

CS 83: COMPUTER VISION

VIDEO TRACKING

Amittai Siavava

03/03/2024

Abstract

This project implements Lucas Kanade, Lucas Kanade Affine, and Inverse Compositional algorithms for tracking optical flow, helping keep track of objects in a video.

Credit Statement

I discussed ideas with **Ivy (Aiwei) Zhang** and **Angelic McPherson**. However, the code and writeup are entirely my own, with reference to class notes.

CONTENTS

1. Theory Questions	2
3. Video Tracking Results: Lucas Kanade	4
4. Video Tracking Results: Lucas Kanade Affine	7
5. Video Tracking Results: Inverse Compositional Affine	11

1. THEORY QUESTIONS

Problem 1.

Assuming the affine warp model $\mathbf{W} = \begin{bmatrix} 1 + p_1 & p_3 & p_5 \\ p_2 & 1 + p_4 & p_6 \\ 0 & 0 & 1 \end{bmatrix}$, derive the expression for the Jacobian matrix \mathbf{J} in terms of the warp parameters $\mathbf{p} = [p_1, p_2, p_3, p_4, p_5, p_6]^\top$.

The Jacobian matrix \mathbf{J} is the matrix of partial derivatives of the warp \mathbf{W} with respect to the warp parameters \mathbf{p} . We have

$$\mathbf{W} = \begin{bmatrix} 1 + p_1 & p_3 & p_5 \\ p_2 & 1 + p_4 & p_6 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{J} = \begin{bmatrix} \frac{\partial \mathbf{W}}{\partial p_1} & \frac{\partial \mathbf{W}}{\partial p_2} & \frac{\partial \mathbf{W}}{\partial p_3} & \frac{\partial \mathbf{W}}{\partial p_4} & \frac{\partial \mathbf{W}}{\partial p_5} & \frac{\partial \mathbf{W}}{\partial p_6} \end{bmatrix}$$

We can compute the partial derivatives of \mathbf{W} with respect to the warp parameters \mathbf{p} as follows:

$$\frac{\partial \mathbf{W}}{\partial p_1} = \begin{bmatrix} x \\ 0 \end{bmatrix}, \quad \frac{\partial \mathbf{W}}{\partial p_2} = \begin{bmatrix} 0 \\ x \end{bmatrix}, \quad \frac{\partial \mathbf{W}}{\partial p_3} = \begin{bmatrix} y \\ 0 \end{bmatrix},$$

$$\frac{\partial \mathbf{W}}{\partial p_4} = \begin{bmatrix} 0 \\ y \end{bmatrix}, \quad \frac{\partial \mathbf{W}}{\partial p_5} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \frac{\partial \mathbf{W}}{\partial p_6} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Therefore, the Jacobian matrix \mathbf{J} is:

$$\mathbf{J} = \begin{bmatrix} x & 0 & y & 0 & 1 & 0 \\ 0 & x & 0 & y & 0 & 1 \end{bmatrix}$$

Note: Since the last row of \mathbf{W} is always $[0, 0, 1]$ and therefore leaves the x -coordinate (in homogeneous coordinates) unchanged, we do not need to include the partial derivatives of the last row of \mathbf{W} with respect to the warp parameters \mathbf{p} since the last row would be all zeros.

Problem 2.

Find the computational complexity (Big O notation) for the initialization step (pre-computing \mathbf{J} and \mathbf{H}^{-1}) and for each runtime iteration (Equation 13) of the Matthews-Baker method:

$$\Delta \mathbf{p}^* = \mathbf{H}^{-1} \mathbf{J}^\top [\mathbf{I}(\mathbf{W}(\mathbf{x}; \mathbf{p})) - \mathbf{T}]$$

Express your answers in terms of n , m , and p , where

- (i) n is the number of pixels in the template \mathbf{T} ,
- (ii) m is the number of pixels in an input image \mathbf{I} , and
- (iii) p is the number of parameters used to describe the warp \mathbf{W} .

How does this compare to the runtime of the regular Lucas-Kanade method?

1. Initialization Step:

- (a) **Pre-computing \mathbf{J} :** The computational complexity of pre-computing \mathbf{J} is $O(np)$ since we need to compute the partial derivatives of the warp \mathbf{W} with respect to each of the warp parameters \mathbf{p} , for each of the n pixels in the image.
- (b) **Pre-computing \mathbf{H}^{-1} :** The computational complexity of pre-computing \mathbf{H}^{-1} is $O(p^3)$ since we need to compute the inverse of the Hessian matrix \mathbf{H} .

