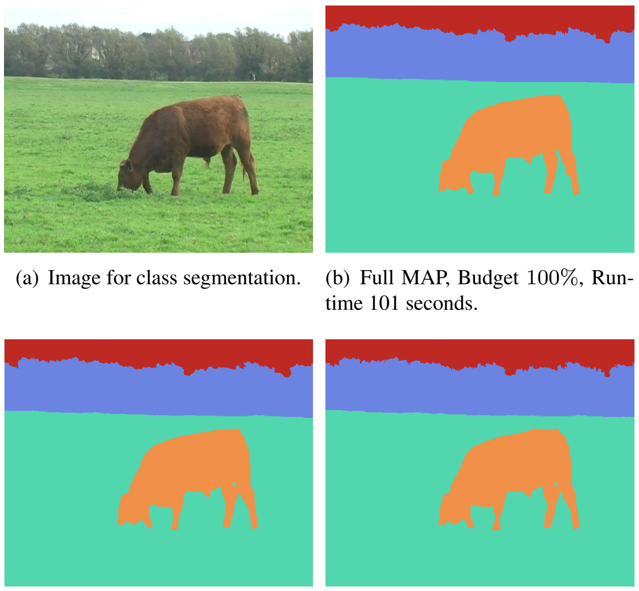
Variable Grouping for Energy Minimization

Taesup Kim[†], Sebastian Nowozin[‡], Pushmeet Kohli[‡], Chang D. Yoo[†]
Dept. of Electrical Engineering, KAIST, Daejeon Korea[†]
Microsoft Research Cambridge, Cambridge UK[‡]
kim.ts@kaist.ac.kr, sebastian.nowozin@microsoft.com, pkohli@microsoft.com, cdyoo@ee.kaist.ac.kr

Abstract

This paper addresses the problem of efficiently solving large-scale energy minimization problems encountered in computer vision. We propose an energy-aware method for merging random variables to reduce the size of the energy to be minimized. The method examines the energy function to find groups of variables which are likely to take the same label in the minimum energy state and thus can be represented by a single random variable. We propose and evaluate a number of extremely efficient variable grouping strategies. Experimental results show that our methods result in a dramatic reduction in the computational cost and memory requirements (in some cases by a factor of one hundred) with almost no drop in the accuracy of the final result. Comparative evaluation with efficient super-pixel generation methods, which are commonly used in variable grouping, reveals that our methods are far superior both in terms of accuracy and running time.



(a) Image for class segmentation.

(b) Full MAP, Budget 100%, Runtime 101 seconds.

(c) Grouped MAP, Budget 25%, Runtime 26.8 seconds, 99.62% label agreement with full MAP.

(d) Grouped MAP, Budget 1.56%, Runtime 6.63 seconds, 99.40% label agreement with full MAP.

Figure 1. Variable grouping reduces the energy minimization problem size with only a small loss in accuracy. The *budget* is the relative number of variables used in our approximation.

1. Introduction

Many computer vision problems such as image segmentation, stereo matching, and image restoration are formulated using probabilistic models such as discrete Markov and conditional random fields (MRF and CRF respectively). Computing the maximum a posteriori (MAP) solution under these models can be seen as minimizing an energy function defined over some discrete variables, which in general is an NP-hard problem [11, 3]. A number of powerful methods exist in the literature which are able to compute exact solutions for restricted classes of problems, such as those composed of submodular terms or having low tree-width, and approximate solutions for general problems. Graph cuts [11, 2, 3], loopy belief propagation (LBP) [26, 16], tree re-weighted message passing (TRW) [10, 25], max-sum diffusion [27] and FastPD [13] are some examples of such methods.

The energy functions encountered while solving labeling problems such as image segmentation and 3D reconstruction generally contain one discrete variable per image pixel (or 3D volume voxel). The minimization of such energy functions which may be defined over millions (and sometimes billions) of variables is an extremely computationally expensive operation. Labeling problems are becoming even larger as we move towards higher resolution images and videos that are generated using latest image acquisition devices. This observation has inspired research on the problem of developing more efficient energy minimization methods such as [1, 12, 9, 13]. However, the computational cost and memory requirements of minimization methods are still highly super-linear in the number of variables and terms present in the energy function.

In addition to developing more efficient methods for MAP inference, researchers have also tried reducing the size of the labeling problem itself. This simple and widely used technique works by merging the variables in the energy into a small number of groups and representing each group by a single variable. The technique has been successfully employed for solving image labeling problems such as object segmentation, stereo and single view reconstruction [6, 7, 18]. The number of variables is reduced by partitioning the image into small number of segments (also called *super-pixels*) [20]. This results in a smaller energy function containing one variable per segment, which is constructed from the original energy by assuming that all pixels that belong to the same segment (super-pixel) take the same label in the MAP solution (*label-consistency*). This function is minimized to produce a solution for the original image labeling problem. If the image partitioning is indeed *label-consistent*, then the energy minimizing solutions under both the original and scale-reduced (based on fewer variables) energies are the same, and thus minimization of the scale-reduced energy will lead to the MAP solution of the original problem.

