16-720 Computer Vision: Homework 3

Yanjia Duan

October 2020

Q1.1

For $\frac{\partial \mathcal{W}(\mathbf{x};\mathbf{p})}{\partial \mathbf{p}^T}$,

$$\mathcal{W}(\mathbf{x};\mathbf{p}) = \begin{pmatrix} x + p_x \\ y + p_y \end{pmatrix} = \begin{pmatrix} 1 & 0 & p_x \\ 0 & 1 & p_y \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

$$\frac{\partial \mathcal{W}(\mathbf{x};\mathbf{p})}{\partial \mathbf{p}^T} = \begin{pmatrix} \frac{\partial \mathcal{W}_x}{\partial p_x} & \frac{\partial \mathcal{W}_x}{\partial p_y} \\ \frac{\partial \mathcal{W}_y}{\partial p_x} & \frac{\partial \mathcal{W}_y}{\partial p_y} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

For $\mathbf{A}$ and $\mathbf{b}$,

$$\begin{aligned}
&\text{argmin}_{\Delta\mathbf{p}} \sum_{\mathbf{x}\in\mathbb{N}} \|\mathcal{I}_{t+1}(\mathbf{x}' + \Delta\mathbf{p}) - \mathcal{I}_t(\mathbf{x})\|_2^2 \\
=&\text{argmin}_{\Delta\mathbf{p}} \sum_{\mathbf{x}\in\mathbb{N}} \left\| \mathcal{I}_{t+1}(\mathbf{x}') + \frac{\partial \mathcal{I}_{t+1}(\mathbf{x}')}{\partial \mathbf{x}'^T} \frac{\partial \mathcal{W}(\mathbf{x};\mathbf{p})}{\partial \mathbf{p}^T} \Delta\mathbf{p} - \mathcal{I}_t(\mathbf{x}) \right\|_2^2 \\
=&\text{argmin}_{\Delta\mathbf{p}} \sum_{\mathbf{x}\in\mathbb{N}} \left\| \frac{\partial \mathcal{I}_{t+1}(\mathbf{x}')}{\partial \mathbf{x}'^T} \frac{\partial \mathcal{W}(\mathbf{x};\mathbf{p})}{\partial \mathbf{p}^T} \Delta\mathbf{p} - (\mathcal{I}_t(\mathbf{x}) - \mathcal{I}_{t+1}(\mathbf{x}')) \right\|_2^2 \\
=&\text{argmin}_{\Delta\mathbf{p}} \|\mathbf{A}\Delta\mathbf{p} - \mathbf{b}\|_2^2
\end{aligned}$$

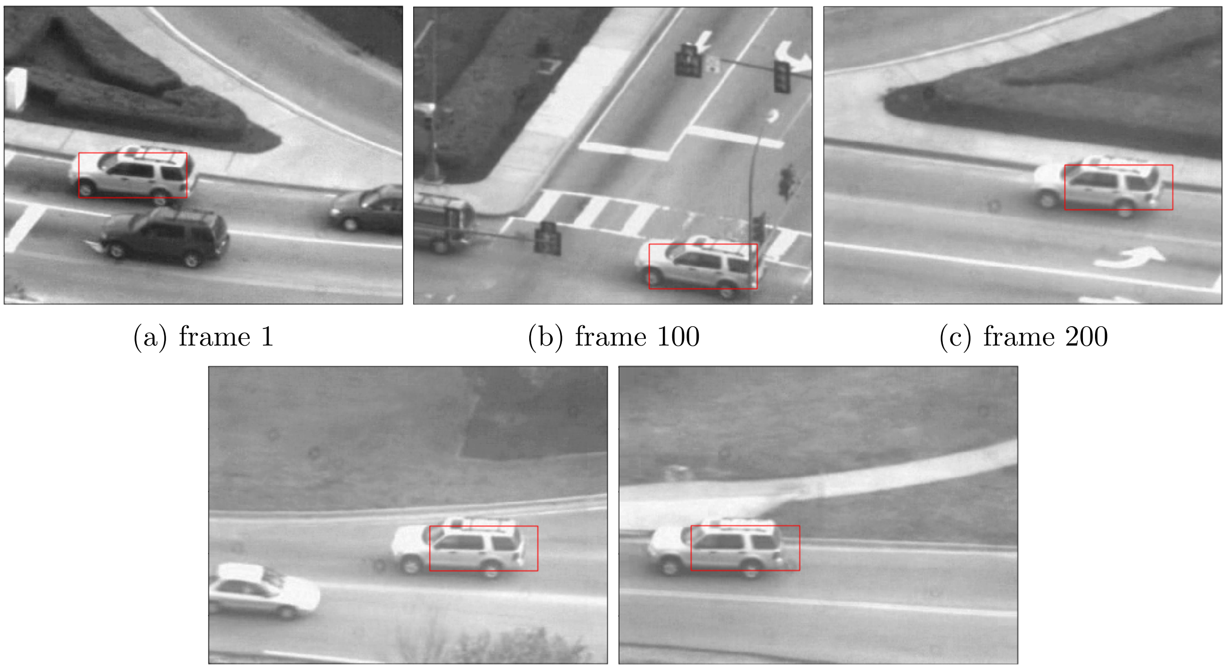
$$\therefore \mathbf{A} = \frac{\partial \mathcal{I}_{t+1}(\mathbf{x}')}{\partial \mathbf{x}'^T} \frac{\partial \mathcal{W}(\mathbf{x};\mathbf{p})}{\partial \mathbf{p}^T} = \frac{\partial \mathcal{I}_{t+1}(\mathbf{x}')}{\partial \mathbf{x}'^T} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \frac{\partial \mathcal{I}_{t+1}(\mathbf{x}')}{\partial \mathbf{x}'^T}, \text{where } \mathbf{x}' = \mathbf{x} + \mathbf{p}$$
$$\mathbf{b} = \mathcal{I}_t(\mathbf{x}) - \mathcal{I}_{t+1}(\mathbf{x}') = \mathcal{I}_t(\mathbf{x}) - \mathcal{I}_{t+1}(\mathbf{x} + \mathbf{p})$$

