16-820: Advanced Computer Vision
HW-2

Akanksha Singal (asingal2)

September 26, 2025

Section 3 Affine Motion Subtraction Ant

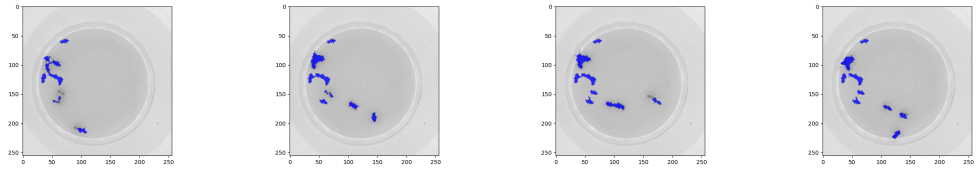

Figure 1: ITERATION FOR EROSION AND DILATION = 1, num iters = 1000, threshold = 0.01, tolerance = 0.2

Section 3 Affine Motion Subtraction Aerial

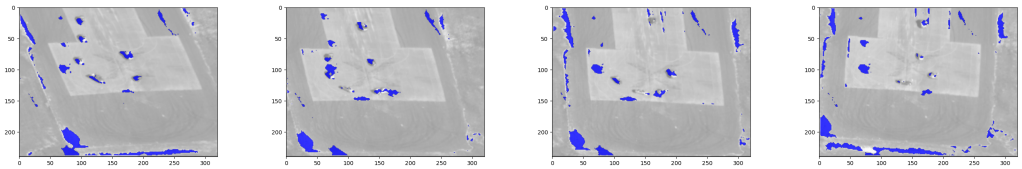

Figure 2: ITERATION FOR EROSION AND DILATION = 2, num iters = 1000, threshold = 0.01, tolerance = 0.3

Section 4 Inverse Affine Ant

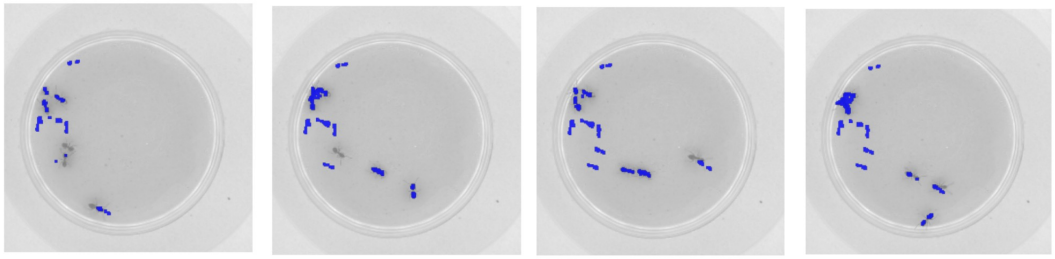

Figure 3: ITERATION FOR EROSION AND DILATION = 1, num iters = 1000, threshold = 0.02, tolerance = 0.2

Section 4 Inverse Affine Aerial

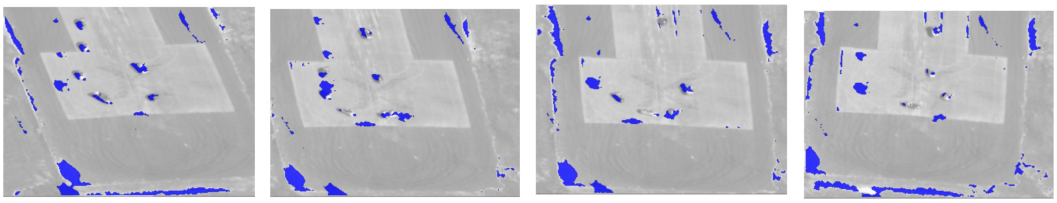

Figure 4: ITERATION FOR EROSION AND DILATION = 2, num iters = 1000, threshold = 1.0, tolerance = 0.3

Section 4 Theory

Q4.2.1 Compare the runtime of the algorithm using inverse composition (as described in this section) with its runtime without inverse composition (as detailed in the previous section) in the context of the ant and aerial sequences:

===== your answer here! =====
Inverse composition Results:
Ant Sequence takes 8.863341 seconds
Aerial Sequence takes 43.708784 seconds

Without Inverse Composition:
Ant Sequence takes 13.120779 seconds
Aerial Sequence takes 1813.514410 seconds

===== end of your answer ====

Q4.2.2 In your own words, please describe briefly why the inverse compositional approach is more computationally efficient than the classical approach:

===== your answer here! =====

We see that the inverse compositional approach is more computationally efficient than the classical approach. This is because in classical approach, we were computing the gradients, A matrix and hessian in each iteration. In inverse compositional approach, the gradients, A matrix and the hessian are computed once on the fixed template image and reused for all iterations making it faster. Only the inverse of hessian needs to be computed in each iteration. The difference is not very huge in my implementation as I am using opencv functions in classical approach to run on colab. Classical approach from scratch was taking up approximately 2hrs for the aerial sequence hence had to migrate to opencv and numpy functions that use cpp wrapper and are relatively faster. Also in case of ant sequence the camera movement is not as much as in aerial sequence. Hence the time difference is less in ant sequence as compared to aerial sequence.

===== end of your answer ====

Figure 5: 4.2.1 and 4.2.2

Section 5.3 b Hammer Handle sequence

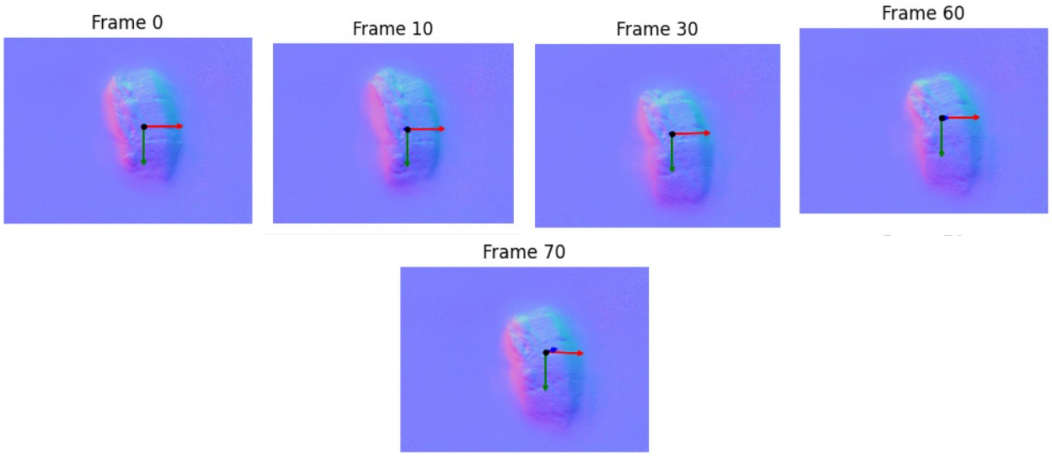

Figure 6: Normal flow part b