The *super-pixelization* approach described above has been used for solving problems like object segmentation, stereo and single view reconstruction. A number of sophisticated image partitioning methods have been proposed for merging pixels into super-pixels [5, 19, 15, 4, 24, 20]. However, these methods only consider color or appearance information for partitioning the image. They work under the belief that pixels having similar color (appearance) are likely to take the same label and do not take into account the energy function of the problem. As a result, these methods may group nodes together whose MAP solutions in the original problem are not same. In fact the partitioning computed by these methods may be highly inconsistent with the MAP solution of the original problem. Furthermore, most of these methods are computationally quite expensive and are thus not suitable for our end goal of reducing the computational cost for finding the MAP solution. We will call the super-pixelization methods that do not take the energy to be minimized into account as *energy-agnostic*.

In this paper, we propose a new *energy-aware* method for merging the variables present in a labeling problem. It is based on the linear running-time image partitioning method proposed by Felzenszwalb and Huttenlocher [5] which works by repeatedly merging groups of pixels which have similar appearance. Instead of using appearance, our method uses the terms in the energy associated with the variables to decide if they should be merged or not. We propose and evaluate a number of variable grouping scores that can be computed in constant time $O(1)$. Experimental results show that our methods result in a dramatic reduction in the computational cost and memory requirements (in some cases by a factor of 100) with almost no drop in the accuracy of the final result. Comparative evaluation with the method of [5] reveals that our methods are far superior both in terms of accuracy and running time.

Other Related Work Our work is also related to multi-scale methods for image labeling. These methods solve a large-scale labeling problem defined by a large image (or 3D volume) by first constructing a problem defined on lower-resolution image (or 3D volume) [8, 22, 14, 17]. This effectively means that they partition the image into regular non-overlapping patches. The smaller problem is solved to

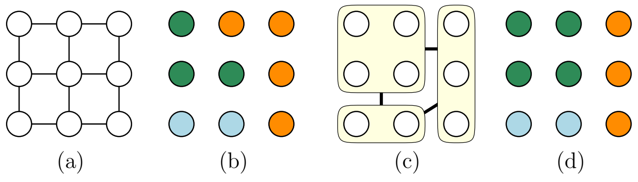


Figure 2. The 9 variables in the original energy (a) are merged to obtain a new energy (c) defined on only 3 grouped variables. The final solution (d) is now computed by solving the smaller problem (c). The solution may differ slightly from the MAP solution (b) of the original problem.

generate a partial labeling for the original high-resolution problem. The partial solution is used to fix the values of most of the variables resulting in an energy defined over few variables. These energies are then minimized using standard energy minimization methods. While effective, these methods are energy-agnostic and it is possible to improve upon them by taking into account the energy function when grouping the variables.

Outline of the Paper The notation and definitions used in the paper are provided in Section 2. Our proposed method for grouping variables is explained in Section 3. The details of our experiments and the analysis of the results are provided in Section 4.

2. Notations and Preliminaries

Most image and volume labeling problems in computer vision are formulated using probabilistic models such as MRF and CRF. Any random field (denoted by $\mathbf{X}$) is defined over a lattice $\mathcal{V} = \{1, 2, \ldots, n\}$ with a neighborhood system $\mathcal{E}$. Each random variable $x_i \in \mathbf{X}$ is associated with a lattice point $i \in \mathcal{V}$ and takes a value from the label set $\mathcal{L} = \{l_1, l_2, \ldots, l_m\}$. The neighborhood system $\mathcal{E}$ of the random field is a set of edges (i, j) which probabilistically links x_i and x_j. Any possible assignment of labels to the random variables will be called a *labeling* (denoted by $\boldsymbol{x}$) which takes values from the set $\mathbf{L} = \mathcal{L}^n$.

The most probable or MAP labeling $\boldsymbol{x}^{\text{opt}}$ of the random field can be computed by maximizing the posterior or alternatively minimizing the energy of the random field as:

$$\boldsymbol{x}^{\text{opt}} = \underset{\boldsymbol{x}\in\mathbf{L}}{\text{argmax}} \Pr(\boldsymbol{x}|\mathbf{D}) = \underset{\boldsymbol{x}\in\mathbf{L}}{\text{argmin}}\, E(\boldsymbol{x}). \qquad (1)$$

where $\mathbf{D}$ is the image data. The random field models used for most vision problems are *pairwise*, *i.e.* their energy function $E : \mathcal{L}^n \to \mathbb{R}$ can be written as a sum of unary ϕ_i and pairwise ϕ_{ij} functions,

$$E(\boldsymbol{x}) = \sum_{i\in\mathcal{V}} \phi_i(x_i) + \sum_{(i,j)\in\mathcal{E}} \phi_{ij}(x_i, x_j). \qquad (2)$$

For image labeling problems, the set $\mathcal{V}$ corresponds to the set of all image pixels, $\mathcal{E}$ is set of all edges between pixels in a 4 or 8 neighborhood, and the random variable x_i denotes the labeling of pixel i of the image.

The problem of minimizing the general functions of the form (2) is NP-hard. However, particular families of problems can be solved exactly in polynomial time. There are also efficient methods for getting approximate solutions for general problems. Graph cuts [11, 2, 3], loopy belief propagation (LBP) [26, 16], tree re-weighted message passing (TRW) [10, 25], max-sum diffusion [27] and FastPD [13] are some popular examples.

3. Energy Based Variable Grouping

Let us first formalize the notion of *variable grouping* applied to an energy minimization problem.