In order to find a unique solution to $\Delta\mathbf{p}$,

$$\frac{\partial(\|\mathbf{A}\Delta\mathbf{p} - \mathbf{b}\|_2^2)}{\partial \Delta\mathbf{p}} = 2\mathbf{A}^T(\mathbf{A}\Delta\mathbf{p} - \mathbf{b}) = 0$$
$$\mathbf{A}^T\mathbf{A}\Delta\mathbf{p} = \mathbf{A}^T\mathbf{b}$$
$$\Delta\mathbf{p} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}$$

$\mathbf{A}^T\mathbf{A}$ should be invertible, so $\det(\mathbf{A}^T\mathbf{A}) \neq 0$. Therefore, $\mathbf{A}^T\mathbf{A}$ needs to be full-rank.

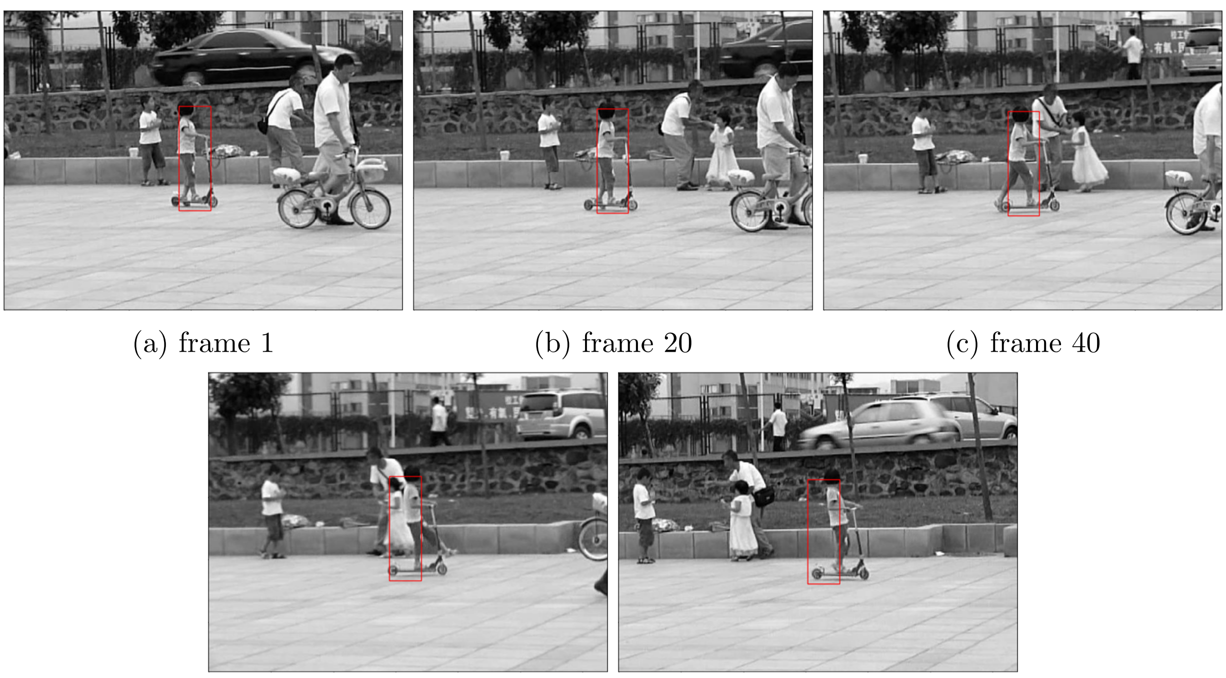
Q1.3



(a) frame 1 (b) frame 100 (c) frame 200

(d) frame 300 (e) frame 400

Figure 1: Car tracking performance at frames 1, 100, 200, 300 and 400 using Lucas-Kanade with one single template.

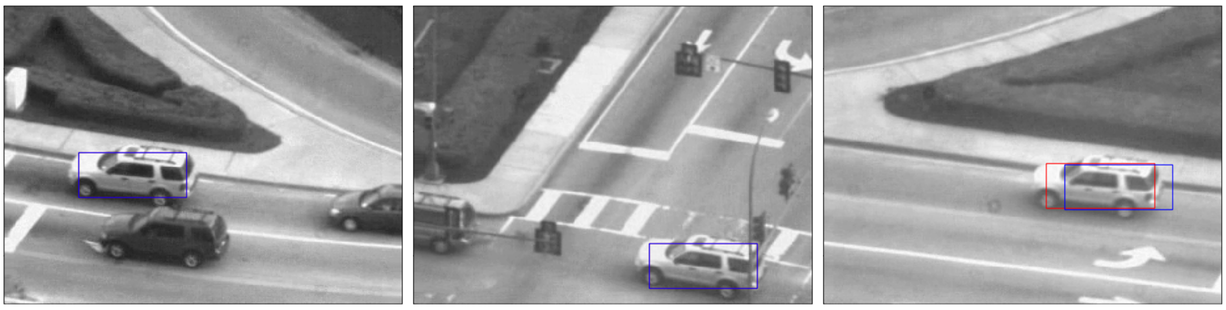


(a) frame 1 (b) frame 20 (c) frame 40

(d) frame 60 (e) frame 80

Figure 2: Girl tracking performance at frames 1, 20, 40, 60 and 80 using Lucas-Kanade with one single template.

