# Contraction Moves for Geometric Model Fitting

Oliver J. Woodford, Minh-Tri Pham, Atsuto Maki,
Riccardo Gherardi, Frank Perbet, Björn Stenger

Toshiba Research Europe Ltd., Cambridge, UK

**Abstract.** This paper presents a new class of moves, called *$\alpha$-expansion-contraction*, which generalizes $\alpha$-expansion graph cuts for multi-label energy minimization problems. The new moves are particularly useful for optimizing the assignments in model fitting frameworks whose energies include *Label Cost* (LC), as well as *Markov Random Field* (MRF) terms. These problems benefit from the contraction moves' greater scope for removing instances from the model, reducing label costs. We demonstrate this effect on the problem of fitting sets of geometric primitives to point cloud data, including real-world point clouds containing millions of points, obtained by multi-view reconstruction.

## 1 Introduction

With recent advances in 3D data capture, the problem of fitting multiple instances of geometric primitives to 3D data, which was studied extensively in the early computer vision literature [1,2,3,4], has received renewed interest [5,6,7]. Such models are necessary not only when data is noisy or poorly sampled, requiring greater regularization, but also when scenes can be represented by a small number of primitives of known type, *e.g.* quadric surfaces [4].

The task of fitting multiple instances is generally challenging due to a large configuration space of potential models, leading to a difficult optimization problem consisting of a discrete assignment of points to primitives, as well as a continuous parameter optimization problem. With energy-based frameworks that include both label cost (LC) and MRF terms (like ours), the assignment problem can be solved using multi-label graph cut methods [7,8]. However, standard moves such as $\alpha$-expansion will be shown to have limitations when applied to this task. We introduce $\alpha$-expansion-contraction moves to overcome this deficiency.

We now review model fitting methods, and the relevant discrete energy optimization literature.

### 1.1 Model fitting

Both the Hough transform [9] and RANSAC [10] are model fitting algorithms that, given a set of features (here, points), find the most likely model instance (here, primitives) that generated those features. The former discretizes the instance parameter space, then iterates through features, incrementing the vote

count for every instance that feature might be generated by. While exhaustive to within the discretization accuracy, this approach scales poorly to high-dimensional parameter spaces. RANSAC solves this problem by instead iterating through instances, hypothesizing one instance at a time, computed from random, minimal sets of features, counting the votes for that instance, then storing the most-voted-for instance.

Neither method is suited to multi-instance fitting, as features, though in general generated by a single instance, vote for multiple instances, creating false peaks. This issue was first addressed in the Hough transform by Gerig [11], who proposed greedily assigning features to instances in a one-to-one mapping, the importance of which was recently rediscovered [12,13]. RANSAC has been extended using a similar greedy approach: running it multiple times, and removing points associated with successfully fitted instances [5,14]. One drawback, that points, once assigned, cannot change assignments, is addressed by the J-linkage algorithm [15], which uses a clustering step to make the assignments. Similarly, both families of methods have been extended to locally optimize instance parameters [16,17] while computing assignments.

Several more recent, energy-based methods [7,8,18] incorporate the notion of the one-to-one mapping of features to instances in an assignment labelling, which is updated iteratively, along with the parameters of each instance, in a coordinate descent manner. These can be seen as generalizations to both Hough and RANSAC approaches, initializing instances randomly like RANSAC, and maintaining a list of potential instances like the Hough transform, whilst also enforcing a feature-to-instance assignment similarly to [11,12,13,15], and performing local optimizations similarly to [16,17]. Importantly, the assignments are free to change throughout the optimization.

### 1.2 Discrete optimization

The assignment update of energy-based model fitting algorithms (like ours) is essentially a discrete optimization over labellings, given data likelihood and regularization terms. When the latter incorporates a pairwise MRF spatial smoothness cost on the labelling, this problem is generally NP-hard,[1] but an evaluation [19] indicates that optimizers such as TRW-S and graph cuts find good approximations. In our case, graph cuts is preferable as it has much lower memory requirements, which is important given the size of our point clouds and number of potential primitives.

Using graph cuts, multi-label energies can be iteratively minimized through a series of graph cut optimizations, called *moves*, each of which *fuse* a proposal labelling to the current labelling [20,21,22,23]. At each iteration the optimal output labelling, $\mathcal{L} = (l_i)_{i=1}^{N}$, can be found in polynomial time when all pairwise terms, $V_{ij}$, satisfy the submodularity constraint [24]:

$$V_{ij}(l_i^c, l_j^c) + V_{ij}(l_i^p, l_j^p) \leq V_{ij}(l_i^p, l_j^c) + V_{ij}(l_i^c, l_j^p), \tag{1}$$

[1] Some energies can be minimized optimally in polynomial time, but these fall outside the scope of our work.

where $i$ and $j$ represent indices of points, and $l_i^c$ and $l_i^p$ their current and proposed labels respectively. Existing strategies for choosing the proposal labelling, which determines a *move space* (the set of possible output labellings), include $\alpha$-expansion [20], $\alpha\beta$-swap [20], $\alpha$-expansion $\beta$-shrink [21], range moves [25] (multi-label) and fusion moves [22,23]. Two such move spaces, given by the two labellings fused at each graph cut optimization, are

$$l_i = \begin{cases} \alpha & \text{for } \; l_i^c = \alpha \\ \alpha \text{ or } l_i^c & \text{otherwise} \end{cases}, \qquad l_i = \begin{cases} \alpha \text{ or } \beta & \text{for } \; l_i^c = \alpha \\ \alpha \text{ or } l_i^c & \text{otherwise} \end{cases}, \tag{2}$$

($\alpha$-expansion [20]) ($\alpha$-expansion $\beta$-shrink [21])

where $\alpha$ and $\beta$ change to a new label in every iteration. The submodularity constraint requires that every multi-label pairwise term, $V_{ij}$, be *metric* [20] for both these moves. Other moves have other requirements; fusion moves allow general pairwise terms and proposal labellings, but guarantee only not to increase the energy at each iteration [23].