Definition 1 (Variable Grouping) *Given a graph $G = (\mathcal{V}, \mathcal{E})$ with corresponding functions ϕ_i, ϕ_{ij}, a* variable grouping *is a graph $G' = (\mathcal{V}', \mathcal{E}')$ with energy function E' produced by a surjective*[1] *map $m : \mathcal{V} \to \mathcal{V}'$ mapping all vertices in G to vertices in G' and taking the edge set $\mathcal{E}' = \{(s,t) \in \mathcal{V}' \times \mathcal{V}' | \exists (i,j) \in \mathcal{E} : m(i) = s$ and $m(j) = t\}$. The energy function of the grouping is formulated in terms of new unary and pairwise functions $\hat{\phi}_j$, $\hat{\phi}_{st}$,*

$$\begin{aligned} E'(\hat{\boldsymbol{x}}) &= \sum_{i \in \mathcal{V}} \phi_i(\hat{x}_{m(i)}) + \sum_{(i,j) \in \mathcal{E}} \phi_{ij}(\hat{x}_{m(i)}, \hat{x}_{m(j)}) \\ &= \sum_{j \in \mathcal{V}'} \Big(\sum_{i \in m^{-1}(j)} \phi_i \Big)(\hat{x}_j) \\ &\quad + \sum_{(s,t) \in \mathcal{E}'} \Big(\sum_{\substack{(i,j) \in \\ m^{-1}(s) \times m^{-1}(t)}} \phi_{ij} \Big)(\hat{x}_s, \hat{x}_t) \\ &= \sum_{j \in \mathcal{V}'} \hat{\phi}_j(\hat{x}_j) + \sum_{(s,t) \in \mathcal{E}'} \hat{\phi}_{st}(\hat{x}_s, \hat{x}_t). \end{aligned}$$

where $\hat{\boldsymbol{x}}$ is the labeling of a graph G'. The grouping construction is illustrated in Figure 2.

For any variable grouping the resulting graph is smaller in scale than the original graph. If the smaller instance is a "good approximation" to the original graph then we can solve the smaller problem instead and use its solution to produce a solution to the original problem. We make this idea clear in the next definition.

Definition 2 (Solution Recovery) *Given a solution $\hat{\boldsymbol{x}}$ to a variable grouping $G' = (\mathcal{V}', \mathcal{E}')$ with energy E' produced by the map $m : \mathcal{V} \to \mathcal{V}'$ from $G = (\mathcal{V}, \mathcal{E})$ with energy E, we define the* recovered solution *to the original problem as*

$$\hat{\boldsymbol{x}}{\uparrow} = (\hat{\boldsymbol{x}}_{m(i)})_{i \in \mathcal{V}}.$$

The effectiveness of the approximation using the variable grouping can be evaluated in terms of energy: how much did we suffer in terms of the original energy by solving the approximation? We define this as follows.

[1]Surjectivity: $\forall v' \in \mathcal{V}' : \exists v \in \mathcal{V} : m(v) = v'$.

Definition 3 (Approximation Loss) *Given a variable grouping $G' = (\mathcal{V}', \mathcal{E}')$ with E' of $G = (\mathcal{V}, \mathcal{E})$ with E based on the map m, the* approximation loss *is defined as*

$$r(m) = \frac{E'(\hat{\boldsymbol{x}}^{opt})}{E(\boldsymbol{x}^{opt})} = \frac{E(\hat{\boldsymbol{x}}^{opt}{\uparrow})}{E(\boldsymbol{x}^{opt})}, \tag{3}$$

where $\hat{\boldsymbol{x}}^{opt}$ is the optimal solution to the grouped energy and $\boldsymbol{x}^{opt}$ is the optimal solution to the original problem. The equality follows from the construction of E'.

By construction we always have $r(m) \geq 1$. In case we have $r(m) = 1$ the approximation is exact in that it allows us to recover the solution to the original problem.

How to find a variable grouping m that has a small loss? Ideally we would like to explicitly minimize (3) over all possible groupings but this requires us to know $\boldsymbol{x}^{\text{opt}}$ apriori. Because the overall goal of inference is to obtain $\boldsymbol{x}^{\text{opt}}$, this approach is impractical. Moreover, we would like to have a variable grouping method that is much faster than inference itself, producing a good grouping of small size. To this end we take (3) as inspiration to construct a localized weight function for each edge in $\mathcal{E}$. We use this weight function in a fast graph-based clustering method to group variables that are likely to be labeled with the same label. In case the local weight function is expressive enough, then this produces a grouping that has low loss. We describe the details of our approach in the next section.

3.1. Efficient Energy-Based Variable Grouping

We assume we have a local score function $w : \mathcal{E} \to \mathbb{R}$ that measures how dissimilar the two connected nodes are, such that small values indicate a strong similarity, and large values indicate dissimilarity. We can then group variables as follows. We first sort all edges $(i,j) \in \mathcal{E}$ by their weights in ascending order so that edges that link similar variables come first. Initially we assign each node its own group. Then, for each edge in the ordered list we merge nodes together until we have sufficiently reduced the problem size.

This simple procedure, while plausible and computationally efficient, is also *myopic*: the only information used for merging groups of variables is in the local weighting function. Global information such as the size of the group or their internal dissimilarities are ignored.

This resembles the problem of image segmentation: finding globally coherent groups of pixels from local measurements. We therefore use a procedure that was originally proposed for image segmentation to solve our variable grouping problem. We use the *efficient graph-based segmentation method*, proposed by Felzenszwalb and Huttenlocher [5]. The method follows the same procedure as above except that the merging decisions are based on a more global criterion, balancing the size of the group and its internal coherence.