3. VIDEO TRACKING RESULTS: LUCAS KANADE

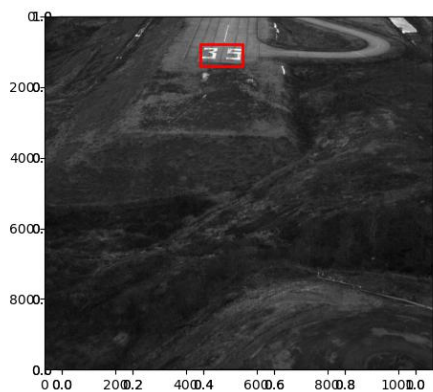


FIGURE 1. Frame 1



FIGURE 2. Frame 10



FIGURE 3. Frame 20



FIGURE 4. Frame 1

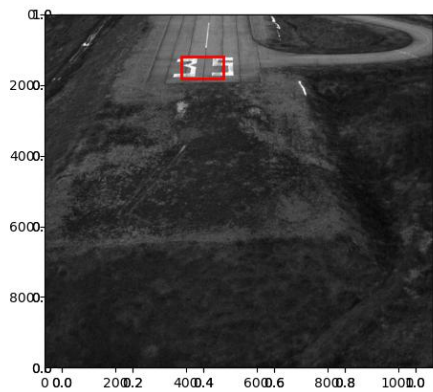


FIGURE 5. Frame 10

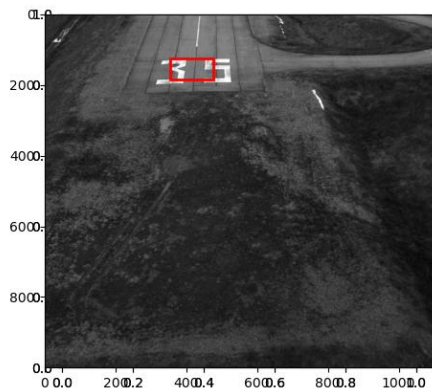


FIGURE 6. Frame 20

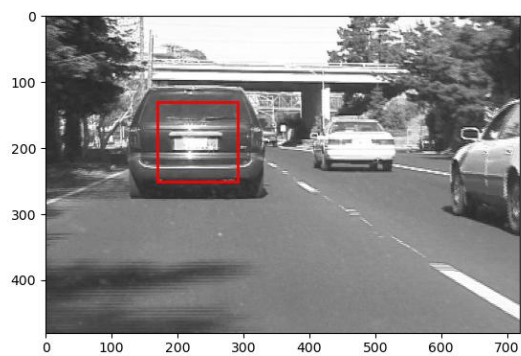


FIGURE 7. Frame 1

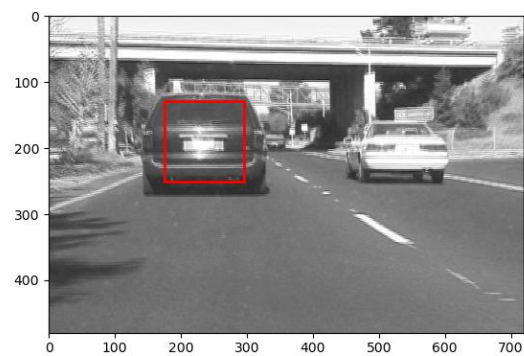


FIGURE 8. Frame 50



FIGURE 9. Frame 100

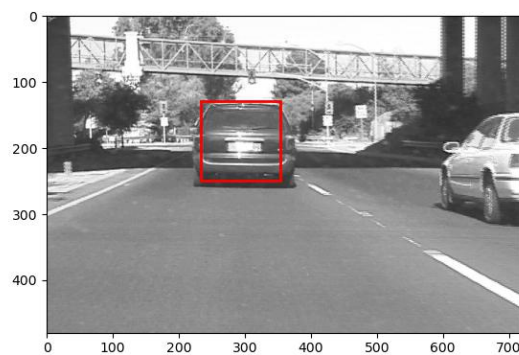


FIGURE 10. Frame 150

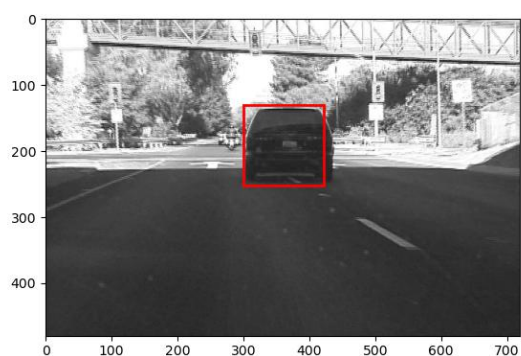


FIGURE 11. Frame 200



FIGURE 12. Frame 250



FIGURE 13. Frame 1



FIGURE 14. Frame 50



FIGURE 15. Frame 100



