Solving Jigsaw Puzzles with Linear Programming

Rui Yu
R.Yu@cs.ucl.ac.uk
Chris Russell
C.Russell@cs.ucl.ac.uk
Lourdes Agapito
L.Agapito@cs.ucl.ac.uk

Department of Computer Science
University College London
London, UK

In this paper we propose a novel Linear Program (LP) based assembly strategy. We show that this LP formulation is a convex relaxation of the original discrete non-convex jigsaw problem. Instead of solving the difficult NP-hard problem in a single optimization step, we introduce a sequence of LP relaxations. Starting with an initial set of pairwise matches, the method increasingly builds larger and larger connected components that are consistent with the LP. Our LP formulation naturally addresses the so-called Type 1 puzzles[1], where the orientation of each jigsaw piece is known in advance and only the location of each piece is unknown. However, we show that our approach can be directly extended to the more difficult Type 2 puzzles, where the orientation of the pieces is also unknown, by first converting it into a Type 1 puzzle with additional pieces.

As shown in Figure 2, given an oriented pair of matching jigsaw pieces \((i, j, o)\) there are four different possibilities for their relative orientation \(o\). Accordingly, we define 4 matching constraints on their relative positions on the puzzle grid \((x_i, y_i)\) and \((x_j, y_j)\), where \(x_i\) and \(y_i\) indicate the horizontal and vertical positions of piece \(i\). We define the desired offsets \(\delta^0_x, \delta^0_y\) between two pieces in \(x\) and \(y\) for an oriented pair \((i, j, o)\), as shown in Figure 2.

Given the entire set of puzzle pieces \(U = \{i \mid i = 1, \ldots, N \times M\}\), where \(M\) and \(N\) are the dimensions of the puzzle, we define the set \(U = \{(i, j, o) \mid \forall i \in V, \forall j \in V, \forall o \in \{0, 1, 2, 3\}\}\) as the universe of all possible oriented matches \((i, j, o)\) between pieces (see figure (2)). We define \(D_{ijo}\) to be the MCG distance (see [1] for full definition) between pieces \(i\) and \(j\) with orientation \(o\) (see figure 2). The matching weight \(w_{ijo}\) associated with the oriented pair \((i, j, o)\) can now be computed as

\[
w_{ijo} = \frac{\min(\delta\min_{k < i}(D_{ijo}), \min_{k < j}(D_{kjo}))}{D_{ijo}},
\]

i.e. the inverse ratio between \(D_{ijo}\) and the best alternative match. Note that these weights are large when the matching distance between pieces is relatively small and vice versa.

Instead of solving the original NP-hard jigsaw problem, we relax the problem as a linear program:

\[
C^+(x, y) = C^+(x) + C^+(y)
\]

\[
= \sum_{(i, j, o) \in U} w_{ijo}|x_i - x_j - \delta^0_x|_1
\]

\[
+ \sum_{(i, j, o) \in U} w_{ijo}|y_i - y_j - \delta^0_y|_1
\]

In practice, this relaxation does not give rise to a plausible solution as the \(L_1\) norm acts in a similar way to the weighted median, and results in a compromised solution that overly favours weaker matches. Instead we consider the same objective over an active subset \(A \subseteq U\).

\[
C^+(x, y) = C^+(x) + C^+(y)
\]

\[
= \sum_{(i, j, o) \in A} w_{ijo}|x_i - x_j - \delta^0_x|_1
\]

\[
+ \sum_{(i, j, o) \in A} w_{ijo}|y_i - y_j - \delta^0_y|_1
\]

Note that if \(A\) was chosen as the set of ground-truth neighbors, solving this LP would return the solution to the jigsaw, and so finding the optimal set \(A\) is N.P. hard. Instead we propose a sequence of estimators of \(A\), which we write as \(A(k)\). In practice, throughout the LP formulation, we consider two sets of matches \(U(k)\) and \(A(k)\) for each LP iteration \(k\). The first set \(U(k)\) begins as the universal set of all possible candidate matches for all orientations and choices of piece \(U(0) = U\) and is of size \(4n^2\). In subsequent iterations of our algorithm, its \(k^{th}\) form \(U(k)\) steadily decreases in size as our algorithm runs and rejects matches. The second set of matches \(A(k)\) is the active selection at iteration \(k\) of best candidate matches taken from \(U(k)\). For all iterations, \(A(k)\) is always of size \(4n\) and can be computed as

\[
A(k) = \{(i, j, o) \in U(k) : j = \arg\min_{j \in \{i, j, o\}(U(k))} D_{ijo}\},
\]

We take advantage of the symmetry between \(C^+(x)\) and \(C^+(y)\), and only illustrate the paper in how to solve the \(x\) coordinate subproblem.

The minimiser of \(C^+(x)\) can be written as

\[
\arg\min_{x} \sum_{(i, j, o) \in A(k)} w_{ijo}|x_i - x_j - \delta^0_x|_1.
\]

By introducing auxiliary variables \(h\), we transform the minimisation of (9) into the following linear program

\[
\arg\min_{x, h} \sum_{(i, j, o) \in A(k)} w_{ijo}h_{ijo}
\]

subject to

\[h_{ijo} \geq x_i - x_j - \delta^0_x, \quad (i, j, o) \in A(k)\]

\[h_{ijo} \geq x_i - x_j + \delta^0_x, \quad (i, j, o) \in A(k)\]

Son [2] has shown that verification of loop constraints significantly improves the precision of pairwise matches, and our work can be considered as a natural generalisation of these loop constraints that allows loops of all shapes and sizes, not necessarily rectangular, to be solved in a single LP optimization. Figure 1 shows that our method outperforms Son [2] on the difficult examples from MIT Dataset.