Algorithm 1 Energy based Variable Grouping

```
 1: $(\mathcal{V}', m) = \textsc{VariableGrouping}(G, \phi, \boldsymbol{w})$
 2: Input:
 3:     $G = (\mathcal{V}, \mathcal{E})$, graph instance,
 4:     $\phi_i$, $\phi_{ij}$, node and edge energies,
 5:     $\boldsymbol{w} : \mathcal{E} \to \mathbb{R}$, dissimilarity weights
 6: Output:
 7:     $\mathcal{V}'$, set of grouped variables,
 8:     $m : \mathcal{V} \to \mathcal{V}'$, variable grouping.
 9: Algorithm:
10: $\mathcal{V}' \leftarrow \mathcal{V}, \mathcal{E}' \leftarrow \mathcal{E}$
11: $m \leftarrow \{(i,i) | i \in \mathcal{V}\}$
12: $\pi \leftarrow \text{sort}(\mathcal{E}, w)$ {Sort weights}
13: for $e = 1, \dots, |\pi|$ do
14:     $(i,j) \leftarrow \pi_e$
15:     if $m(i) = m(j)$ then
16:         continue {Already merged}
17:     end if
18:     if $w_{ij} \leq \text{MInt}(\mathcal{C}_i, \mathcal{C}_j)$ then
19:         Merge $\mathcal{C}_i$ and $\mathcal{C}_j$ in $m$, $\mathcal{V}'$
20:     end if
21: end for
```

Algorithm 1 is identical to [5] but uses our variable grouping notation. To make the merging decisions, the method measures a property of each group of variables, the so called the *internal difference*,

$$\text{Int}(\mathcal{C}) = \max_{(i,j) \in \text{MST}(\mathcal{C}, \mathcal{E})} w_{ij},$$

where $MST(\mathcal{C}, \mathcal{E})$ is the minimum-weight spanning tree within the component $\mathcal{C}$ with a set of edges $\mathcal{E}$ in it. Therefore Int($\mathcal{C}$) is small if the component is tightly grouped, *i.e.* most of its internal edge weights are small. When deciding whether to merge two components or to preserve them individually, the method compares the weight of the candidate edge linking component $\mathcal{C}_1$ and $\mathcal{C}_2$ with the internal differences of each component. Therefore, an edge with higher dissimilarity can still be used for merging two group in case these groups are already somewhat incoherent. On the other hand, if the two groups are coherent individually but the candidate edge has a high weight, then the merge is not performed, preserving the coherence of each group. In light of our goal of grouping variables for energy minimization, this is a natural criterion: we want to divide the graph into groups of variables that agree about their labeling while preserving distinct groups that are more likely to take different labels. To make the decision, we us the MInt function of [5], defined as

$$\text{MInt}(\mathcal{C}_1, \mathcal{C}_2) = \min\{\text{Int}(\mathcal{C}_1) + \tau(\mathcal{C}_1), \text{Int}(\mathcal{C}_2) + \tau(\mathcal{C}_2)\}, \tag{4}$$

where $\tau(\mathcal{C}) = \frac{k}{|\mathcal{C}|}$ is an additional size bias depending on a free parameter k.

The method is very efficient and easily implemented in $O(|\mathcal{E}| \log |\mathcal{E}|)$ time and space. An important choice for our application is how to determine the weights w_{ij} from the original energy function. We discuss various options in the next section.

3.2. Weight Functions

We consider three classes of weight functions w_{ij}, i) an image-based baseline method (ImageBaseline), ii) two methods using only unary functions ϕ_i and ϕ_j (UnaryDiff and MinUnaryDiff), and iii) a method combining unary ϕ_i, ϕ_j, and pairwise ϕ_{ij} functions (MeanCompat).

Baseline. In all our experiments, the variables will correspond to pixels of an image. Our simplest baseline is to take the pixel similarity of the image as grouping weights (ImageBaseline), *i.e.* to set $w_{ij}^{\text{img}} = \|P_i - P_j\|$, where $P_i \in \mathbb{R}^3$ is the RGB color vector of pixel i in the image. These weights are energy-agnostic and as we will see from comparing to this baseline there is indeed an advantage in performing a variable grouping on the energy instead of using the image information only.

In addition, this baseline is identical to the Felzenszwalb and Huttenlocher image segmentation method. Therefore solving an energy minimization problem on top of this variable grouping can be seen to correspond to the existing practice of using *image-derived superpixels* as preprocessing step and defining the energy minimization problem on superpixels instead of pixels. Our experiments will show that this existing method is performing poorly because it ignores the properties of the energy function.

