CMSC426

Project 3 – ROTOBRUSH

- Rachith Prakash

In this project, object of interest is tracked throughout the frames/video. This project involves several stages.

- The main idea behind this project is segmentation by identifying local features through windows of certain dimensions placed along the boundary of the object and tracking these windows movement between frames using affine transformation and probability masks.
- The features used to describe each local window are shape and color model. Color model for each window is a set of 3 GMM's for each of foreground and background data. For each window, the probability mask is calculated based on the above calculated GMM models.
- Using these probability values, color confidence is calculated for each window. Based on these color confidences, shape confidence is calculated. This concept of shape confidence is important aspect as it accounts for deformations.
- Once probability mask for each window is calculated, they are summed up (weighted) to get the probability mask for the entire image/frame.
- Iterative refinement: The process of recalculating the color and shape model is called iterative refinement. We have implemented this to improve the area of coverage of foreground mask. This helps in making a clear boundary around the object.
    - An example of iterative refinement. For the same frame, color model and shape model are recalculated. This will improve the segmentation accuracy. More of foreground would be covered using this method. But it is computationally expensive!

Fig: Need to Zoom in and see!

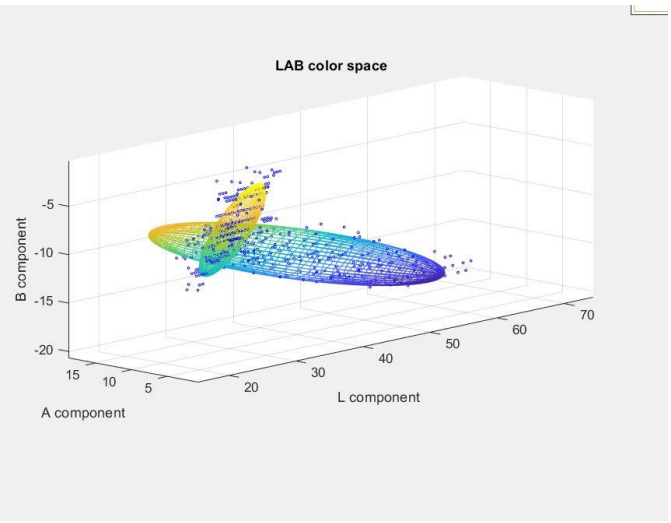
- Chose LAB color space and 3 Gaussians for fitting the color model, for all datasets because the ellipsoids were best fitting in case of LAB color space.

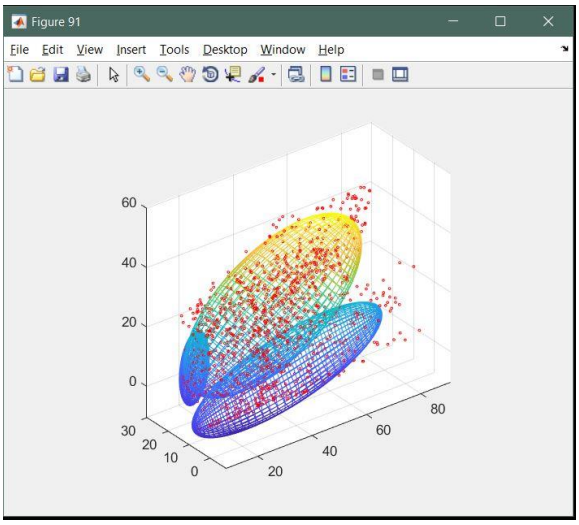
First dataset: Turtle

I believe this was the simplest case of all the datasets given. Why? Because, the background and foreground were easily separable. The background was just plain water (blue) and foreground was the turtle (green and few shades). The result was eventually very good.