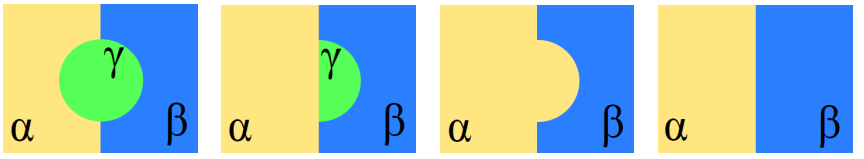
**Label cost terms** Model complexity terms formalize the notion that simple models are *a priori* more likely than complex ones. One example, the minimum description length (MDL) criterion [26], has been used in image [27,28] and motion [29] segmentation. Recent model fitting algorithms use a simpler term, called *labels costs* [8], consisting only of a cost for each instance in the model [7,8,12], which is paid when at least one feature is assigned to the instance.

It is known that such global label costs can be constructed from pairwise terms; see [8] for a discussion. However, the most intuitive way this can be seen is by creating an explicit, binary latent variable (*e.g.* visibility variables in [22]), $b_\gamma$, to represent the existence of a particular instance, say with label $\gamma$, with unary costs of $U_\gamma(b_\gamma = 0) = 0$ and $U_\gamma(b_\gamma = 1) = c$, $c$ being the instance's label cost. Connecting each point, $i$, to the latent variable with a pairwise constraint term of the form

$$V_{i\gamma}(l_i, b_\gamma) = \begin{cases} \infty & \text{for } \; l_i = \gamma, \; b_\gamma = 0 \\ 0 & \text{otherwise} \end{cases}, \tag{3}$$

ensures that if at least one point is assigned to the instance, then the label cost is paid. If all labels of a particular value are associated with the same binary label in each graph cut optimization, then the pairwise terms are identical and either all submodular, or non-submodular, in which case the meaning of $b_\gamma$ can be inverted to ensure submodularity of all the pairwise terms connected to it. This is the case for $\alpha$-expansion, since the label $\alpha$ is associated with 0 and all other labels are associated with 1.

Given a model consisting of several instances, one potential way to reduce the energy is to remove redundant instances/labels, thus removing their label costs. However, all current submodular moves can only remove a label in a *single* graph cuts optimization by replacing that label entirely with a single other label. This means that if the optimal replacement of a label involves several other labels,




(a) Current (b) Intermediate (c) Intermediate (d) Optimal

Fig. 1: **Removing labels**. (a) A given suboptimal labelling. (b), (c) Two example intermediate labellings on the way to the optimal labelling (d), which cannot be reached from (a) in a single move using any of $\alpha$-expansion, $\alpha\beta$-swap or $\alpha$-expansion $\beta$-shrink, as none allow one label to be replaced with two others.

multiple moves must be used to achieve this (see figure 1). If the energies of all intermediate labellings are higher than the current energy, then the optimal outcome is not achievable—the labelling is in a local minimum. The $\alpha$-expansion-contraction moves proposed here can replace labels with multi-label proposals in a single optimization, allowing these local minima to be avoided.

Alternatively, *merge* steps [7] can alleviate this problem, by proposing to merge pairs of labels while simultaneously optimizing the instance parameters for the merged assignments, but this approach is task specific.

### 1.3 Contributions

In §3 of this paper, we propose the new class of $\alpha$-*expansion-contraction* moves, which generalizes $\alpha$-expansion with the ability to remove, or *contract*, the $\alpha$ label with a proposal labelling of *any* configuration. Using $\alpha$-expansion-contraction as our basic scheme, we introduce two methods for selecting a *proposal* labelling for the labels currently assigned to $\alpha$. In §4 we demonstrate the benefits of $\alpha$-expansion-contraction in the context of geometric primitive fitting, as well as evaluating on a standard stereo dataset, showing that both proposals improve on energies found using $\alpha$-expansion when label costs are predominant. We further show the suitability of this approach to large-scale problems. First we introduce our energy-based geometric primitive model fitting framework.

## 2 Model fitting framework

Given a set of input points, $\{\mathbf{x}_i\}_{i=1}^N$, consisting of 3D position and normal direction, our goal is two-fold: to compute the parameters, $\boldsymbol{\Theta} = \{\theta_k\}_{k=1}^M$ ($\theta_k$ being the parameter vector of the $k^{\text{th}}$ primitive) of a set of primitives of unknown (*i.e.* variable) size, $M$, and to find the labelling $\mathcal{L} = (l_i)_{i=1}^N$ that assigns points to primitives, s.t. $l_i \in \{0, .., M\}$, 0 denoting the unassigned state. We use an energy-based framework, defining the energy and optimization strategy below.