Unary-only weighting functions. The unary functions ϕ_i and ϕ_j are typically derived from a discriminative classifier. As such it contains valuable information about the class but can also be used to measure about the task-specific similarity of two nodes i and j. The first unary-only weighting function we propose is

$$w_{ij}^{\text{ud}} = \|\phi_i - \phi_j\| \qquad (\textsc{UnaryDiff}).$$

These weights consider all states of the node equally important. We argue that it makes sense to consider the prefered state of each node. If we take $x_i = \text{argmin}_l\, \phi_i(l)$, $c_i = \phi_i(x_i)$, and likewise, $x_j = \text{argmin}_l\, \phi_j(l)$, $c_j = \phi_j(x_j)$ to denote the prefered states, then we define the weights as the disagreement between the preferences, *i.e.*

$$w_{ij}^{\text{mud}} = \max(\phi_i(x_j) - c_i, \phi_j(x_i) - c_j) \quad (\textsc{MinUnaryDiff}).$$

Pairwise weighting function. The unary-only weighting functions ignore the influence of the pairwise function ϕ_{ij}. This is justified in the energy minimization problems where the unary functions are indicative of the true label. For other problems, however, the pairwise term is important, and including it into the weight function accounts for its influence during grouping. We do so by measuring the difference of the mean energy of agreeing labelings ($x_i = x_j$) versus the difference of the mean energy of disagreeing labelings ($x_i \neq x_j$). This yields the weights (MEANCOMPAT)

$$w_{ij}^{\text{mc}} = \frac{1}{m} \sum_{x_i = x_j} (\phi_{ij}(x_i, x_j)) - \frac{1}{m^2 - m} \sum_{x_i \neq x_j} (\phi_{ij}(x_i, x_j)).$$

4. Experiments and Results

We first discuss our experimental setup and the performance measures we use.

4.1. Setup

We evaluate our method on the Middlebury MRF dataset [23] and the data used in [1].[2] The experiments use real-world computer vision problem instances to examine the typical trade-off between the runtime gained and the accuracy lost due to the variable grouping. The used datasets provide typical labeling problem instances for a number of computer vision tasks, such as binary foreground-background segmentation, color segmentation, semantic object segmentation, and stereo matching [1, 23].

Each type of labeling problem has a different number of labels. In the above datasets we have between three to five problem instances for each type. Images are presented with a different number of labels. For *binary image segmentation* there are two labels — foreground and background — and the task is to segment the foreground object. For the *color segmentation* and the *object segmentation* task we have a small number of labels corresponding to different object classes. The energy functions are constructed using the method proposed in [21]. In the *stereo matching* task we are given two images and reconstruct a disparity map. All energy functions are computed using the public benchmark of Szeliski et al. [23] and we do not attempt to improve on the energy formulation itself. The number of disparity labels in different problem instances were: Tsukuba (16), Venus (20), Cones (60) and Teddy (60).

To apply our variable grouping algorithm to the energy function we compute the graph weights using one of the weight functions. Because each problem has a different range of energy values, we first normalize all unary and pairwise energies into a fixed range before evaluating the weight function. We do this only to compute the weights used for variable grouping, and do not modify the original energy function is both in the full and grouped energy.

For all experiments we set $k = 10$ in $\tau(\mathcal{C}) = \frac{k}{|\mathcal{C}|}$ as the threshold parameter in (4).

[2]Available at http://vision.middlebury.edu/MRF/ and http://cms.brookes.ac.uk/staff/Karteek/data.tgz.

To obtain a complete picture of the available runtime-accuracy tradeoffs we evaluate multiple variable groupings of varying size for each problem instance. In particular we use different rates of reduction in the size of the original problem, measured as "*budget*", where a budget of 100% corresponds to the original problem. A smaller budget specifies the number of variables in the grouped instance as a fraction of the number of variables in the original instance. We use 10 budgets in total: $50\%, 25\%, 12.5\%, \dots, \approx 0.1\%$ as the integer powers of 0.5. A grouped problem with a budget of 0.1% therefore has only 0.1% the number of variables of the original instance.

For both the original and the grouped energies we use two types of inference algorithms to obtain an approximate MAP labeling: sequential tree-reweighted message passing (TRW-S) [10] and loopy belief propagation (LBP) [26]. For each inference method, we use a maximum of 300 iterations for each problem. The results for TRW-S are shown in figure 4. The results obtained with LBP are similar.

For each budget and instance we evaluate the approximate solution $\hat{\boldsymbol{x}}^{\text{opt}}$ against the optimal solution $\boldsymbol{x}^{\text{opt}}$ of the original problem using the following three types of performance measures.

4.2. Evaluation Measures

Our variable grouping allows us to infer an approximate MAP labeling for a given problem instance from the MAP solution of a reduced size problem. Naturally, the smaller the reduced graph size the less accurate the inferred labeling becomes.

We quantify the trade-off between gain in runtime performance versus loss in accuracy using three performance measures, i) the approximation loss $r(m)$ measuring the solution quality with respect to the original energy, ii) the per-pixel agreement of the approximate solution with the MAP solution of the original problem, and iii) the ratio of runtimes of the proposed method (including the variable grouping), versus the runtime of solving the original problem. We now describe the latter two measures in more detail.

Per-pixel agreement. We define a measure of the agreement between the approximate labeling $\hat{\boldsymbol{x}}$ of the reduced problem and the MAP labeling of the original problem as

$$R_{\text{MAP}}(\hat{\boldsymbol{x}}^{\text{opt}}, \boldsymbol{x}^{\text{opt}}) = \frac{\sum_{i \in \mathcal{V}} I((\hat{\boldsymbol{x}}^{\text{opt}} \uparrow)_i = \boldsymbol{x}_i^{\text{opt}})}{|\mathcal{V}|}, \quad (5)$$

where $I(\cdot)$ is the indicator function that returns one when the argument is true, zero otherwise. By definition we have $0 \leq R_{\text{MAP}}(\hat{\boldsymbol{x}}^{\text{opt}}, \boldsymbol{x}^{\text{opt}}) \leq 1$, and in case the measure is one the approximation is exact.