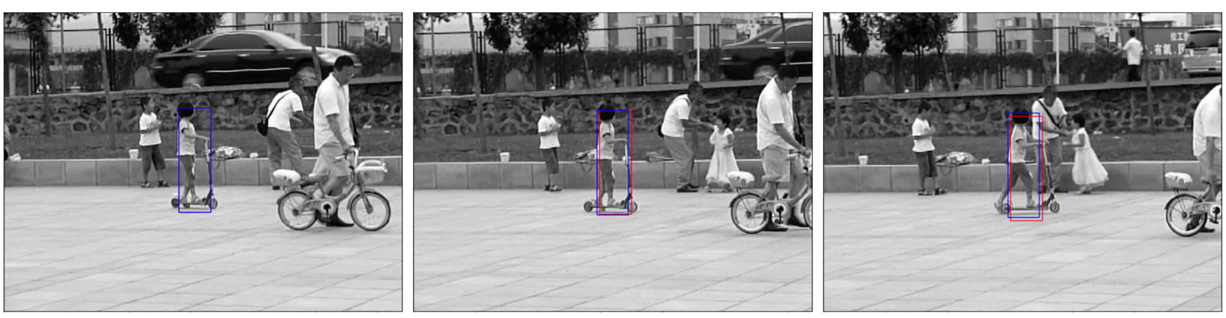
Q1.4



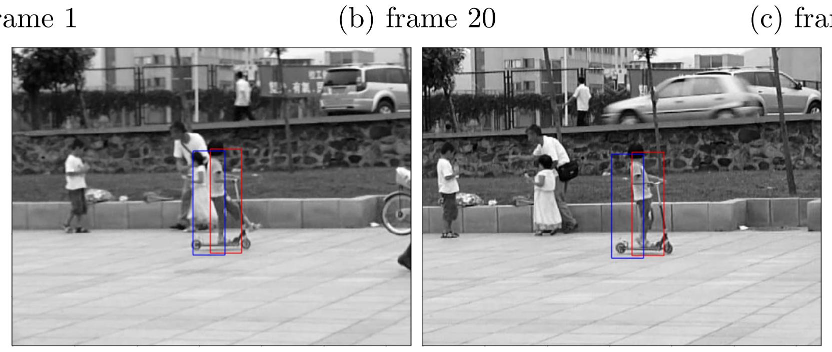
(a) frame 1 (b) frame 100 (c) frame 200

(d) frame 300 (e) frame 400

Figure 3: Car tracking performance at frames 1, 100, 200, 300 and 400 using Lucas-Kanade with template correction (red) and one single template (blue).



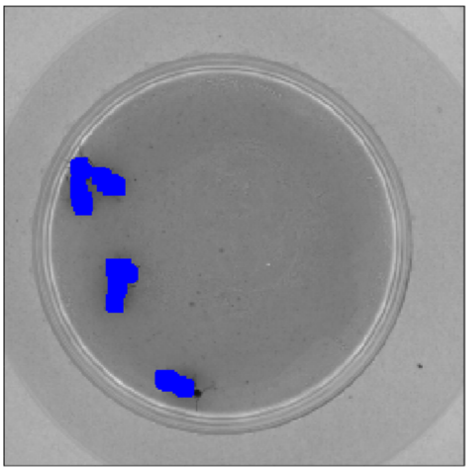
(a) frame 1 (b) frame 20 (c) frame 40



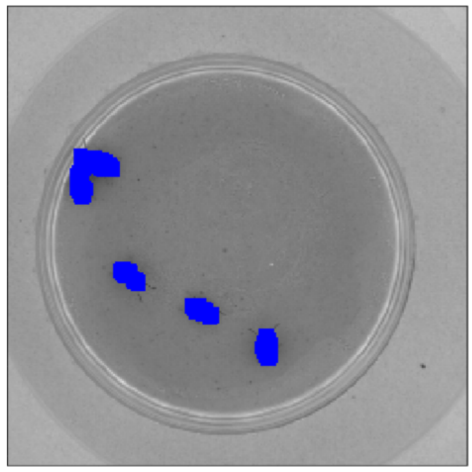
(d) frame 60 (e) frame 80

Figure 4: Girl tracking performance at frames 1, 20, 40, 60 and 80 using Lucas-Kanade with template correction (red) and one single template (blue).