### 2.1 Energy function

Our energy, consisting of a data term which encourages fidelity of the model to the data, an MRF term which encourages spatially coherent labellings, and an LC term which encourages simpler models, is defined as

$$E(\mathcal{L}, \boldsymbol{\Theta}) = \underbrace{\sum_{i=1}^{N} D_i(\mathbf{x}_i, \theta_{l_i})}_{\text{data term}} + \lambda_{\text{MRF}} \underbrace{\sum_{(i,j)\in\mathcal{N}} V(l_i, l_j)}_{\text{MRF term}} + \lambda_{\text{LC}} \underbrace{\sum_{k=1}^{M} w_{T(k)} \delta_{\mathcal{L}}(k)}_{\text{LC term}}, \quad (4)$$

where $\lambda_{\text{MRF}}$ and $\lambda_{\text{LC}}$ are input parameters which weight the importance of the regularization terms. We describe the three terms below.

We assume that each $\mathbf{x}_i$ is a measurement of some point $\mathbf{y}_i$ on $\theta_{l_i}$ (*i.e.* the position of $\mathbf{y}_i$ is on the surface of $\theta_{l_i}$ and its normal direction is the normal vector of $\theta_{l_i}$ at that position), with known Gaussian noise. While it is theoretically correct to marginalize over the possible values of $\mathbf{y}_i$, in practice we approximate this by using only the closest point to $\mathbf{x}_i$ on $\theta_{l_i}$, denoted by $\theta_{l_i}(\mathbf{x}_i)$. Our per-point data cost is therefore

$$D_i(\mathbf{x}_i, \theta_{l_i}) = (\theta_{l_i}(\mathbf{x}_i) - \mathbf{x}_i)^{\mathsf{T}} \bar{\boldsymbol{\Sigma}}_i^{-1} (\theta_{l_i}(\mathbf{x}_i) - \mathbf{x}_i), \quad (5)$$

where $\{\bar{\boldsymbol{\Sigma}}_i\}_{i=1}^{N}$ represents the set of given per-point covariance matrices. The data cost of the unassigned label, $l_i = 0$, is a fixed cost equivalent to two standard deviations of the noise term.

The MRF term regularizes instance boundary length, encouraging spatial coherence, by penalizing pairs of neighbouring points which do not share the same assignment, *i.e.* a Potts model, thus:

$$V(l_i, l_j) = \begin{cases} 0 & \text{for } l_i = l_j \\ 1 & \text{otherwise} \end{cases}. \quad (6)$$

The neighbourhood set, $\mathcal{N}$, is a list of all pairs of points which are close to each other, closeness here defined using Euclidean distance below some threshold.

The LC term penalizes model complexity by paying a cost for each primitive, indexed by $k$, when at least one point is assigned to it, thus:

$$\delta_{\mathcal{L}}(k) = \begin{cases} 1 & \exists i : l_i = k \\ 0 & \text{otherwise} \end{cases}. \quad (7)$$

This cost is weighted by a factor, $w_{T(k)}$, which is defined according to the type of primitive, denoted $T(k)$, and is equal to the number of free parameters of each primitive type.

This energy function is very similar to that of [7,8], but additionally incorporates normal errors into the data term.

**Algorithm 1** Optimization pseudo-code for fitting primitives to a point cloud

**Require:** point cloud $\{\mathbf{x}_i\}, i = 1, .., N$
1: $t = 0$ : Initialize assignments $\mathcal{L}$ and parameters $\mathbf{\Theta}$, compute energy $E_0$
2: **while** $E_t < E_{t-1}$ **do**
3: Add new primitives (optional)
4: Update assignments $\mathcal{L}$ with graph cuts using $\alpha$-expansion-contraction
5: Remove primitives with no points assigned to them
6: Optimize parameters $\mathbf{\Theta}$ using Levenberg-Marquardt
7: Recompute energy $E_t$
8: t = t+1
9: **end while**

### 2.2 Optimization

We adopt a very similar optimization strategy to those of [7,8], outlined in Algorithm 1, referred to below. Given current values of $\mathcal{L}$ and $\mathbf{\Theta}$, we use a coordinate descent approach, alternating once between optimizing the discrete labelling, $\mathcal{L}$, and the continuous parameters, $\mathbf{\Theta}$, per algorithm iteration.

For the latter optimization (step 6), *i.e.* minimizing $E(\mathcal{L}, \mathbf{\Theta})$ w.r.t. $\mathbf{\Theta}$, the fixed assignments mean that the parameters of each primitive can be optimized independently of each other and also of the regularization terms, using Levenberg-Marquardt, as follows:

$$\theta_k = \underset{\theta}{\operatorname{argmin}} \sum_{i|l_i=k} D_i(\mathbf{x}_i, \theta). \tag{8}$$

The discrete optimization (step 4), *i.e.* minimizing $E(\mathcal{L}, \mathbf{\Theta})$ w.r.t. $\mathcal{L}$, is achieved through a series of $\alpha$-expansion-contraction moves, described in the next section, iterating through each label once (rather than until convergence) per algorithm iteration, in the fixed order $\{1, .., M, 0\}$.

There are two other model update steps per iteration, once at the start, when new primitives are optionally added to the model (step 3), and once after the assignment update, when primitives with no points assigned to them are removed from the model (step 5). Adding primitives is achieved by selecting a seed point from the input set at random, but favouring those with currently high data costs, and locally fitting one primitive of each available type to a neighbourhood of points (we use 20 here) around that point.

At the end of each iteration the energy, which is guaranteed to not increase, as only steps 4 and 6 change the energy and neither can increase it, is recomputed (step 7), and the algorithm is looped until convergence (step 2). In the case where primitives are being added, the algorithm only stops when five iterations in a row fail to decrease the energy.