Ratio of runtimes. To quantify the reduction in overall runtime we measure the ratio of the runtime of the proposed method versus the runtime of solving the full problem. Formally, we take

$$R_{\text{time}} = \frac{T_{\text{group}}(G) + T_{\text{MAP}}(G')}{T_{\text{MAP}}(G)} \qquad (6)$$

where $T_{\text{group}}(G)$ is the runtime to group the variables of G to produce G', and $T_{\text{MAP}}(\cdot)$ is the time to solve for the MAP solution of its argument.

4.3. Results and Discussion

Our results are visualized for all instances and budgets in Figures 3 and as average performance for each problem type in Figure 4.

In Figure 3 we use color-coded label maps to visualize the approximate MAP solutions obtained from our variable grouping. The leftmost solution in column (b) shows the ground truth provided with the data set, and the column (c) adjacent to it the MAP solution of the full sized problem.

The following columns (d) to (g) show that accuracy is affected as the budget is decreased; but it is somewhat surprising how small a variable budget is sufficient to obtain accurate labelings across all tasks. For example, consider the quality of the solutions shown in column (e). These use only 12.5% of the variables and around 1/5'th of the runtime of the full MAP solution including variable grouping. Despite this, the result is almost indistinguishable. Only for the stereo matching task with a large number of labels there are visible differences. Naturally, for very small budgets, such as the 0.1% shown in column (g) there is indeed a visible deterioration in the labeling.

In Figure 4 we visualize the quantitative results of our method. For each of the four different problem types we show the average performance for all instances of that type using the performance measures described in Section 4.2.

The performances of our proposed weight functions are differ between the different problem types. The first observation is that the UNARYDIFF, MINUNARYDIFF, and MEANCOMPAT functions consistently show a higher agreement and lower loss than the IMAGEBASELINE function in all problems. What this means is that variable grouping on the energy function is superior to superpixelization on the image, as it can take into account the task-specific properties of the energy minimization task.

The second observation is that among the different weight functions, the MINUNARYDIFF function gave the best performance in terms of per-pixel agreement in the binary foreground-background segmentation task and in the object segmentation task. Here the unary cues are strong and sufficient to make safe grouping decisions. On the other hand, the MEANCOMPAT weight function worked better in color-based segmentation and stereo matching, because here there are multiple labels and both the unary and the pairwise potentials are needed.

Finally, the reduction in runtime is comparable for all weight functions: the overall runtime is dramatically reduced, and the overall inference time including the variable grouping can be as low as 1% compared to inference for the original problem. Eventually, for very small budgets the overall runtime increases again essentially because the energy minimization is performed very quickly but clustering the graph has a small overhead.

To highlight the performance once more: in the most extreme example, the binary foreground-background segmentation, our variable grouping gave 99.57% per-pixel agreement at a budget of 0.1%, reducing the overall runtime by a factor of 100. For the other tasks the budget needs to be larger to achieve a similar accuracy, but as can be seen from the graphs, for the color-based segmentation and object segmentation tasks one can use a budget of 25% corresponding to a five-fold speedup without noticable differences in the solutions obtained. The stereo matching results are not as good, but note that we count labels deviating by only one depth level as an error. Qualitatively the labeling is still acceptable, as apparent from Figure 3.

5. Conclusion

Our method is simple and widely applicable to MAP inference problems in computer vision. It provides a complementary method to speed up any existing inference method. The user can select from a wide range of a trade-offs between obtained speedup and the retained labeling accuracy.

Our method demonstrates that performing a variable grouping on top of a problem-specific energy function is superior to making such a grouping decision apriori and agnostic about the problem type, as is commonly done when using the image to produce a superpixelization.

We plan to release a wrapper compatible with the TRW-S interface, so as to make the benefits of our method widely available without requiring changes to existing code.

References

[1] K. Alahari, P. Kohli, and P. H. S. Torr. Dynamic hybrid algorithms for MAP inference in discrete MRFs. *PAMI*, 2010. 1, 5

[2] E. Boros and P. Hammer. Pseudo-boolean optimization. *Discrete Applied Mathematics*, 2002. 1, 3

[3] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *PAMI*, 2001. 1, 3

[4] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *PAMI*, 24(5):603–619, 2002. 2

[5] P. Felzenszwalb and D. Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 59(2):167–181, 2004. 2, 3, 4

[6] X. He, R. Zemel, and D. Ray. Learning and incorporating top-down cues in image segmentation. In *ECCV*, 2006. 1

[7] D. Hoiem, A. Efros, and M. Hebert. Automatic photo pop-up. *SIGGRAPH*, 24(3):577–584, 2005. 1