FIGURE 16. Frame 150



FIGURE 17. Frame 200



FIGURE 18. Frame 250

4. VIDEO TRACKING RESULTS: LUCAS KANADE AFFINE

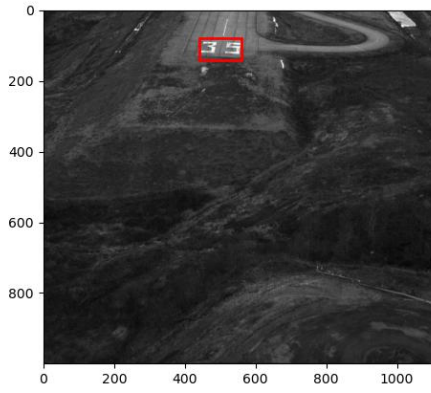


FIGURE 19. Frame 1

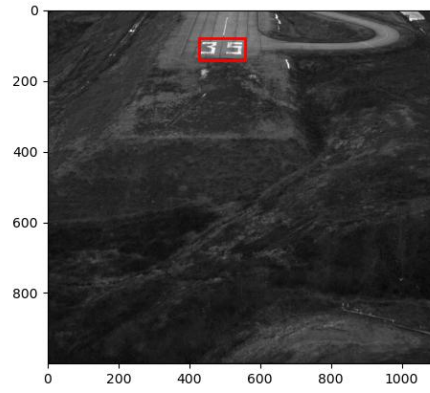


FIGURE 20. Frame 10

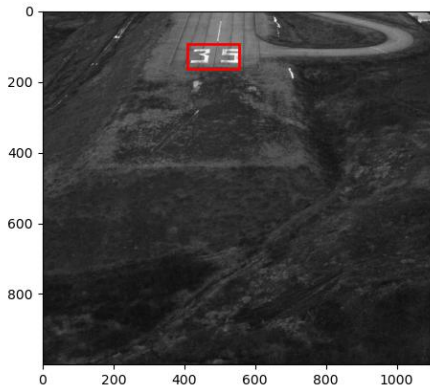


FIGURE 21. Frame 20

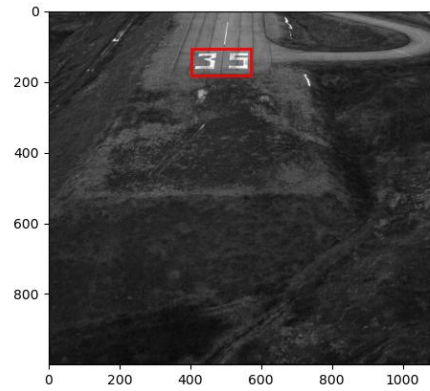


FIGURE 22. Frame 1

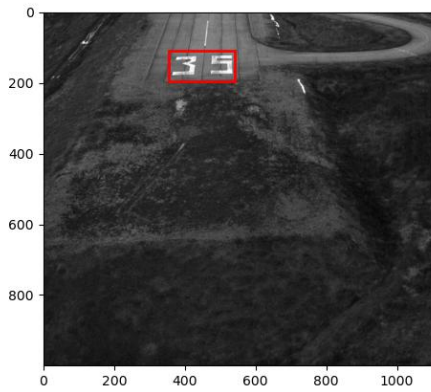


FIGURE 23. Frame 10

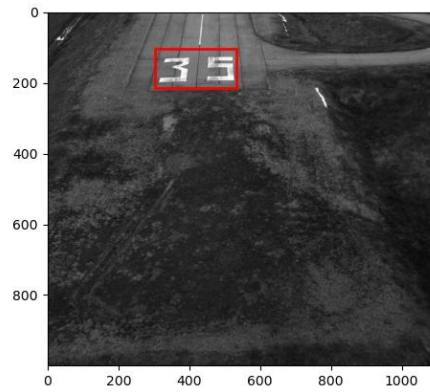


FIGURE 24. Frame 20

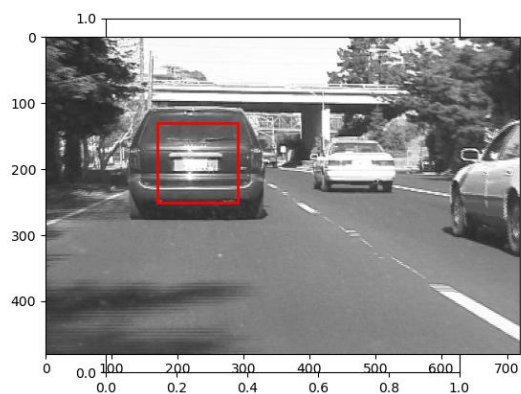


FIGURE 25. Frame 1

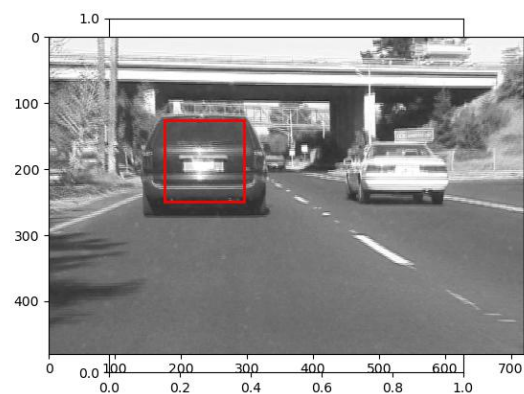


FIGURE 26. Frame 50

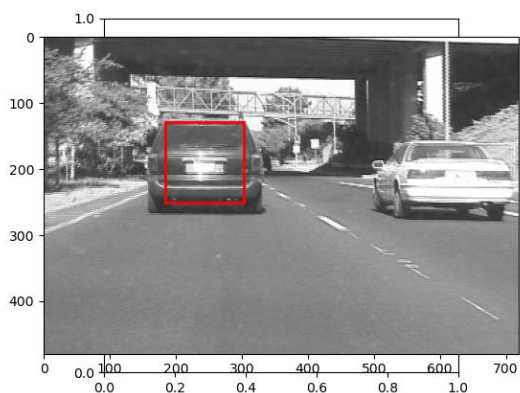


FIGURE 27. Frame 100