Finally, we have two model initialization schemes (step 1). Both involve initializing all points as unassigned, s.t. $l_i = 0$, $i = 1, .., N$. However, the primitives, $\mathbf{\Theta}$, can either be initialized as an empty set, or filled with a number of primitives initialized in the same manner as those added in step 3, but with seed points selected uniformly at random.

## 3 Contraction moves

We introduce a new class of moves, called $\alpha$-expansion-contraction, which generalizes both $\alpha$-expansion and $\alpha$-expansion $\beta$-shrink moves. For every point whose current label $l_i^c = \alpha$, it proposes a new label $l_i^p$. Importantly, the set of labels $\mathcal{L}^p = (l_i^p)_{\forall i | l_i^c = \alpha}$ is allowed to have *any* configuration. The $\alpha$-expansion-contraction move space is therefore defined as

$$l_i = \begin{cases} \alpha \text{ or } l_i^p & \text{for } \; l_i^c = \alpha \quad (\alpha\text{-contraction}) \\ \alpha \text{ or } l_i^c & \text{otherwise} \qquad (\alpha\text{-expansion}) \end{cases} . \tag{9}$$

The move has two key properties, which can be seen by inspection of equations (2) and (9). Firstly, like $\alpha$-expansion [20], given *metric* pairwise terms, it can be solved optimally in polynomial time, since our move is equivalent to an $\alpha$-expansion on the labelling $(l_i^p \text{ for } l_i^c = \alpha,\ l_i^c \text{ otherwise})_{i=1}^N$. Secondly, for any choice of $l_i^p$, our move is guaranteed to have an equal or lower energy than that of the equivalent (*i.e.* same $\alpha$) $\alpha$-expansion move, since the former's move space is a superset of the latter's, and we find the optimal solution. Finally, we can see that this class of move subsumes the $\alpha$-expansion $\beta$-shrink class, where $l_i^p = \beta$ (*i.e.* the proposal labelling, $\mathcal{L}^p$, must be homogeneous).

The motivation of $\alpha$-expansion-contraction is to better enable the removal of label $\alpha$ in one move, thereby reducing label cost. The removal of a label in a single $\alpha$-expansion or $\alpha$-expansion $\beta$-shrink move can only happen by replacing the label entirely with the $\alpha$ (or $\beta$) label. In contrast, $\alpha$-expansion-contraction has the potential to replace a label with any number of other labels in a single move, benefiting situations where the optimal removal involves multiple labels. However, it should be noted that $\alpha$-expansion-contraction does not necessarily propose the optimal removal labelling—finding this labelling is NP-hard, but even computing an approximation would require a multi-label optimization using an inner loop of $\alpha$-expansion moves, which is not practical. Instead we propose two heuristic methods for selecting these proposal labels, which we call variant I and variant II, described below.

### 3.1 $\alpha$-expansion-contraction: variant I

Here, each proposal label, $l_i^p$, is computed independently, by selecting the label (excluding the current label, $l_i^c = \alpha$), with the lowest data cost given by equation (5). This variant allows the $\alpha$ label to be replaced with multiple labels. This proposal is optimal when there are no MRF costs, *i.e.* $\lambda_{\text{MRF}} = 0$, therefore intuitively we can expect this variant to perform better with lower $\lambda_{\text{MRF}}$.

*Implementation details*: To avoid either storing or recomputing the data costs of every label for each point, $i$, we only store the current label's data cost, $d_i^{l_i}$, and the lowest (or second lowest if the current label has the lowest) data cost, $d_i^*$, and recompute the data cost for the label $\alpha$ during each move. The label of the currently stored $d_i^*$ is therefore used as $l_i^p$. However, due to changes in primitive parameters, this label may not always strictly fulfil the description of

$l_i^p$ given above. Each $d_i^*$ is compared against the $\alpha$ label data cost after each move, and updated if necessary.

Furthermore, in each $\alpha$-expansion-contraction optimization, if $l_i^c \neq \alpha$ and $D_i(\mathbf{x}_i, \theta_\alpha) > D_i(\mathbf{x}_i, \theta_0) + \lambda_{\text{MRF}}|\mathcal{N}_i|V(0, \alpha)$, where $|\mathcal{N}_i|$ is the number of neighbours of the $i^{\text{th}}$ node, then that node is guaranteed to produce a lower energy with the label 0 rather than the label $\alpha$, regardless of the labels of other nodes, therefore it is pruned from the optimization to save computation time.

### 3.2 $\alpha$-expansion-contraction: variant II

Here, every proposal label $l_i^p$ is set to the same value, $\beta$. This variant coincides with an $\alpha$-expansion $\beta$-shrink move [21], for which four choices of $\beta$ were suggested: $\alpha - 1$, $\alpha + 1$, random, or iterating over all labels. The first two are meaningless in applications where labels have no intrinsic ordering, as with primitive fitting, while the last has $M^2$ iterations per loop, rather than the $M$ of $\alpha$-expansion, making it computationally undesirable. Finally, with a large $M$ but only a few suitable primitives per point, random selection in our application is like a shot in the dark. Instead we propose a new heuristic for selecting a value for $\beta$ that is sensible given our application: we histogram the labels $l_i^p$ proposed by variant I, and choose $\beta$ as the label of the bin with most entries.

This variant proposes a homogeneous labelling. Since homogeneous labellings are more likely when $\lambda_{\text{MRF}}$ is high, we can intuitively expect this variant to perform better with higher $\lambda_{\text{MRF}}$.

