Q1

- The expression $\frac{\partial W(x;p)}{\partial p^T}$ represents the jacobian required to calculate the taylor series solution for gradient descent, and its value is calculated as:

$$\frac{\partial W(x;p)}{\partial p^T} = \begin{bmatrix} \frac{\partial W_x}{\partial p_1} & \cdots\cdots & \frac{\partial W_x}{\partial p_n} \\ \frac{\partial W_y}{\partial p_1} & \cdots\cdots & \frac{\partial W_y}{\partial p_n} \end{bmatrix}$$

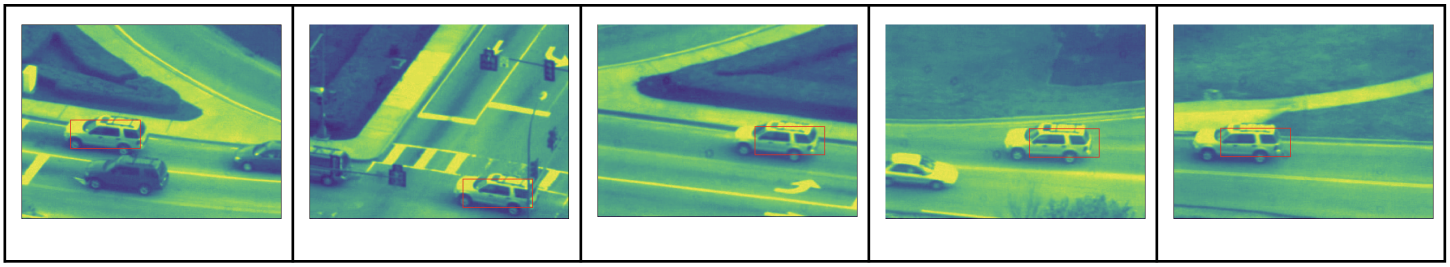
- The values of A and B are:

$$A = \nabla I \frac{\partial W}{\partial P}$$

$$B = (T(x) - I(W(x,p)))$$

- For there to exist a unique solution for ΔP, the matrix A^TA needs to be non-singular, since it needs to be invertible.

