# Matching Bags of Regions in RGBD images

Hao Jiang
Boston College, USA

We propose a method to match a bag of unstructured regions between two RGBD images. Instead of relying on segmentation algorithms to give consistent partitions, we resort to a large set of region candidates from different segmentation methods. Our method automatically selects the best set of regions and match them simultaneously. Fig. 1 shows an example of region matching using the proposed method. The fundamental problem of our region selection and matching task is min-cost bipartite matching with global region constraints as shown in Fig. 1(e). Traditional bipartite matching minimizes the matching costs with the constraint that each site from one image at most matches one site on the other image. To match a bag of overlapping regions, the optimization has to satisfy more conditions such as the max-covering constraints, overlapping penalty constraints, and the number constraints. Previous min-cost bipartite matching methods, such as the Hungarian algorithm, cannot be directly used any more. Region matching methods [1, 2] also cannot be used to solve the proposed problem. Finding the global optimal solution of the proposed region matching problem is a challenging combinatorial problem, which has not been studied before.

We formulate region matching between RGBD images as the following integer linear program.

$$\begin{aligned}\min \sum_{i\in I,\, j\in J} c_{i,j} z_{i,j} + \sum_{i\in I} (\phi l_i + \mu l_i f_i + \gamma) x_i + \\ \sum_{j\in J} (\phi r_j + \mu r_j g_j + \gamma) y_j - \eta\Big(\sum_{m\in\mathcal{M}} p_m + \sum_{n\in\mathcal{N}} q_n\Big) \\ \text{s.t.}\quad x_i = \sum_{j\in J} z_{i,j},\ y_j = \sum_{i\in I} z_{i,j} \sum_{i\in P_m} x_i \ge p_m, \forall m\in\mathcal{M},\ \sum_{j\in Q_n} y_j \ge q_n, \forall n\in\mathcal{N} \\ x,y,z = 0 \text{ or } 1,\ 0\le p,q\le 1\end{aligned} \tag{1}$$

We use binary variable $z_{i,j}$ to indicate the matching from region $i\in I$ in source image to region $j\in J$ in target image. If the matching is true $z_{i,j}=1$ and otherwise 0. $c_{i,j}$ is the cost of matching region $i$ to region $j$. We also introduce variables $x_i$ and $y_j$, which are the region selection variables on image one and two for region $i$ and $j$ respectively. If a region is selected in the matching, the corresponding selection variable is 1, and otherwise 0. We require the regions selected for matching to have small overlaps by penalizing the overall region size. $l_i$ and $r_j$ are number of pixels in region $i$ and $j$ in source and target images. We also enforce the max-covering constraints. We partition the source and target images into small tiles. We use $p_m$ to indicate whether tile $m$ in source image is covered by a selected region; $p_m = 1$ if the tile is covered and otherwise $p_m = 0$. Similarly we denote $q_n$ as the indicator variable for tile $n$ in target image. In the formulation, $f_i$ is the concavity of region $i$ in image one and $g_j$ is the concavity of region $j$ in image two. We penalize the overall concavity of selected regions. $\phi, \mu, \gamma$ and $\eta$ are weight coefficients.

The integer linear program is a hard combinatorial problem. We solve it with a branch and bound method. Our integer program has the following structure:

$$\min(c^T z - e^T w) \text{ s.t. } Az\le 1,\ Bz\ge w,\ 0\le w\le 1,\ z \text{ is binary}, \tag{2}$$

in which $c$, for which we abuse the notation a bit, is determined by the local matching cost, the area intersection cost, the concavity cost and the number cost, $z$ denotes the vector of matching variables, $e$ is the weight for the covering variables $w$, which are $p$ and $q$ in the original notation. The $x$ and $y$ terms have been absorbed into the $z$ terms. $Az\le 1$ is the bipartite matching constraint, $Bz\ge w$ is the max-covering constraint, and other constraints set the bounds for variables.

Its Lagrangian relaxation, $\max_\lambda \min[c^T z - e^T w + \lambda^T(w - Bz)]$, is much easier to solve than the original integer program. For each given $\lambda$, the internal minimization can be separated:

$$\text{[P1]: } \min(c^T - \lambda^T B)z,\ \text{ s.t. } Az\le 1, z \text{ is binary}. \tag{3}$$

$$\text{[P2]: } \min(\lambda^T - e^T)w,\ \text{ s.t. } 0\le w\le 1\,. \tag{4}$$

This is an extended abstract. The full paper is available at the Computer Vision Foundation webpage.