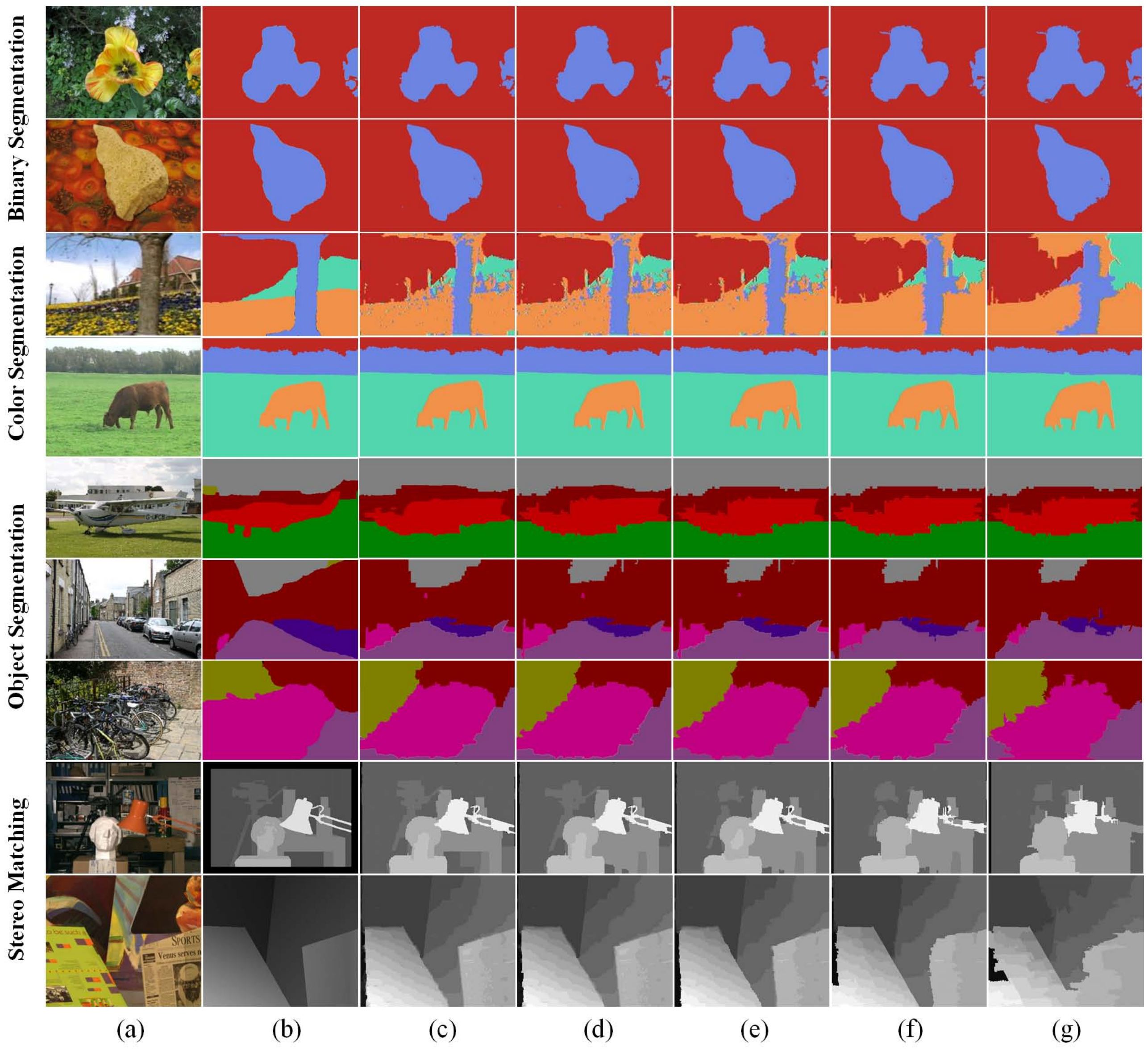


Figure 3. Solutions obtained using variable groupings. Each row corresponds to a different instance; each column to a different budget, (a) original image, (b) ground truth, (c) original MAP solution, (d) approximate MAP solution with grouping budget 50%, (e) 12.5%, (f) 1.56%, and (g) 0.1%. All solutions were obtained by TRW-S MAP inference.

[8] P. Kohli, V. Lempitsky, and C. Rother. Uncertainty driven multi-scale optimization. In *DAGM*, 2010. 2

[9] P. Kohli and P. Torr. Efficiently solving dynamic Markov random fields using graph cuts. In *ICCV*, 2005. 1

[10] V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *PAMI*, 28(10):1568–1583, 2006. 1, 3, 5

[11] V. Kolmogorov and R. Zabih. Multi-camera scene reconstruction via graph cuts. In *ECCV*, 2002. 1, 3

[12] N. Komodakis. Towards more efficient and effective LP-based algorithms for MRF optimization. In *ECCV*, 2010. 1

[13] N. Komodakis, G. Tziritas, and N. Paragios. Fast, approximately optimal solutions for single and dynamic MRFs. In *CVPR*, 2007. 1, 3

[14] H. Lombaert, Y. Sun, L. Grady, and C. Xu. A multilevel banded graph cuts method for fast image segmentation. In *ICCV*, 2005. 2

[15] A. P. Moore, S. Prince, J. Warrell, U. Mohammed, and G. Jones. Superpixel lattices. In *CVPR*, 2008. 2

[16] J. Pearl. Fusion, propagation, and structuring in belief networks. *Artif. Intell.*, 29(3):241–288, 1986. 1, 3

[17] P. Pérez and F. Heitz. Restriction of a Markov random field on a graph and multiresolution statistical image modeling. *IEEE Transactions on Information Theory*, 1996. 2

[18] A. Rabinovich, S. Belongie, T. Lange, and J. Buhmann. Model order selection and cue combination for image segmentation. In *CVPR*, 2006. 1

[19] X. Ren and J. Malik. Learning a classification model for segmentation. In *ICCV*, 2003. 2