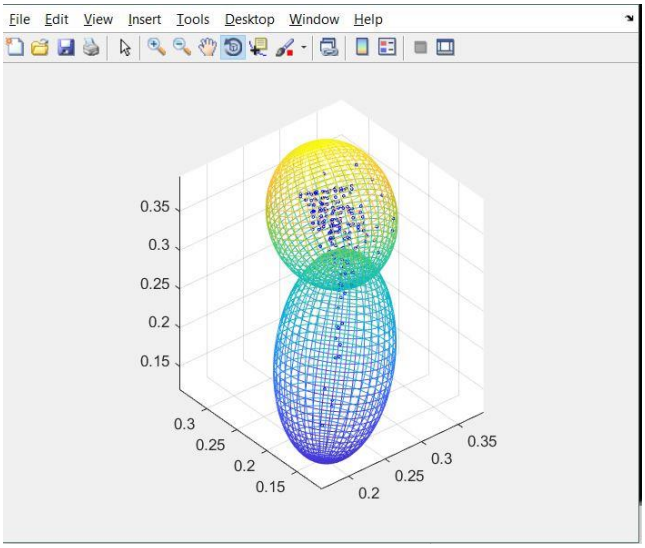
Window Width: 80

Number of windows: 30





The above pictures depict the ellipsoid encapsulation of foreground and background data in LAB color space for a local window. The figure below depicts the encapsulation of foreground and background data in RGB color space. We can see that, in both color space encapsulation is good, but LAB color space is chosen because among the volume enclosed by the ellipsoids, RGB's tend to have a lot of free space compared to LAB. Choosing LAB reduces the wrong classification probability.



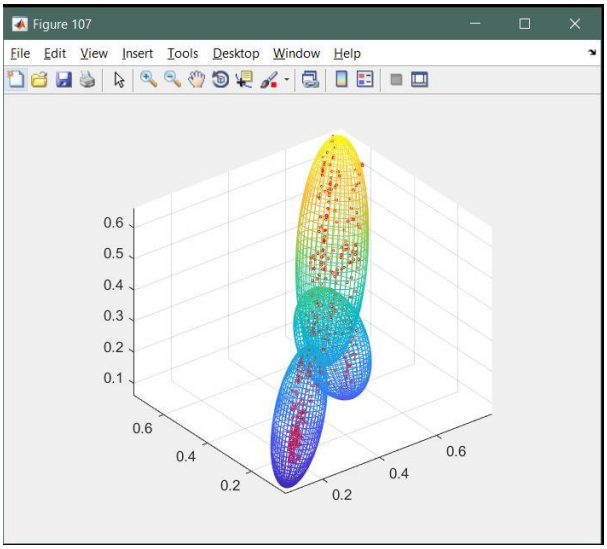


Figure 1: Starting frame- Red points correspond to warped points, blue corresponds to optical flow adjusted points

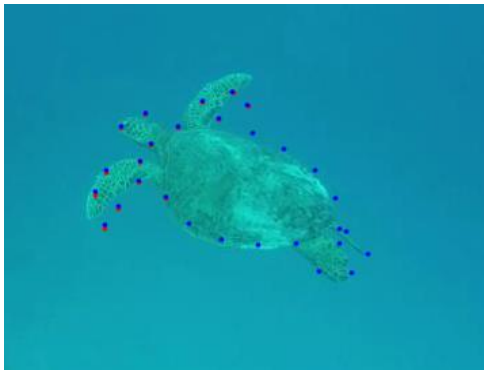


Figure 2: Middle frame

From the above figures, we can see that in the initial frame, there isn't much difference in the warped points and optical flow adjusted points. But, as the turtle moves, the compensation needed by the optical flow part increases. Hence, only warping cannot be used to estimate the new set of points as they wouldn't be accurate. Thus, we need optical to compensate the change needed.

Window centers are moved based on the optical flow calculations.

NOTE: For each window only, the foreground corresponding optical flow is considered and is averaged and added to the center point of that window.

Once we have the new set of window centers, we next calculate the probability mask, color confidence and shape confidence for each window. Figure below shows the probability mask and shape confidence.

