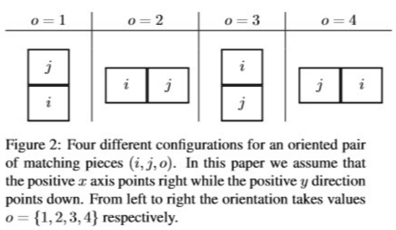
Jigsaw Puzzle

Fuzuo Zhang zfz_11@163.com

January 3, 2019

This is an application of **convex relaxation**.

1 Assumption



(a) four orientations

$$\delta_o^x = \begin{cases} 0 & \text{if } o = 1 \\ -1 & \text{if } o = 2 \\ 0 & \text{if } o = 3 \\ 1 & \text{if } o = 4 \end{cases}$$

$$\delta_o^y = \begin{cases} 1 & \text{if } o = 1 \\ 0 & \text{if } o = 2 \\ -1 & \text{if } o = 3 \\ 0 & \text{if } o = 4 \end{cases}$$

(b) desired offset:namely $\delta_0^x = x_i - x_j, \delta_o^y = y_i - y_j$

Figure 1: Assumption

2 Calculation Procedure

2.1 Compute D_{ijo}

We define D_{ijo} to be the MGC distance between pieces i and j with orientation o1(a).We give the calculation of MGC distance for the example when $o = 2$: Before the formulations, we give the assumption: *P:size of the image-block;c:color channel*.

$$G_{iL}(p,c) = x_i(p,P,c) - x_i(p,P-1,c) \tag{1}$$

$$\mu_{iL}(c) = \frac{1}{P}\sum_{p=1}^{P} G_{iL}(p,c) \tag{2}$$

$$G_{ijLR}(p,c) = x_j(p,1,c) - x_i(p,P,c) \tag{3}$$

$$D_{LR}(x_i,x_j) = \sum_{p=1}^{P}(G_{ijLR}(p) - \mu_{iL})S_{iL}^{-1}(G_{ijLR}(p) - \mu_{iL})^T \tag{4}$$

where the 3×3 covariance S_{iL} is estimated form G_{iL} captures the relationship of the gradients near the edge of the jigsaw piece between the color channels. To avoid numerical problems related to the inversion of S and the inherent issues of quantized pixel values, we include nine"dummy gradients" in the calculations, in the real project, I use $1e-6 \times [[0,0,1],[1,1,1],[1,0,0]]$.

2.2 Compute w_{ijo}

The matching weight w_{ijo} associated with the oriented pair (i,j,o) can be computed as:

$$w_{ijo} = \frac{min(min_{k\neq i}(D_{kjo}), min_{k\neq j}(D_{iko}))}{D_{ijo}} \tag{5}$$

2.3 Compute $U^{(k)}$

Given the entire set of puzzle pieces $V = \{i|i = 1, ..., N \times M\}$

$$U^{(0)} = U = \{(i,j,o), \forall i \in V, \forall j \in V, \forall o \in \{0,1,2,3\}\} \tag{6}$$

$$R^{(k)} = \{\forall(i,j,o) \in A^{(k)} : |x_i - x_j - \delta_o^x| \geq 10^{-5}\} \tag{7}$$

$$U^{(k)} = U^{(k-1)} - R^{(k)} \tag{8}$$

2.4 Compute $A^{(k)}$

$$A^{(k)} = \{(i,j,o) \in U^{(k)} : j = \underset{j:(i,j,o)\in U^{(k)}}{\arg\min}\ D_{ijo}\} \tag{9}$$

2.5 Solve Convex Problem

$$\begin{aligned} \underset{x,h}{\arg\min} \quad & \sum_{(i,j,o)\in A^{(k)}} w_{ijo}h_{ijo} \\ \textit{subject to} \quad & h_{ijo} \geq x_i - x_j - \delta_o^x, \quad (i,j,o) \in A^{(k)} \\ & h_{ijo} \geq -x_i + x_j + \delta_o^x, \quad (i,j,o) \in A^{(k)} \end{aligned} \tag{10}$$

3 Algorithm Procedure

Algorithm 1: LP based Type 1 jigsaw puzzle solver

Input: Scrambled jigsaw puzzle pieces
Output: Assembled image
Initialisation:Generate initial pairwise matches $U^{(0)}$(6) ;
Compute $A^{(0)}$ according to (9);
Solve (10) to get the initial solution $x^{(0)}, y^{(0)}$;
while *not converged* **do**
    Generate Rejected Matches $R^{(k)}$ using (7);
    Update $U^{(k)}$ (6) by discarding $R^{(k)}$;
    Compute pairwise matches $A^{(k)}$ according to eqrefeq:A;
    Solve (10), get new solution $x^{(k)}, y^{(k)}$
end
Trim and fill as necessary to get rectangular shape

References

[1] Rui Yu, Chris Russell & Lourdes Agapito.*Solving Jigsaw Puzzles with Linear Programming*.2015.

[2] Andrew C. Gallagher & Rochester.*Jigsaw Puzzles with Pieces of Unknown Orientation*.2012.