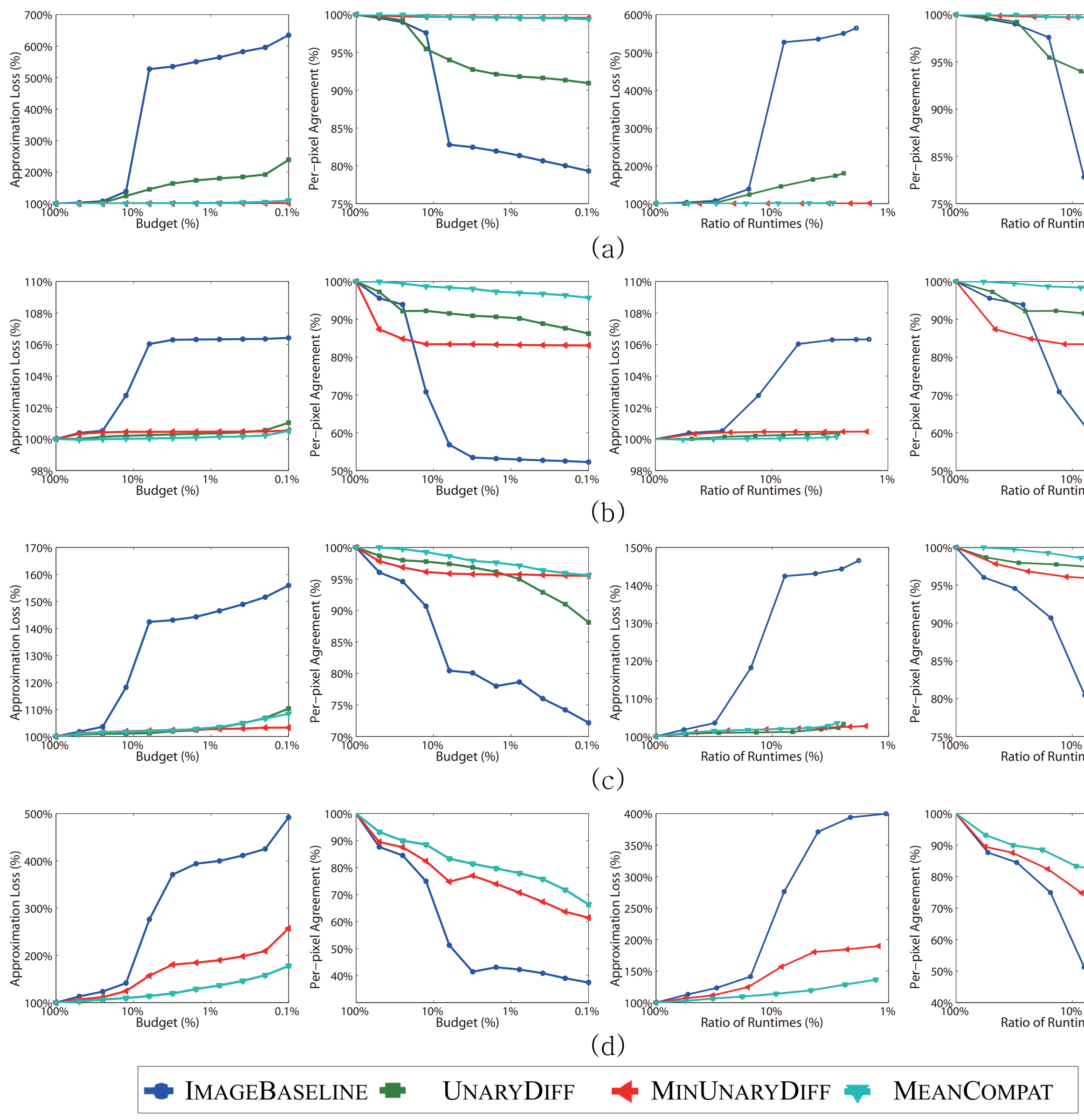


Figure 4. Evaluation measures: Approximation loss, per-pixel agreement and ratio of runtimes (solution from TRW-S MAP inference). Average performance for (a) binary segmentation, (b) color-based segmentation, (c) object segmentation, (d) stereo matching.

[20] J. Shi and J. Malik. Normalized cuts and image segmentation. *PAMI*, 22(8), 2000. 1, 2

[21] J. Shotton, J. Winn, C. Rother, and A. Criminisi. *TextonBoost*: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *ECCV (1)*, pages 1–15, 2006. 5

[22] A. K. Sinop and L. Grady. Accurate banded graph cut segmentation of thin structures using laplacian pyramids. In *MICCAI*, 2006. 2

[23] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A comparative study of energy minimization methods for Markov random fields. In *ECCV*, 2006. 5

[24] O. Veksler, Y. Boykov, and P. Mehrani. Superpixels and supervoxels in an energy optimization framework. In *ECCV*, 2010. 2

[25] M. Wainwright, T. Jaakkola, and A. Willsky. Map estimation via agreement on trees: message-passing and linear programming. *IEEE Transactions on Information Theory*, 51(11):3697–3717, 2005. 1, 3

[26] Y. Weiss and W. Freeman. On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs. *Transactions on Information Theory*, 2001. 1, 3, 5

[27] T. Werner. A linear programming approach to max-sum problem: A review. Research Report CTU–CMP–2005–25, Czech Technical University, December 2005. 1, 3