Diverse M-Best Solutions by Dynamic Programming Supplementary

Carsten Haubold[1], Virginie Uhlmann[2], Michael Unser[2], Fred A. Hamprecht[1]

[1] University of Heidelberg, IWR/HCI, 69115 Heidelberg, Germany.
[2] École Polytechnique Fédérale de Lausanne (EPFL), BIG, 1015 Lausanne, Switzerland.

1 Optimal Second Best Tree Solutions: Graph Construction

We formally construct the auxiliary directed graph $\tilde{G} = (\tilde{\mathcal{V}}, \tilde{\mathcal{E}})$ composed of two instances of the original directed graph $G = (\mathcal{V}, \mathcal{E})$ stacked up vertically. We address the lower layer with index 1, and the upper one with index 2. The new set of nodes and edges is given by

$$\tilde{\mathcal{V}} := \{1, 2\} \times \mathcal{V},$$
$$\tilde{\mathcal{E}} := \mathcal{E}^1_{\text{in}} \cup \mathcal{E}^2_{\text{in}} \cup \mathcal{E}_{\text{jump}} \cup \mathcal{E}_{\text{cross}}.$$

In the following, the first and second layer copies of an original node $v \in \mathcal{V}$ are written as v^1 and v^2, respectively.

$$\mathcal{E}^1_{\text{in}} := \{(u^1, v^1) | (u, v) \in \mathcal{E}\} \tag{1}$$
$$\mathcal{E}^2_{\text{in}} := \{(u^2, v^2) | (u, v) \in \mathcal{E}\} \tag{2}$$
$$\mathcal{E}_{\text{jump}} := \{(v^1, v^2) | v \in \mathcal{V}\} \tag{3}$$
$$\mathcal{E}_{\text{cross}} := \{(u^1, v^2) | (u, v) \in \mathcal{E}\} \tag{4}$$

Edges are duplicated for each layer (1), (2). Additionally, we introduce *layer-jump-edges* (3) that directly go from any node v^1 in layer 1 to its duplicate v^2 in layer 2. Lastly, we add edge duplicates that originate in layer 1 and go to layer 2. These *layer-crossing-edges* (4) are needed for message passing at branching points. Unary and pairwise potentials $\tilde{\theta}$ are also replicated for all nodes $v \in \mathcal{V}$ and edges $(u, v) \in \mathcal{E}$ as

$$\forall v \in \mathcal{V} \;\; \tilde{\theta}_{v^1} := \theta_v, \, \tilde{\theta}_{v^2} := \theta_v$$
$$\forall u, v \in \mathcal{E} \;\; \tilde{\theta}_{u^1 v^1} := \theta_{uv}, \, \tilde{\theta}_{u^2 v^2} := \theta_{uv}, \tilde{\theta}_{u^1 v^2} := \theta_{uv}.$$

To run dynamic programming on this new graph $\tilde{G}$, messages are propagated starting at all leaves in layer 1, towards the designated root $r^2 \in \tilde{\mathcal{V}}$ on the upper layer. From every node v^1 in the lower layer, a message – embodying the partial solution of the subtree rooted at v^1 in layer 1 – is propagated in three directions: directly to its successors within layer 1, crossing layers to the successors' duplicates in the upper layer, and as a jump to this node's duplicate v^2 subject to a user-specified jumping criterion.

In summary, in the lower layer, standard messages are being sent as in the lowest energy solution. In the upper layer, every junction point is reached by three kinds of

messages: those from within layer 2, those that are incoming from predecessors in layer 1, and those along layer jump edges between node duplicates. At junction points in layer 2, these incoming messages from both layers must be combined. The important requirement for a valid configuration of any layer 2 junction point is that *at least one* of the incoming messages must have come from layer 2 (we denote this *nonempty* set of predecessor nodes as L_2), and must thus have *jumped* to layer 2 in the subtree rooted at this junction point. We therefore change the DP rules in layer 2 to