## 4 Evaluation

We evaluate our proposed framework on synthetic and real-world point cloud datasets, fitting models consisting of the following three types of primitives: planes, spheres and cylinders. The synthetic dataset, shown in figure 2, is generated by randomly sampling 50,000 points uniformly over the area of each of ten meshes from the Princeton Shape Benchmark database [30], computing mesh normals at each point, then adding Gaussian noise to both position and normals. In our quantitative evaluation, each synthetic point cloud is modelled three times, using both variants of $\alpha$-expansion-contraction in step 4 of Algorithm 1, as well as $\alpha$-expansion for comparison.

### 4.1 Comparison of initialization methods

In this experiment we evaluate our two approaches to initializing the algorithm: starting with an empty set of primitives, and starting with a number of random primitives. In contrast to the empty set initialization, we do not add primitives during step 3 when starting with a number of primitives.

The plots in figure 4 compare the average energy values after convergence for four different settings of $\lambda_{\text{MRF}}$ and $\lambda_{\text{LC}}$. Dashed lines represent the final energies

of the empty-set initialization approaches, the $\alpha$-expansion version of which is used to normalize all other energy values.

A first observation is that the final energies obtained using the two proposed $\alpha$-expansion-contraction variants are, on average, less than or equal to the $\alpha$-expansion result in all four cases. The improvement is particularly significant in the case of high $\lambda_{\mathrm{LC}}$ and low $\lambda_{\mathrm{MRF}}$ (fig. 4(d)); for these settings the ability to efficiently remove labels is important in reducing energy. Secondly, for all settings of $\lambda_{\mathrm{MRF}}$ and $\lambda_{\mathrm{LC}}$, starting with a sufficiently large number of primitives leads to a lower energy, on average, than when starting with zero primitives. In our experiments the energy decrease saturates at around $M = 3200$; we use this test case for all further experiments on this dataset.

### 4.2 Comparison over different regularization weights

In the second experiment we compare the algorithms in terms of final energy after convergence over a range of values of the regularization weights, $\lambda_{\mathrm{MRF}}$ and $\lambda_{\mathrm{LC}}$. Each row of figure 6 compares two algorithms by showing the percentage reduction in final energy. Positive numbers correspond to a lower energy solution of the first method.

Shown in the first and second rows are the improvements by $\alpha$-expansion-contraction over standard $\alpha$-expansion. The average energy of both contraction variants is less than or equal to the value obtained by $\alpha$-expansion (first column), and the improvement is greatest for lower values of $\lambda_{\mathrm{MRF}}$, where the homogeneous labelling assumptions inherent in the $\alpha$-expansion moves are least true. The second and third columns show the minimum and maximum values of energy decrease over the ten point clouds. Both contraction variants generally outperform the standard $\alpha$-expansion approach, with only a few negative values, corresponding to low values of $\lambda_{\mathrm{LC}}$. By contrast, the maximum improvement over standard $\alpha$-expansion can be significant, in some cases exceeding a 50% energy reduction.

The difference between variant I and variant II of $\alpha$-expansion-contraction, shown in the third row, is less pronounced but it is still evident that for most scenarios, particularly for low values of $\lambda_{\mathrm{MRF}}$, variant I leads to lower energies on average. Important also is that, for larger values of $\lambda_{\mathrm{MRF}}$, variant I does not perform worse on average than variant II, which one might expect. This may be due to the fact that variant I incorporates both homogeneous and inhomogeneous regions. The result indicates that it is preferable to use the more general variant I, which does not coincide with any previously introduced moves, in all cases.

The number of primitives at convergence greatly depends on the parameters used; final primitive counts range from 50 up to 2500 for small $\lambda_{\mathrm{MRF}}$ and $\lambda_{\mathrm{LC}}$. However, the final number of primitives produced by the $\alpha$-expansion-contraction variants is always less than or equal to that of $\alpha$-expansion, with decreases of up to 40% with variant II and up to 60% with variant I. Figure 5 shows that these simpler models (*i.e.* reduced label costs) do not increase the point-to-primitive distances (*i.e.* data cost) much; in fact with variant II they decrease compared to $\alpha$-expansion, as does the MRF cost.

Figure 7(a) qualitatively shows improved fitting on the *Microscope* point cloud used in the experiments. It can be seen that $\alpha$-expansion-contraction leads to a lower number of primitives, with an improved segmentation of the dominant surfaces compared to standard $\alpha$-expansion.

### 4.3 Convergence behaviour

In this experiment the convergence and runtime performance of the three methods is compared. Figure 3 shows two cases with fixed $\lambda_{\mathrm{MRF}}$, but low and high $\lambda_{\mathrm{LC}}$ values. Each graph plots the energy versus CPU time for all three algorithms on each of the ten point clouds. Both variants of $\alpha$-expansion-contraction converge more slowly, but to lower energies than standard $\alpha$-expansion, as well as achieving lower energies in the time that $\alpha$-expansion takes to converge. This is particularly visible in the case of large $\lambda_{\mathrm{LC}}$. Another observation is that in most cases variant I of $\alpha$-expansion-contraction converges faster and to a better solution than variant II.

### 4.4 Experiments on multi-view reconstruction data

We apply the algorithm to three point clouds, *Sculpture*, *Forum* and *Temple*, obtained with a multi-view reconstruction pipeline [31]. For these datasets, all methods are initialized with the output of an efficient, RANSAC-based primitive fitting method [5], but new primitives are still added each iteration.