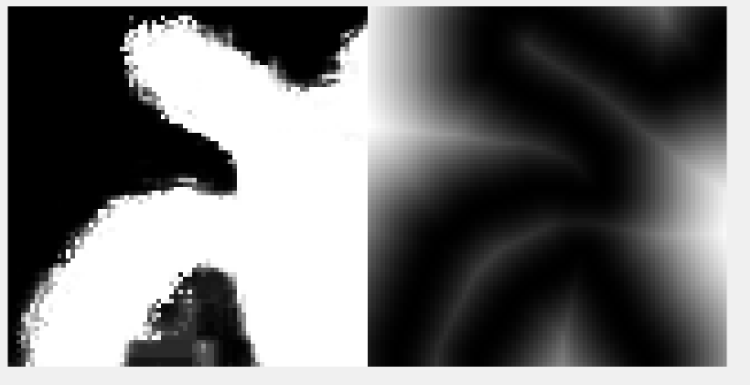


Figure 3: Left side: total probability mask for a window & right side: shape confidence for that window

While calculating the probability mask for each window, the foreground GMM model is kept constant, while the background model is changed. This is in accordance to the paper. This is because, we assume the windows to move with the object, thus the foreground information should be the same in every frame. What can change a lot is the background information, thus, we change the background model, and this has proved to be very helpful.

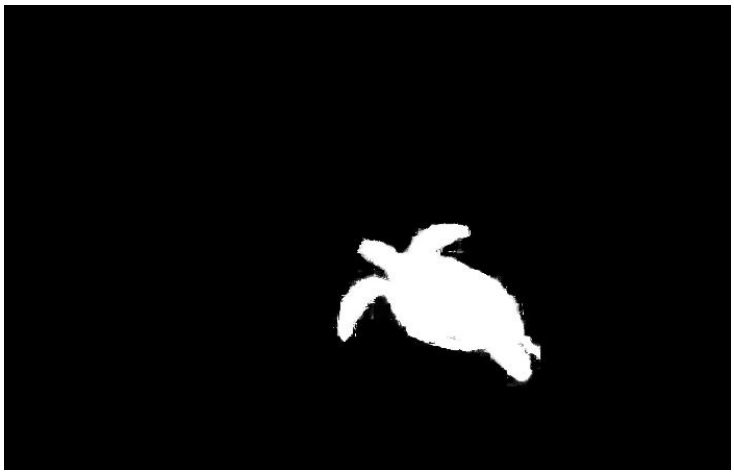


Figure 1: Total foreground probability for entire image

This is the final foreground probability mask of the entire image of a particular frame. This mask is used as an input for the next frame. The cycle continues.

Sample images of turtle being tracked is shown above.

OBSERVATION: The datasets in which the background changes a lot are difficult to handle. The datasets 1,5,4; background was changing a lot in compare to foreground, thus it was getting difficult to maintain the foreground mask accurately.

Second dataset: Motorcycle

The result was bad here. We could cover very few frames. We tried different sets of window width and number of windows, but the one below seemed to work well.

Window Width: 60

Number of windows: 40

In this case, the difference between the two consecutive images was more. By difference we mean deformation of the object. Hence, the affine transformation wasn't accurate in transforming the windows. Hence, the foreground coverage was not accurate. This propagated across the frames, eventually leading to no foreground identification (no windows covered the foreground eventually).

