

In Fig. 7 another clustering result for the horse model is presented, using the planar energy functional. Observe that in case of the VMLC algorithm the clusters are better shaped, resulting in a higher overall fitting quality.

In Fig. 8 we show the dependency between the total CVD energy (Eq. (6)) and the number of clusters for different clustering algorithms, applied to the Fandisk model. For the VC algorithm random initializations have been used. To obtain the energy variation limits, the algorithm was applied 50 times for the same number of clusters with different random seeds. The \( p \) factor refers to the algorithm proposed in Sec. 6.6, i.e. the first \( p \% \) of the merges have been applied without any optimization. The cluster range between 1k to 400 from the total range [13k; 1] is picked to show the general energy behavior. Observe that the energy of the VMLC algorithm...
Table 2: Clustering time for VMLC algorithm on a 2.21GHz AMD Athlon PC

<table>
<thead>
<tr>
<th>Model</th>
<th># Faces (input mesh)</th>
<th>Energy Functional</th>
<th>p %</th>
<th>Time (sec.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fandisk</td>
<td>13k</td>
<td>CVD</td>
<td>-</td>
<td>14</td>
</tr>
<tr>
<td>Fandisk</td>
<td>13k</td>
<td>CVD</td>
<td>50</td>
<td>3</td>
</tr>
<tr>
<td>Fandisk</td>
<td>13k</td>
<td>CVD</td>
<td>75</td>
<td>1</td>
</tr>
<tr>
<td>Fandisk</td>
<td>13k</td>
<td>CVD</td>
<td>90</td>
<td>1</td>
</tr>
<tr>
<td>Fandisk</td>
<td>13k</td>
<td>CVD</td>
<td>95</td>
<td>1</td>
</tr>
<tr>
<td>Fandisk</td>
<td>13k</td>
<td>planar</td>
<td>-</td>
<td>13</td>
</tr>
<tr>
<td>Bunny</td>
<td>70k</td>
<td>CVD</td>
<td>-</td>
<td>463</td>
</tr>
<tr>
<td>Bunny</td>
<td>70k</td>
<td>planar</td>
<td>-</td>
<td>435</td>
</tr>
<tr>
<td>Bunny</td>
<td>70k</td>
<td>CVD</td>
<td>50</td>
<td>167</td>
</tr>
<tr>
<td>Bunny</td>
<td>70k</td>
<td>CVD</td>
<td>90</td>
<td>38</td>
</tr>
<tr>
<td>Horse</td>
<td>97k</td>
<td>CVD</td>
<td>-</td>
<td>988</td>
</tr>
<tr>
<td>Horse</td>
<td>97k</td>
<td>planar</td>
<td>-</td>
<td>852</td>
</tr>
<tr>
<td>Horse</td>
<td>97k</td>
<td>CVD</td>
<td>50</td>
<td>398</td>
</tr>
<tr>
<td>Horse</td>
<td>97k</td>
<td>CVD</td>
<td>90</td>
<td>107</td>
</tr>
</tbody>
</table>

Table is always smaller than that of the HFC or VC algorithms. Thus the VMLC algorithm performs better in this example. This behavior has been observed with similar quality for other models.

In Fig. 8 we also depicted the energy behavior when applying a variation of the VMLC algorithm for different values of the parameter $p$. Observe the energy jump for $p = 95\%$. It appears at the point where 95% of the merging steps are done, so the total energy is higher than that of HFC algorithm. Because after this point the optimization is also applied, the energy starts to decrease approaching the energy of the VMLC algorithm, i.e. for $p = 0\%$. We observed, that for CVD and plane fitting the energy always rapidly approaches the optimal path after the optimization is activated. Thus, the curves for, e.g. $p = 50\%$, is nearly coincident with $p = 0\%$ after a few loops of merging and optimization.

Table 2 provides the timing results for different meshes. The second column gives the number of faces for the input model. The third one specifies the energy functional used, i.e. CVD or planar. The fourth one gives the value for $p$ in the case of a variation of the VMLC algorithm (see Sec. 6.6). The last column shows the time for performing the complete VMLC-based clustering.

8 Conclusion and Future Work

We have presented a novel and versatile algorithm for the computation of an optimized multilevel mesh clustering. We showed that the VMLC algorithm performs better than hierarchical clustering or standard variational clustering. We have provided a very efficient implementation and data structure for this algorithm. As part of this algorithm the generalization of the approximated CVD algorithm was presented. We showed that different variations of the algorithm are possible, thus allowing the user to choose between faster execution or higher quality of the final result.

We also have shown the generic nature of this approach by applying it to different tasks like planar cluster approximation or for constructing a CVD. The $L^2$ metric was simplified to a form which allows a very efficient energy computation using our approach.

We already pointed out (see remark in Sec.6.5) that there are no fundamental limitations regarding the choice of the energy functional to be used with our VMLC algorithm. However, one should be always aware of the computational cost involved with the energy functional. In future we plan to design alternative energy functionals which allow for efficient local energy computations like for CVD (see Eqs. (6)-(8)) or like for planar approximation (see Eqs. (10)-(12)) in order to apply the VMLC algorithm to different tasks.

References

Figure 4: The CVD clustering results using the HFC algorithm (top) and the VMLC algorithm (bottom) for the Bunny model. (a) Hierarchy level=500. (b) Hierarchy level=200.

Figure 5: The planar clustering results using the HFC algorithm (top) and the VMLC algorithm (bottom) for the Fandisk model at the level of 50 clusters.

Figure 6: The planar clustering results using the HFC algorithm (left) and the VMLC algorithm (right) for a sphere model at a level of 100 clusters.

Figure 7: The planar clustering results using the HFC algorithm (left) and the VMLC algorithm (right) for the horse model (top view) at a level of 500 clusters.

Figure 8: CVD Energy versus # of clusters for Fandisk model. HFC (hierarchical face clustering), VC (variational clustering) with 50 random initialization for the given number of clusters. (p:) applying a variation of the VMLC algorithm for p%.