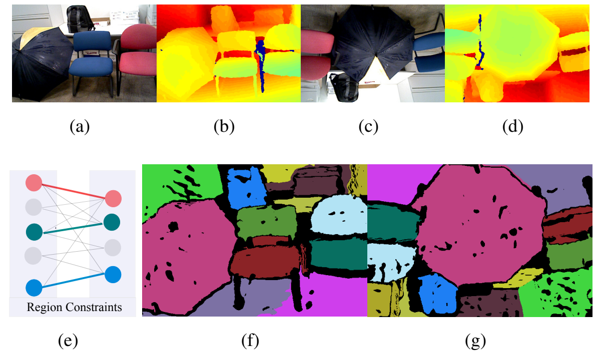


Figure 1: We extract candidate regions to form bags of candidates on the source (a, b) and target (c, d) RGBD images. Our method optimizes the region selection and matching using the graph model in (e) and gives the matching result in (f, g).

For P1, the element of $z$ is 0 if its coefficient is non-negative; other elements of $z$ with negative weights are determined by the min-cost bipartite matching. P2 can be minimized simply by setting $w$ to the upper or lower bound based on the sign of the coefficients; if the coefficient $\lambda_k - e_k \ge 0$, $w_k = 0$, otherwise $w_k = 1$.

The dual problem can be solved using the standard subgradient method that alternates between the optimization of P1 and P2 and updating the $\lambda$. We initialize $\lambda$ to a vector of large numbers so that most of the elements in $(c^T - \lambda^T B)$ is negative. After optimizing P1 and P2 to obtain solution to $z$ and $w$ using the current $\lambda$, we let $\lambda \leftarrow \max\{0, \lambda + \delta(w - Bz)\}$. The iteration goes on until the relative energy increment of the Lagrangian relaxation is less than a threshold. The optimum of the Lagrangian relaxation of our problem equals that of the linear program relaxation.

Based on the Lagrangian dual, we optimize the solution using a branch and bound method. We need to determine on what variables we generate the search tree branches. One apparent choice is that we can branch on the matching variables $z$. However, there can be huge number of $z$ because it is a quadratic function of the number of region candidates. We branch on the region selection variables $x$ and $y$ on image one and two instead. By fixing $x$ and $y$ to 1 or 0, we enforce that some regions have to be part of the matching and some have to be excluded. This has a big advantage because their number is much smaller than that of the matching variables.

The Lagrangian dual can still be computed efficiently for each search tree node. Each tree node fixes some $x$ and $y$ to 1 or 0, and introduces extra constraints $Dz = d$, where matrix $D$ is determined by $x_i = \sum_j z_{i,j}$ and $y_j = \sum_i z_{i,j}$, and $d$ is a vector of 1 and 0s. If we treat this as a complicated constraint, the Lagrangian dual of the search tree node is

$$\max_{\lambda,\xi}\{\min_{z,w}[(c^T - \lambda^T B + \xi^T D)z + (\lambda^T - e^T)w - \xi^T d]\} \tag{5}$$
$$\text{s.t. } Az\le 1, 0\le w\le 1, \text{ and } z \text{ is binary}.$$

We can still decompose the problem into P1 and P2. P1 can be reduced to a min-cost matching problem and P2 can be solved by assigning the upper or lower bound. We use the subgradient method to determine $\xi$, which is similar to how we deal with $\lambda$. For $\xi$, we update it using $\xi \leftarrow \xi + \delta_\xi(Dz - d)$, where $\delta_\xi$ is a positive step size. Note that the Lagrangian relaxation still gives the same bound as the linear program relaxation of the original problem at each search tree node.

[1] Sinisa Todorovic, Narendra Ahuja, "Region-Based Hierarchical Image Matching", IJCV, vol.78, no.1, pp. 47-66, 2008.

[2] S. Hickson, S. Birchfield, I. Essa, and H. Christensen, "Efficient hierarchical graph-Based segmentation of RGBD videos", CVPR 2014.