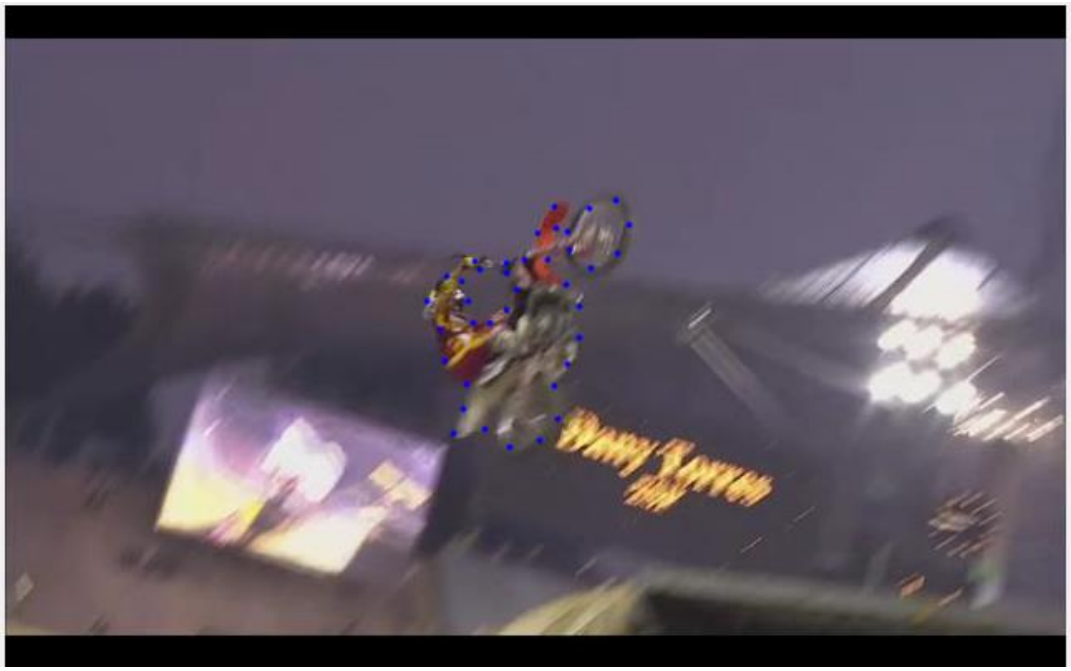


Figure 2: initial frame

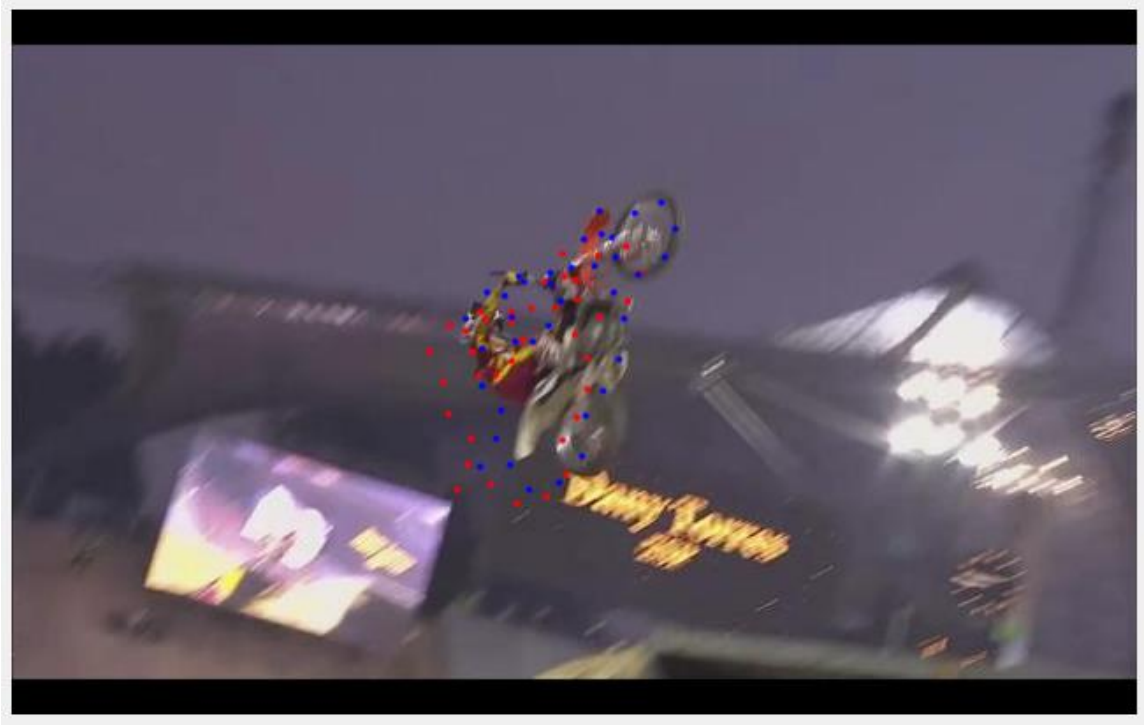


Figure 3: 2nd frame. We can see that though affine (red dots) fails here, optical flow (blue dots) compensates for the error.