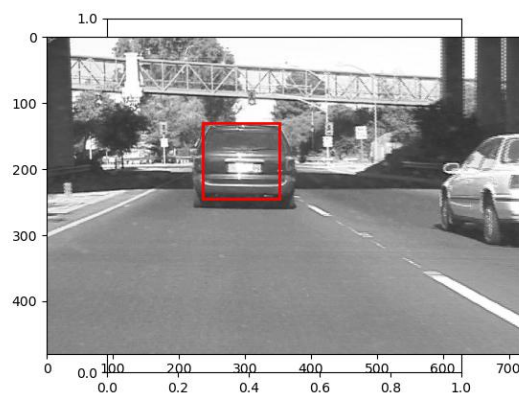


FIGURE 28. Frame 150

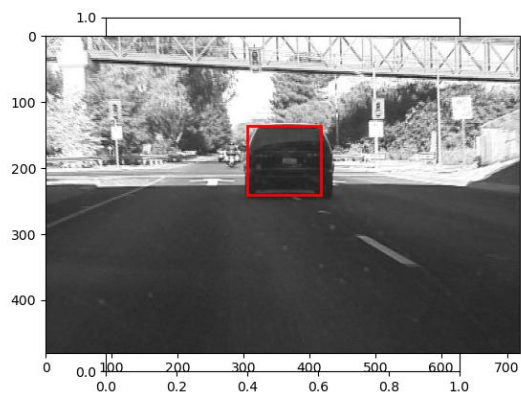


FIGURE 29. Frame 200

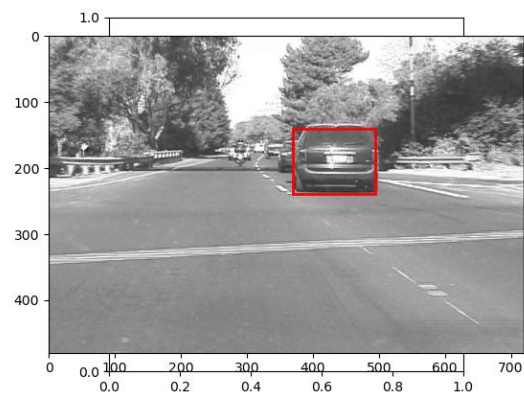


FIGURE 30. Frame 250

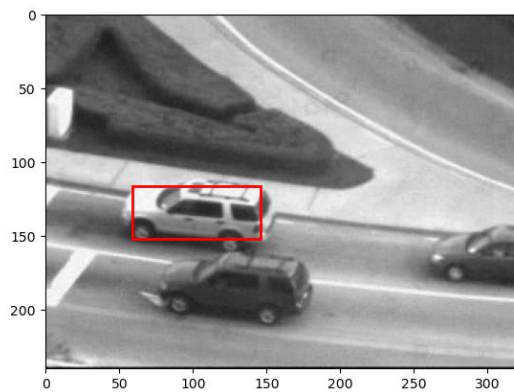


FIGURE 31. Frame 1



FIGURE 32. Frame 50

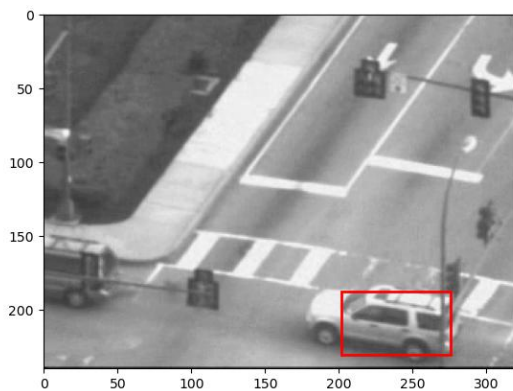


FIGURE 33. Frame 100



FIGURE 34. Frame 150



FIGURE 35. Frame 200



FIGURE 36. Frame 250

I noticed that the affine transformations (Lucas-Kanade affine and Inverse-Compositional Affine) were thrown off in the third video and seemed to stick to one of the street lights as the car moved. Here's the exact place where that happens in Lucas-Kanade:



FIGURE 37. Frame 125



FIGURE 38. Frame 135



FIGURE 39. Frame 145



FIGURE 40. Frame 155

5. VIDEO TRACKING RESULTS: INVERSE COMPOSITIONAL AFFINE

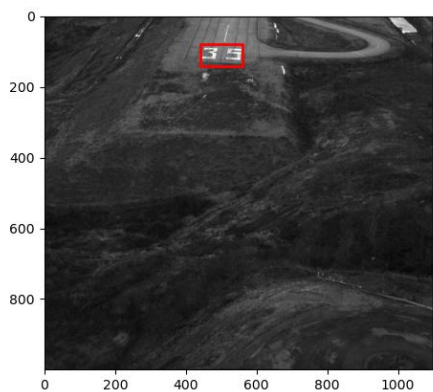


FIGURE 41. Frame 1

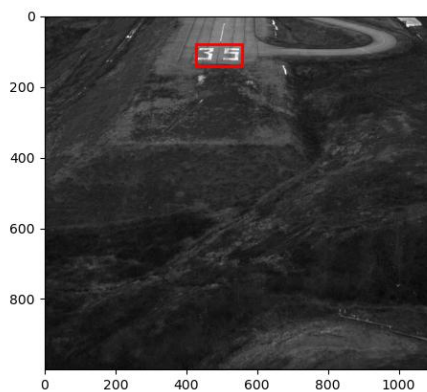


FIGURE 42. Frame 10

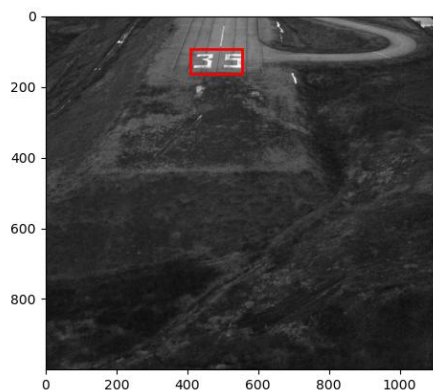


FIGURE 43. Frame 20

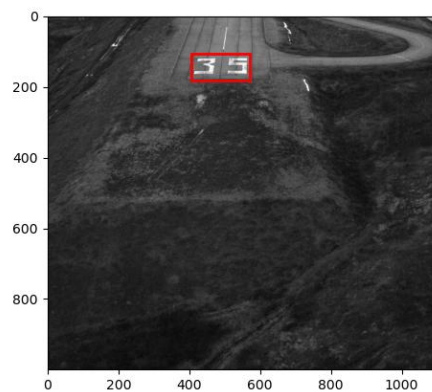


FIGURE 44. Frame 1

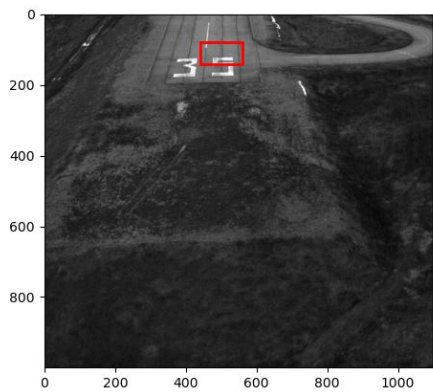


FIGURE 45. Frame 10

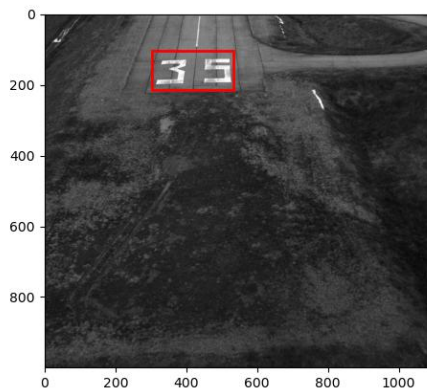


FIGURE 46. Frame 20

