Efficient tracking with dynamic programming
Algorithm

\[ i_t = (x_t, y_t) \]

//Initialize first frame

\[ \text{cost}_1[i_1] = \text{Local}[i_1] \quad \forall i_1 \]

for \( t=2:T, \)

\[ \text{cost}_t[i_t] = \text{Local}[i_t] + \min_{i_{t-1} \in N(i_t)} \left( \text{cost}_{t-1}[i_{t-1}] + \text{Pair}(i_{t-1}, i_t) \right) \quad \forall i_t \]
cost_t[i_t] = Local[i_t] + \min_{i_{t-1} \in N(i_t)} \left( cost_{t-1}[i_{t-1}] + Pair(i_{t-1}, i_t) \right) \quad \forall i_t

Compute the above min (and argmin) using a “localmin” function

\[ cost_t[i_t] = Local[i_t] + m[i_t] \]

\[ [m, ptr] = \text{localmin}(cost_{t-1}, paircost) \]

paircost is a matrix of pairwise displacement costs (actually, two vectors for dx and dy costs)
Template Matching
Cost-to-frame

Template

\[ \text{cost}_t[i_t] = \text{Local}[i_t] \quad \text{for} \quad t = 1 \]
“Min-blurred” cost-to-frame

\[
\text{cost}_{t-1}[i_{t-1}]
\]

Best-cost from previous frame

\[
m[i_t]
\]

“Message” to current frame
Next frame

\[
\text{cost}_t[i_t] = \text{Local}[i_t] + m[i_t]
\]

= new cost-to-frame

\[
\text{cost}_t[i_t]
\]