Figure 4: Lots changed here. Even optical flow isn't able to track the changes.

Third dataset: Gymnastics

This probably was the probably the second difficult dataset. The gymnast could be traced for around half the total frames given. This is similar to dataset2. The object was rotating and deforming a lot around the middle. Hence, keeping track of the object through affine is difficult. Also, in this case the background was changing a lot. Hence, it is harder to keep the foreground mask accurate while propagating through frames. We believe if more of iterative refinement could be done on this, the result may be improved.

Fourth dataset: Heavyweight lifting

This was also pretty much easy dataset as the variation in color composition between foreground and background was not too much. Thus, there weren't much of misclassification, though there are some. In this case, the segmentation worked well. Note that here there wasn't much of rotation of the object as such, but there were deformations but not at the scale of those in dataset2&3, but, affine could capture these changes along with shape model making it work better in this case.

OBSERVATION: For datasets which involves rotation of the object the results are bad. What we observed was that the rotation was leading to transformation which upon applying to masks were not producing accurate results. Thus, in this case, mainly dataset2&3, the objects were rotating and thus tracking it through affine was difficult.

Fifth dataset: Lizard

Window Width: 60

Number of windows: 30

In this dataset, the background was uniform, foreground was uniform, thus segregating foreground and background based on color and shape model is quite accurate. However, there are certain regions (shadow) of the lizard, seen on water which are sometimes classified as foreground. This is because these regions are closer to foreground than background in the given dataset. Thus, there were some classifications as foreground that were supposed to be background. This is again different from dataset2&3. Not much of rotation involved. Few images from this dataset are given below.

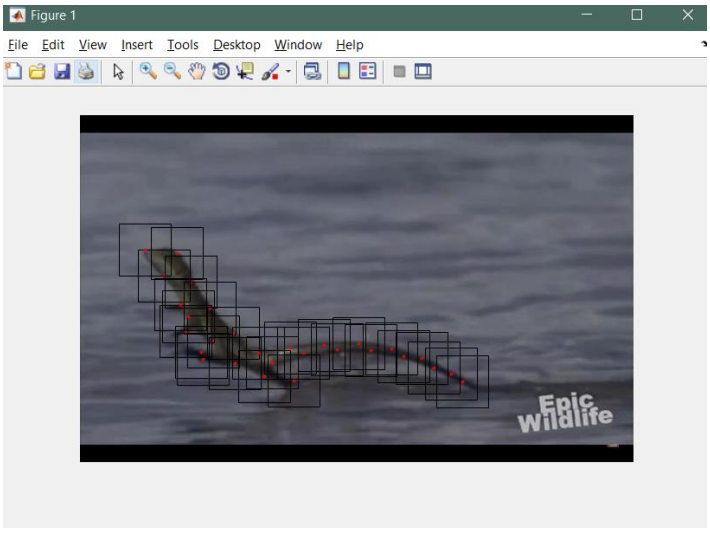


Figure 5: Window placement

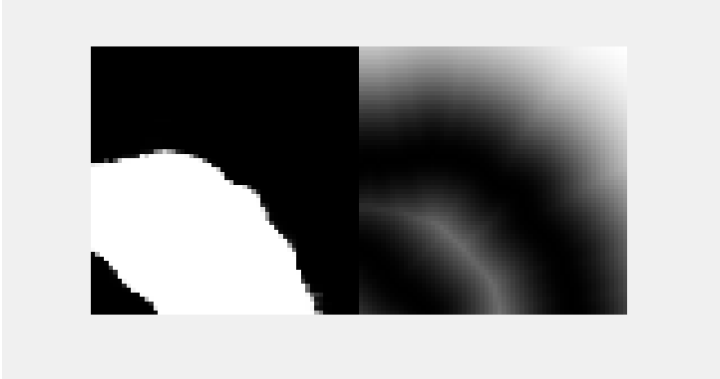


Figure 6: Left – total foreground probability mask. Right- shape confidence