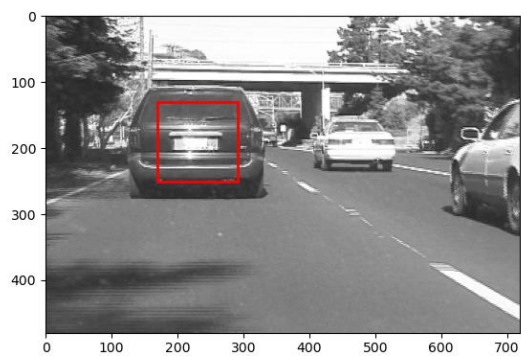


FIGURE 47. Frame 1

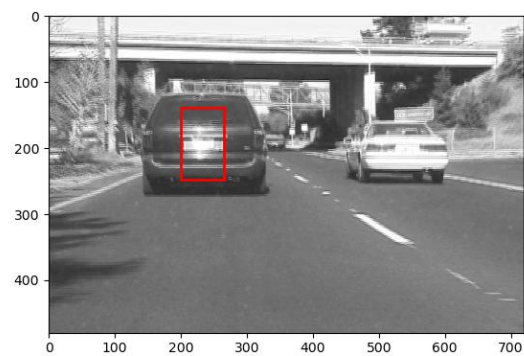


FIGURE 48. Frame 50

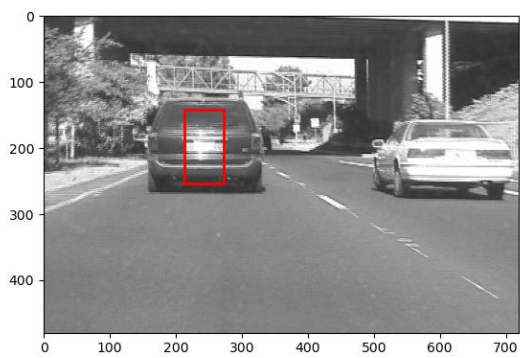


FIGURE 49. Frame 100

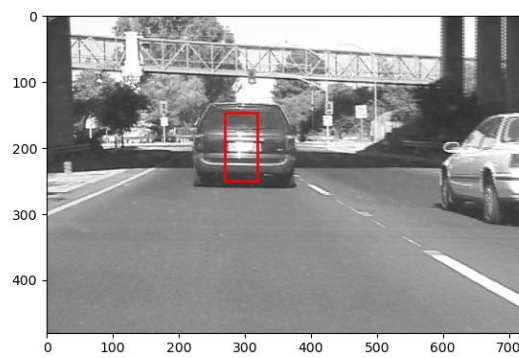


FIGURE 50. Frame 150

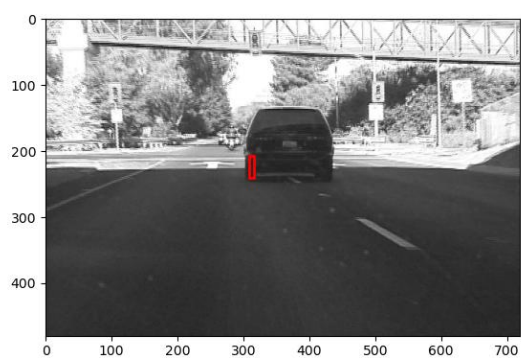


FIGURE 51. Frame 200

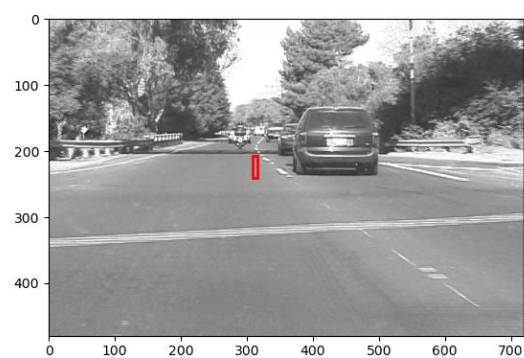


FIGURE 52. Frame 250

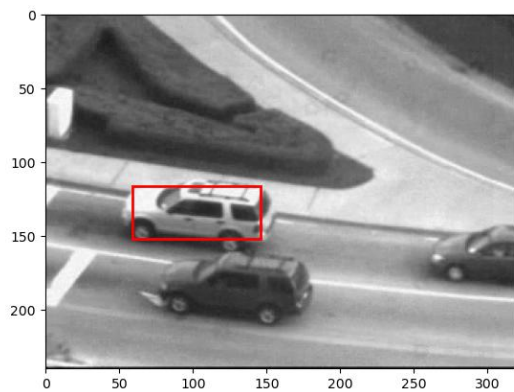


FIGURE 53. Frame 1



FIGURE 54. Frame 50

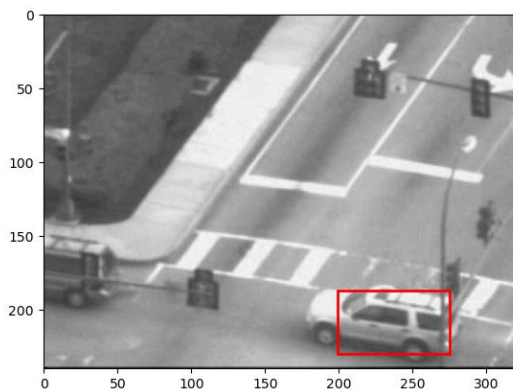


FIGURE 55. Frame 100