$$E_{v^2}(\mathbf{x}_{v^2}) := \min \Bigg(\tilde{\theta}_{v^1 v^2}(\mathbf{x}_{v^1}, \mathbf{x}_{v^2}) + E_{v^1}(\mathbf{x}_{v^1}), \tag{5}$$

$$\tilde{\theta}_{v^2}(\mathbf{x}_{v^2}) + \min_{\substack{L_2 \subseteq \overleftarrow{N}(v) \\ |L_2| \geq 1}} \sum_{u \in L_2} \min_{\mathbf{x}_{u^2}} \left[\tilde{\theta}_{u^2 v^2}(\mathbf{x}_{u^2}, \mathbf{x}_{v^2}) + E_{u^2}(\mathbf{x}_{u^2}) \right]$$

$$+ \sum_{u \in \overleftarrow{N}(v) \setminus L_2} \min_{\mathbf{x}_{u^1}} \left[\tilde{\theta}_{u^1 v^2}(\mathbf{x}_{u^1}, \mathbf{x}_{v^2}) + E_{u^1}(\mathbf{x}_{u^1}) \right] \Bigg). \tag{6}$$

Compared to the standard dynamic programming rules, we now have two options instead of one in layer 2. Firstly, we can reach the node by a layer jump. Note that, in case of a jump (5), we do not account for $\tilde{\theta}_{v^2}(\mathbf{x}_{v^2})$ as $E_{v^1}(\mathbf{x}_{v^1})$ already contains this term. Alternatively, at least one of the incoming messages is coming from layer 2 using edges from $\mathcal{E}^2_{\text{in}}$ (6), while the remaining messages may cross layers and originate from $\mathcal{E}_{\text{cross}}$.

If we choose to set all layer-jump-edge potentials to zero, every vertex qualifies as jump location and we obtain at the root the same solution we would get in the original graph G. If we want to find the second best solution with a Hamming distance of 1, we set the jump potentials of all states used by the previous solution to infinity. Then, a jump can only happen when the current solution differs from the previous one at this node. In addition, note that the duplicates of all leaf nodes v^2 must be reached via a layer jump. This can be achieved by setting their unaries to infinity $\tilde{\theta}_{v^2} := \infty$.

2 Approximate Diverse M-Best Solutions

In (5) and (6), we already considered edges from layer 1 and 2 such that at least one of them came from layer 2 if that specific node was not used for a jump. In the case of $k+1$ layers, we define the set of admissible incoming edge combinations $\mathscr{A}$. We thus

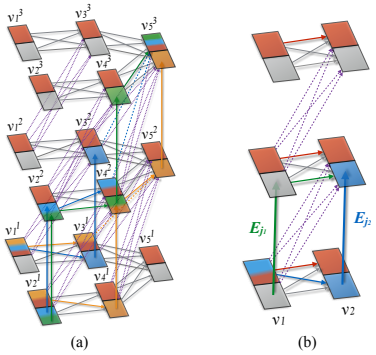


Fig. 1. (a) Visualization of different ways of obtaining a Hamming distance of 2, where the red states show the previous solution. To reach v_5^3, one can jump two times at different nodes (green) and arrive in layer 3, jump once before to layer 2 and then to 3 at v_5 (orange), or combine two incoming branches from layer 2 to get to layer 3 (blue). **(b)** Counterexample for $k = 2$. If $E_{j_2} < E_{j_1}$, the minimization will pick the predecessors shown in blue, which prevents the algorithm from jumping to layer 3 and finding a valid solution.

generalize the update equation as follows:

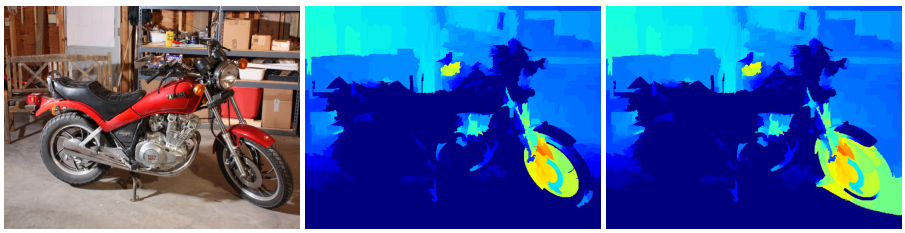
$$\begin{aligned} E_{v^N}(\mathbf{x}_{v^N}) := \min \Big(& \tilde{\theta}_{v^{N-1}v^N}(\mathbf{x}_{v^{N-1}}, \mathbf{x}_{v^N}) + E_{v^{N-1}}(\mathbf{x}_{v^{N-1}}) \\ & + \infty \cdot \delta[\mathrm{Pred}_{v^{N-1}}(\mathbf{x}_{v^{N-1}}) == (v^{N-2}, \mathbf{x}_{v^{N-2}})], \\ & \tilde{\theta}_{v^N}(\mathbf{x}_{v^N}) + \min_{\mathcal{A} \in \mathscr{A}_{v^N}} \min_{\mathbf{x}_a} \sum_{a \in \mathcal{A}} \tilde{\theta}_{a v^N}(\mathbf{x}_a, \mathbf{x}_{v^N}) + E_a(\mathbf{x}_a) \Big) \end{aligned} \tag{7}$$

$$\begin{aligned} \mathscr{A}_{n^N} = \Bigg\{ & \left(u_1^{l_1}, u_2^{l_2}, ..., u_{|\overleftarrow{N}(v)|}^{l_{|\overleftarrow{N}(v)|}} \right) \Bigg| \, u_i^{l_i} \in \overleftarrow{N}(v), \\ & l_i \in \{1, ..., N\}, \sum_{i=1}^{|\overleftarrow{N}(v)|} (l_i - 1) \geq N - 1 \Bigg\} \end{aligned} \tag{8}$$

To prevent two successive jumps at the same variable *(a)*, one must incorporate a check whether a node was reached by a jump. We thus include a dependence on the previous step. We denote by $\mathrm{Pred}_v(\mathbf{x}_v)$ the predecessor node of v and its state on the best path to reach v's state $\mathbf{x}_v$. To model that the cumulative number of jumps to reach layer N must be $N - 1$ *(b)* at each junction on layer $N > 1$, (8) defines $\mathscr{A}$ as the set of admissible combinations of selecting incoming nodes $u_i \in \overleftarrow{N}(v)$ from layers l_i. Some admissible sets are visualized in Figure 1**a**.

3 Applications and Experiments

Disparity Map Estimation from Stereo Images: To generate different disparity maps from stereo images, we build a minimal spanning tree of the pixel grid graph, using



(a) (b) (c)

Fig. 2. Exploring diverse solutions for disparity map estimation, on an image from the Middlebury benchmark [1] resized to 741×500. **(a)** Left view of the motorbike image pair, and corresponding **(b)** best solution found by [2] which struggles inside the front wheel. **(c)** Enforcing a large Hamming distance (here 13000) reveals that the area around the front wheel could have been matched differently, exposing ambiguities in the estimation process.

the intensity gradient as edge weight as in [2]. Neighboring pixels are connected by an edge whenever they have similar intensities. Those that are not similar are not linked and hence are not penalized when generating depth discontinuities. We allow disparities of up to 40 pixels in either direction while computing matching costs on patches of 11×5 pixels, and use a (non-truncated) quadratic attractive potential on the edges. While this setup is far from state-of-the-art in stereo, it demonstrates that our approach scales to large trees with many labels. We used the proposed diversity accumulation method where one unit of diversity is collected at every state that is at least a distance of 5 away from the previous solution in label space, and requested a large amount of diversity to obtain visually different depth maps.

References

1. Scharstein, D., Hirschmüller, H., Kitajima, Y., Krathwohl, G., Nešić, N., Wang, X., Westling, P.: High-resolution stereo datasets with subpixel-accurate ground truth. In: German Conference on Pattern Recognition. pp. 31–42. Springer (2014)
2. Veksler, O.: Stereo correspondence by dynamic programming on a tree. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). vol. 2, pp. 384–390. San Diego, CA, USA (20-25 June, 2005)