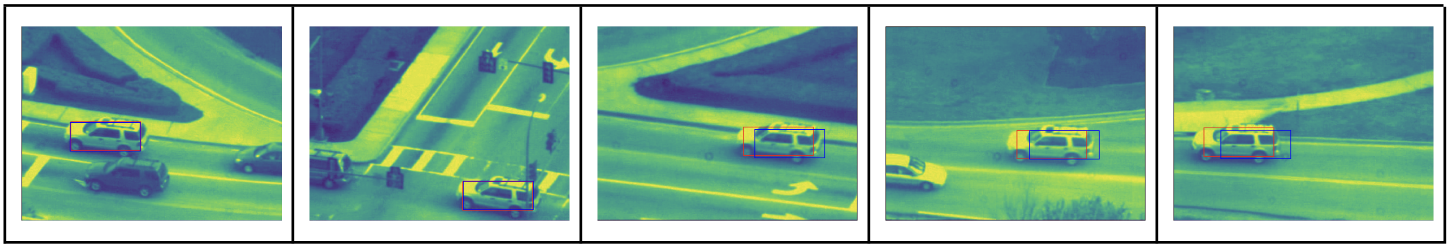
Q1.3



Visualization of the car tracking output at frames 1, 100, 200, 300, 400

Visualization of the girl tracking output at frames 1, 20, 40, 60, 80

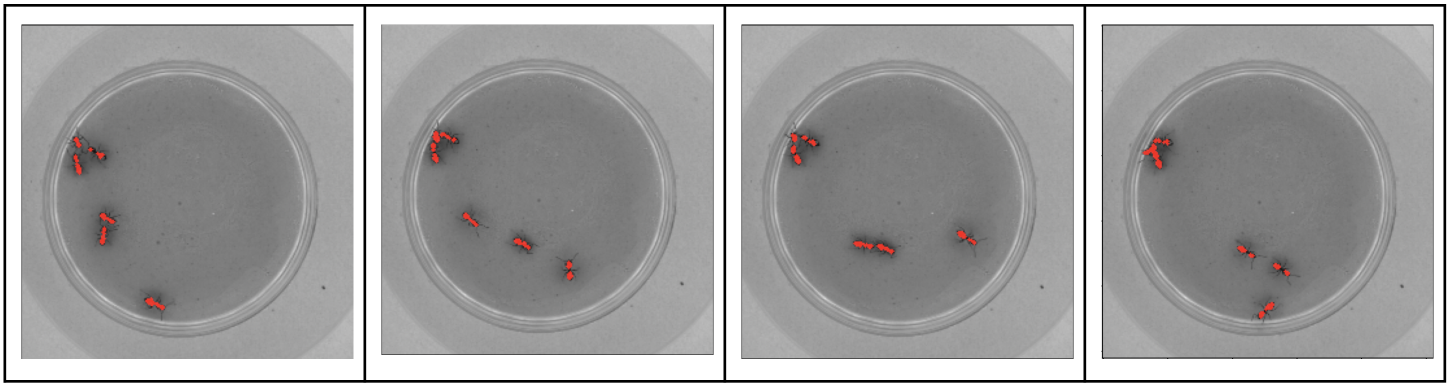
Q1.4



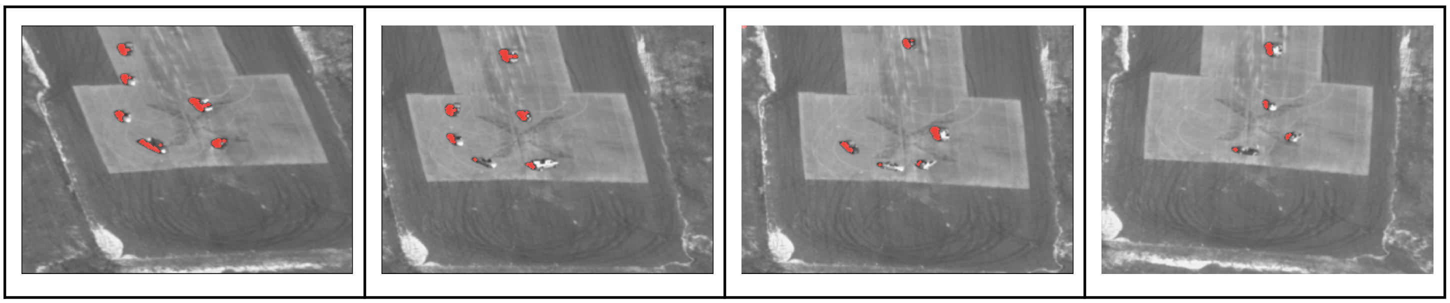
Visualization of the car tracking output with template correction at frames 1, 100, 200, 300, 400

Visualization of the girl tracking output with template correction at frames 1, 20, 40, 60, 80

Q2.3

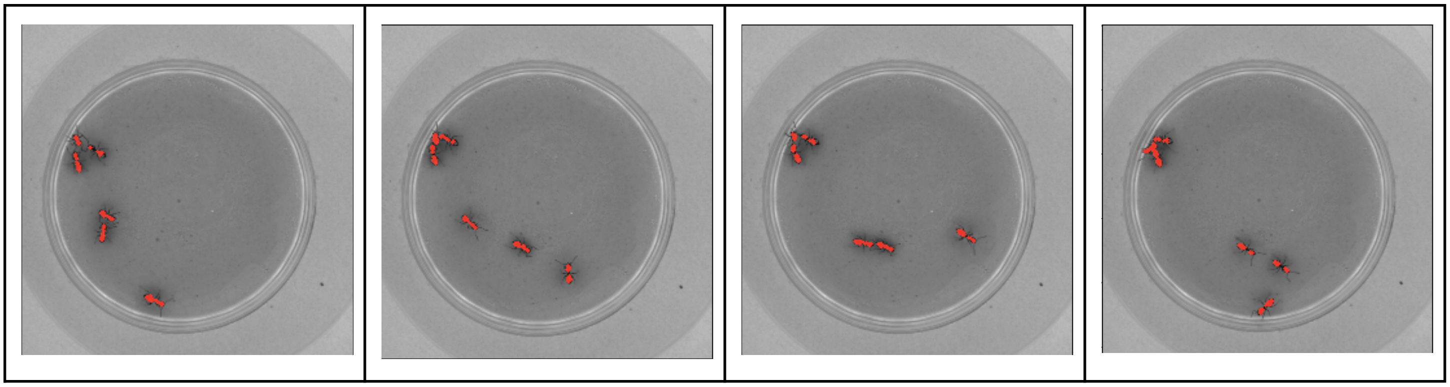


Visualization of the ant tracking output at frames 30, 60, 90, 120

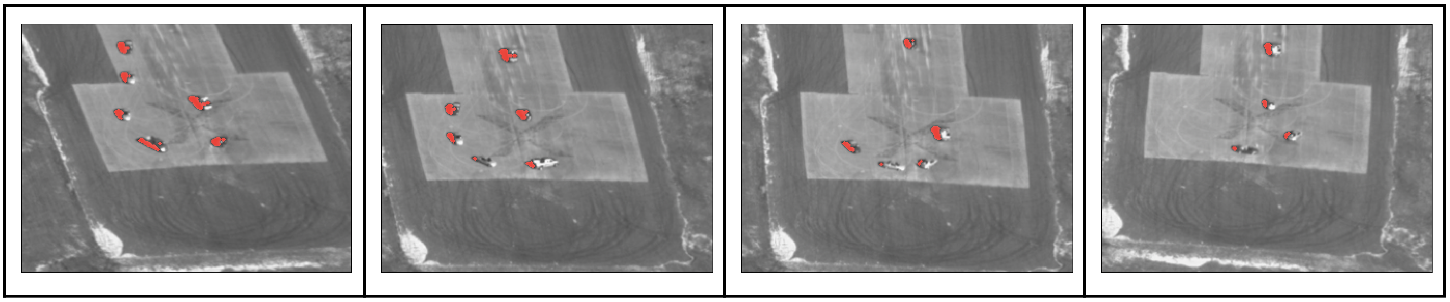


Visualization of the aerial tracking output at frames 30, 60, 90, 120

Q3.1



Visualization of the ant tracking output using Inverse Composition Affine at frames 30, 60, 90, 120



Visualization of the aerial tracking output using Inverse Composition Affine at frames 30, 60, 90, 120

	Ant Sequence		Aerial Sequence	
Frame	Time for one frame using Lucas Kanade Affine (s)	Time for one frame using Inverse Composition Affine (s)	Time for one frame using Lucas Kanade Affine (s)	Time for one frame using Inverse Composition Affine (s)
30	12.68	0.59	2.11	0.40
60	18.09	0.60	16.9	0.55
90	44.35	0.65	2.11	0.17
120	4.62	0.69	0.75	0.37

Table comparing the time taken for one frame using both affine methods on the ant and aerial sequences

As we can see in the table, the time taken for each frame is significantly lesser while using the Inverse Composition Affine, as compared to the Lucas Kanade Affine. This is happening since we are pre-computing the gradient, jacobian, hessian, and the steepest descent of the image. These operations are very computationally heavy and not evaluating them for every iteration makes a significant difference to the time taken.