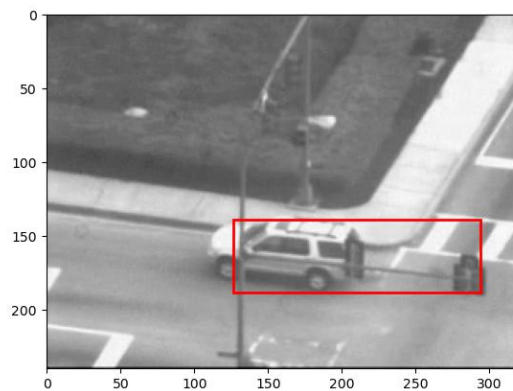


FIGURE 56. Frame 150



FIGURE 57. Frame 200



FIGURE 58. Frame 250

I noticed that the affine transformations (Lucas-Kanade affine and Inverse-Compositional Affine) were thrown off in the third video and seemed to stick to one of the street lights as the car moved. Here's the exact place where that happens in Inverse-Compositional Affine:

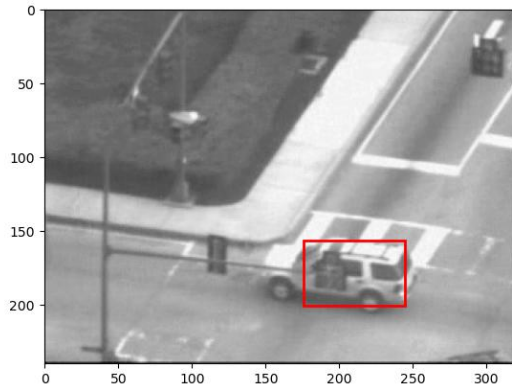


FIGURE 59. Frame 125



FIGURE 60. Frame 135



FIGURE 61. Frame 145



FIGURE 62. Frame 155