The model fitting results (fig. 7(c–d)) show qualitative improvements of the proposed method compared to $\alpha$-expansion. Using $\alpha$-expansion produces many more primitives than $\alpha$-expansion-contraction variant I, a fact that is more obvious in the case of the *Sculpture* and *Temple* datasets (b,d), where curved surfaces are more frequently modelled as fragments of planes by $\alpha$-expansion. For the *Temple* point cloud, which contains more that 2.6M points, $\alpha$-expansion-contraction variant I produced an energy 20% lower than $\alpha$-expansion, using 31% fewer primitives, but converging in 510 iterations compared to the latter's 140. Each algorithm iteration took on average over 700 seconds for both approaches, with 99% of that time being spent on the assignment updates (step 4 of Algorithm 1); however, we do not use any of the available approaches for speeding up $\alpha$-expansion (and consequently $\alpha$-expansion-contraction).

### 4.5 Experiments on stereo

$\alpha$-expansion-contraction moves can replace $\alpha$-expansion moves in any application with a multi-label energy consisting of metric pairwise terms. In computer vision the majority of such applications do not include label costs, therefore we also compare our new moves with $\alpha$-expansion on a standard energy minimization dataset [19] which does not include such costs. However, we only use the tests which have metric[2] pairwise terms—stereo. The results, shown in table 1, suggest

[2] We have not investigated the possibility of truncating non-submodular pairwise terms whilst maintaining convergence properties, *e.g.* as in [21, §4.2].

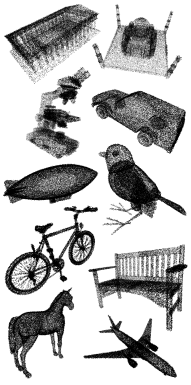
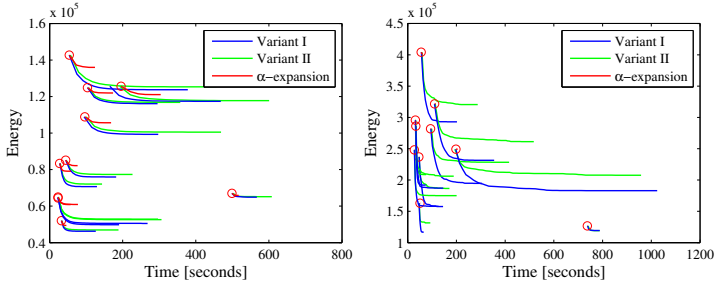
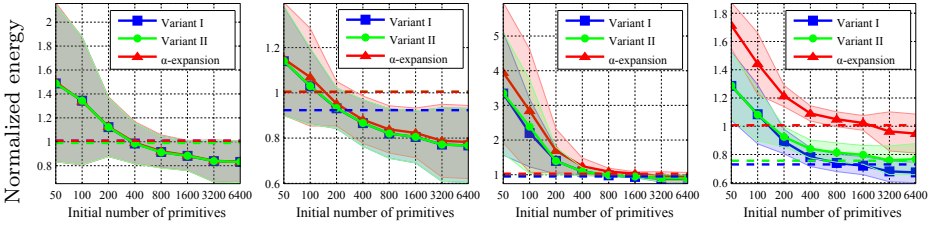


Fig. 2: **Synthetic dataset**.




(a) $\lambda_{\text{MRF}} = 0.125$, $\lambda_{\text{LC}} = 4$ (b) $\lambda_{\text{MRF}} = 0.125$, $\lambda_{\text{LC}} = 256$

Fig. 3: **Energy *vs.* CPU time**. Graphs showing the convergence behaviour of the different algorithms on ten point clouds, for two different values of $\lambda_{\text{LC}}$. The graphs start at the energy values after the first iteration.




(a) $\lambda_{\text{MRF}} = 8$, $\lambda_{\text{LC}} = 4$ (b) $\lambda_{\text{MRF}} = 8$, $\lambda_{\text{LC}} = 256$ (c) $\lambda_{\text{MRF}} = 0.125$, $\lambda_{\text{LC}} = 4$ (d) $\lambda_{\text{MRF}} = 0.125$, $\lambda_{\text{LC}} = 256$

Fig. 4: **Initialization strategies**. Plots showing results for different values of $\lambda_{\text{MRF}}$ and $\lambda_{\text{LC}}$. Each plot shows energy values after convergence for different initialization methods: starting with zero primitives and progressively adding them (dashed lines), and starting with different numbers of primitives (solid lines). The average and standard deviation over the 10 point clouds are shown. The case of standard $\alpha$-expansion with empty-set initialization is used as a baseline value (dashed red line) by which all other values are normalized.

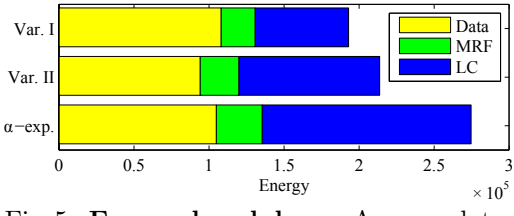



Fig. 5: **Energy breakdown.** Average data, MRF and LC components of the final energies for each method, with $\lambda_{\text{MRF}} = 0.125$ and $\lambda_{\text{LC}} = 256$.

| | Tsukuba | Venus | Teddy |
|---|---|---|---|
| Variant I | 370827 (4.6s) | 3011962 (12.9s) | 1343467 (47.8s) |
| Variant II | 370812 (4.8s) | 3012195 (11.6s) | 1343530 (43.5s) |
| $\alpha$-expansion | 370812 (4.8s) | 3011945 (12.4s) | 1343533 (41.7s) |

Table 1: **Stereo.** Energy and time results for evaluation on a standard stereo dataset [19].