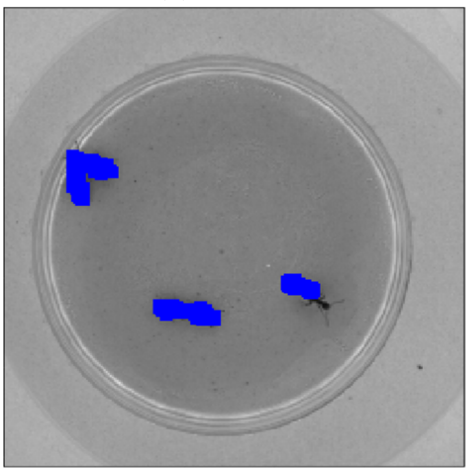
Q2.3



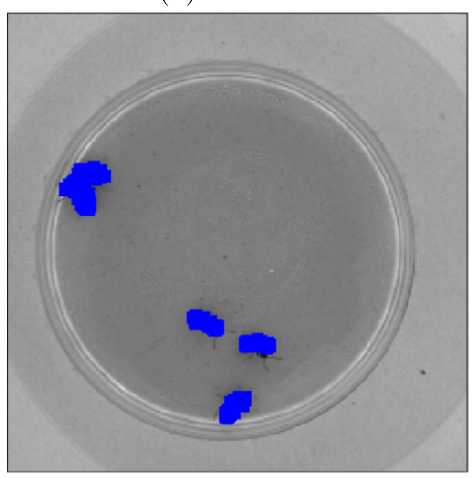
(a) frame 30



(b) frame 60

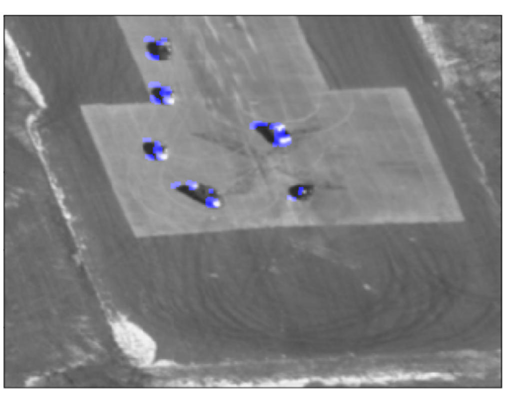


(c) frame 90

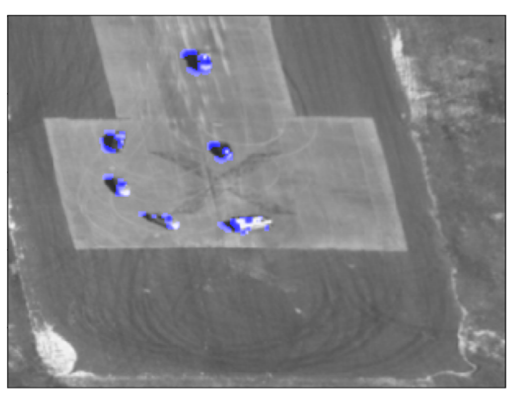


(d) frame 120

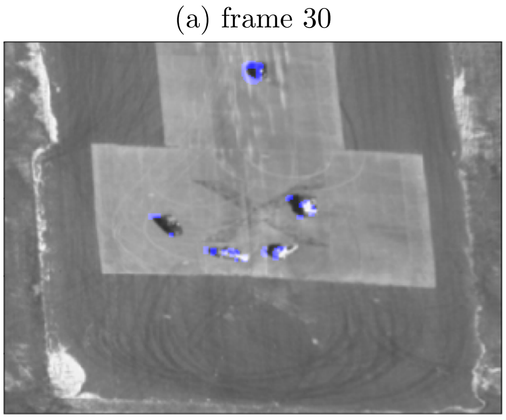
Figure 5: Ant tracking performance at frames 30, 60, 90 and 120 using Lucas-Kanade with Motion Detection.



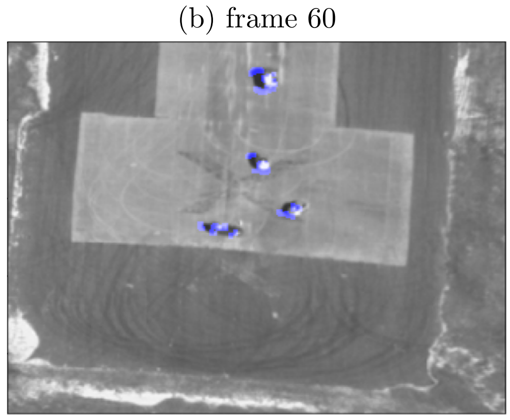
(a) frame 30



(b) frame 60



(c) frame 90



(d) frame 120

Figure 6: Aerial tracking performance at frames 30, 60, 90 and 120 using Lucas-Kanade with Motion Detection.

Q3.1

Inverse compositional approach warps $\mathcal{I}_t$ (template) with $\Delta\mathbf{p}$ and $\mathcal{I}_{t+1}$ with $\mathbf{p}$. After applying Taylor series expansion, $\Delta\mathbf{p}$ is only on the template side. Since the template stays still and the warp is evaluated at $\mathbf{0}$, $\mathbf{A}' = \frac{\partial\mathcal{I}_t(\mathbf{x})}{\partial\mathbf{x}^T}\frac{\partial\mathcal{W}(\mathbf{x};\mathbf{0})}{\partial\mathbf{p}^T}$ is constant and can be calculated outside of iteration. The only thing that depends on $\mathbf{p}$ is $\mathbf{b}' = \mathcal{I}_{t+1}(\mathcal{W}(\mathbf{x};\mathbf{p})) - \mathcal{I}_t(\mathbf{x})$, which needs to be done inside of iteration.

Classical approach (forward additive) warps $\mathcal{I}_{t+1}$ with $\mathbf{p}+\Delta\mathbf{p}$. Since $\Delta\mathbf{p}$ is on the changing $\mathcal{I}_{t+1}$ side, $\mathbf{A}'$ includes the Jacobian matrix which includes $\mathbf{p}$. Therefore, the calculation of $\mathbf{A}'$ needs to be done inside of each iteration.

The inverse compositional approach pre-calculates $\mathbf{A}'$, therefore it's more computationally efficient than the classical approach.