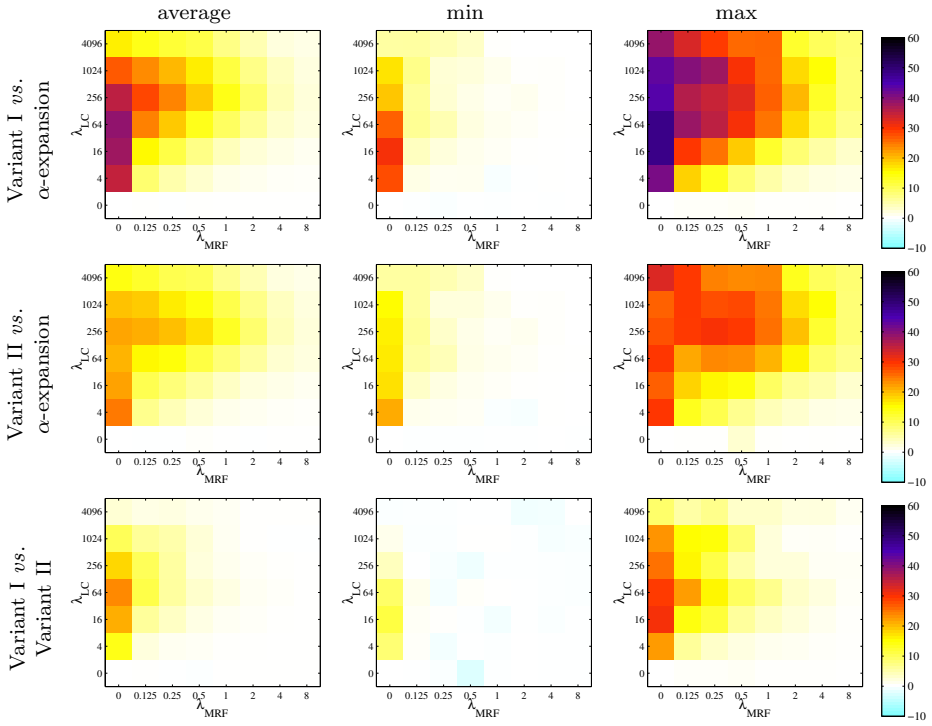



Fig. 6: **Comparison of final energies**. Each row of grids shows the percentage reduction in energy of one method *vs*. another, over a range values for $\lambda_{\text{MRF}}$ and $\lambda_{\text{LC}}$. Each column shows, from left to right, the average, minimum and maximum reductions over the ten datasets.

that little is to be gained by using either variant of $\alpha$-expansion-contraction (and therefore also $\alpha$-expansion $\beta$-shrink) on such applications. This is corroborated by the $\lambda_{\text{LC}} = 0$ rows in figure 6.

## 5 Conclusion

This paper has introduced the class of $\alpha$-expansion-contraction moves, a generalization of $\alpha$-expansion moves, for multi-label graph cuts optimization, and two heuristic methods for generating proposal labellings. We demonstrated this approach in the context of fitting geometric primitive models to 3D point clouds. In experiments on synthetic data, the new moves were shown to result in significantly lower energy solutions, particularly for large weights of the LC term. Results on problems without label costs were less conclusive, suggesting that these moves are of particular benefit when applied to model fitting problems with label costs, where the proposed moves facilitate the removal of instances, thereby reducing model complexity. The scalability of the algorithm was demonstrated by applying it to real-world point cloud datasets with millions of points.

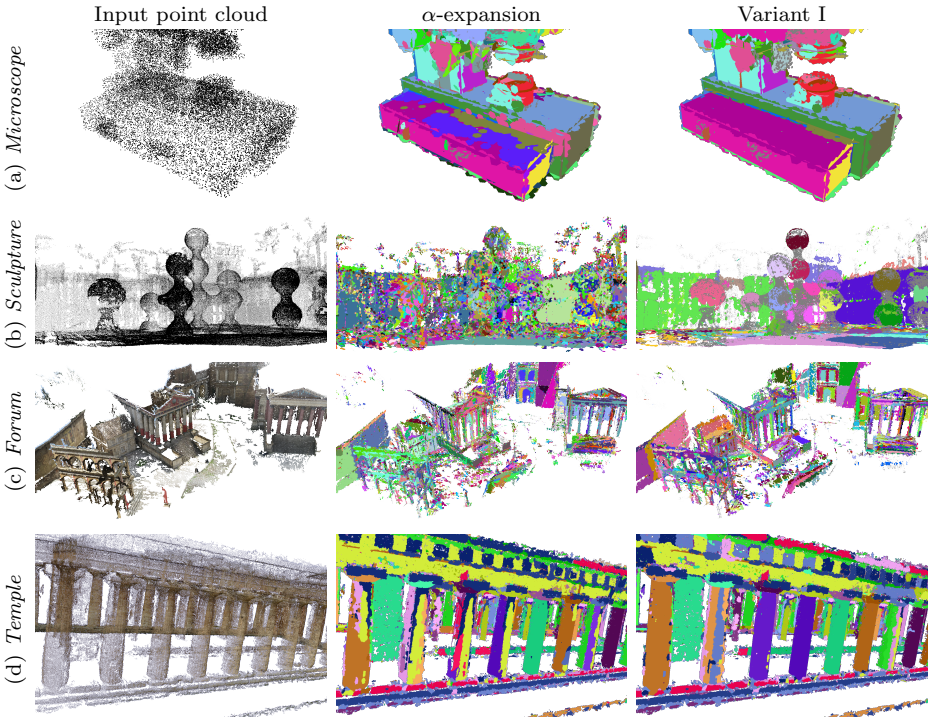



Fig. 7: **Primitive fitting results.** Qualitative results on (a) the synthetic *Microscope* dataset, (b) the *Sculpture* dataset, (c) the *Forum* dataset, and (d) the *Temple* dataset, showing colour coded primitives, visualized using oriented patches at each point, fitted using $\alpha$-expansion (*middle*) and $\alpha$-expansion-contraction variant I (*right*).

## References

1. Medioni, G., Parvin, B.: Segmentation of range images into planar surfaces by split and merge. In: CVPR. (1986) 415–417
2. Leonardis, A., Gupta, A., Bajcsy, R.: Segmentation as the search for the best description of the image in terms of primitives. In: ICCV. (1990) 121–125
3. Fitzgibbon, A.W., Eggerd, D.W., Fisher, R.B.: High-level CAD model acquisition from range images. Computer-Aided Design **29**(4) (January 1997) 321–330
4. Marshall, D., Lukacs, G., Martin, R.: Robust segmentation of primitives from range data in the presence of geometric degeneracy. TPAMI **23**(3) (March 2001) 304–314
5. Schnabel, R., Wahl, R., Klein, R.: Efficient RANSAC for point-cloud shape detection. Computer Graphics Forum **26**(2) (June 2007) 214–226
6. Lafarge F., Keriven R., B.M., Vu, H.: Hybrid multi-view reconstruction by jump-diffusion. In: CVPR. (2010)
7. Isack, H., Boykov, Y.: Energy-based geometric multi-model fitting. IJCV **97** (July 2011)

8. Delong, A., Osokin, A., Isack, H., Boykov, Y.: Fast approximate energy minimization with label costs. IJCV **96**(1) (January 2012)
9. Ballard, D.H.: Generalizing the Hough transform to detect arbitrary shapes. Pattern Recognition **13**(2) (1981) 111–122
10. Fischler, M.A., Bolles, R.C.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Comm. of the ACM **24**(6) (1981) 381–395
11. Gerig, G.: Linking image-space and accumulator-space: A new approach for object-recognition. In: ICCV. (1987) 112–117
12. Barinova, O., Lempitsky, V., Kohli, P.: On detection of multiple object instances using Hough transforms. In: CVPR. (2010)
13. Woodford, O.J., Pham, M.T., Maki, A., Perbet, F., Stenger, B.: Demisting the Hough transform for 3D shape recognition and registration. In: BMVC. (2011)
14. Zuliani, M., Kenney, C.S., Manjunath, B.S.: The multiRANSAC algorithm and its application to detect planar homographies. In: Proc IEEE Int. Conf. on Image Processing (ICIP). (September 2005)
15. Toldo, R., Fusiello, A.: Robust multiple structures estimation with j-linkage. In: ECCV. (2008)
16. Cheng, Y.: Mean shift, mode seeking, and clustering. TPAMI **17**(8) (1995) 790–799
17. Chum, O., Matas, J., Kittler, J.: Locally optimized RANSAC. In: Proceedings of DAGM. (2003)
18. Birchfield, S., Tomasi, C.: Multiway cut for stereo and motion with slanted surfaces. In: ICCV. (September 1999)
19. Szeliski, R., Zabih, R., Scharstein, D., Veksler, O., Kolmogorov, V., Agarwala, A., Tappen, M.F., Rother, C.: A comparative study of energy minimization methods for Markov random fields. In: ECCV. (2006) 16–29
20. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. TPAMI **23**(11) (2001) 1222–1239
21. Schmidt, M., Alahari, K.: Generalized fast approximate energy minimization via graph cuts: $\alpha$-expansion $\beta$-shrink moves. In: UAI. (2011) 653–660
22. Woodford, O.J., Torr, P.H.S., Reid, I.D., Fitzgibbon, A.W.: Global stereo reconstruction under second order smoothness priors. TPAMI **31**(12) (2009) 2115–2128
23. Lempitsky, V.S., Rother, C., Roth, S., Blake, A.: Fusion moves for Markov random field optimization. TPAMI **32**(8) (2010) 1392–1405
24. Ivănescu, P.L.: Some network flow problems solved with pseudo-boolean programming. Operations Research **13**(3) (1965) 388–399
25. Veksler, O.: Graph cut based optimization for MRFs with truncated convex priors. In: CVPR. (2007)
26. Rissanen, J.: Modeling by shortest data description. Automatica **14** (1978) 465–471
27. Leclerc, Y.G.: Constructing simple stable descriptions for image partitioning. IJCV **3**(1) (1989) 73–102
28. Zhu, S.C., Yuille, A.L.: Region competition: Unifying snakes, region growing, and Bayes/MDL for multiband image segmentation. TPAMI **18**(9) (1996) 884–900
29. Schindler, K., Suter, D.: Two-view multibody structure-and-motion with outliers through model selection. TPAMI **28**(6) (2006) 983–995
30. Shilane, P., Min, P., Kazhdan, M., Funkhouser, T.: The Princeton shape benchmark. In: Shape Modeling International. (June 2004)
31. Snavely, N., Seitz, S.M., Szeliski, R.: Photo tourism: Exploring image collections in 3D. ACM Trans. Graphics (Proc. SIGGRAPH) (2006)