Object Tracking

Yona MELLUL (3800646)
Léo PORTE (21200044)
Hugo RIFFAUD DE TURCKHEIM (3800256)

January 2024

1 Mean Shift

1.1 Q1 EXPERIMENT

The Mean Shift algorithm is a non-parametric feature-space analysis technique, a method for locating the maxima—the modes—of a density function given discrete data sampled from that function. It is used for clustering and can also be applied to tracking in computer vision.

In the context of tracking, the Mean Shift algorithm involves the following steps:

1. Color Space Conversion: The algorithm starts by converting the color space of the image to HSV, as the hue component is more robust to changes in lighting conditions compared to RGB space.

2. Histogram Backprojection: It creates a histogram model of the feature space (e.g., color histogram of the hue component in the target region) and backprojects this histogram onto the image to obtain the probability or likelihood image.

3. Iterative Region Localization: The algorithm then iteratively shifts a window (Region of Interest - ROI) towards the peak of the density or the highest probability in the likelihood image. This is done by computing the mean (centroid) of the pixels within the window and shifting the window towards the mean. The process repeats until convergence, i.e., the shift is smaller than a predefined threshold.

After experiments on the mug video, we can see that the performance of Mean Shift is highly dependent of the initialization of the window. Poor initialization can lead to convergence to local maxima or tracking failure. In our experiments, the tracking starts well in the window is small (entirely in the mug) or adjusted to the mug size and shape, and later the focus is lost. If it is too large, the tracking is lost instantly.

We can see that on these tests:

Figure 1: Test 1, with focus lost at the middle of the video

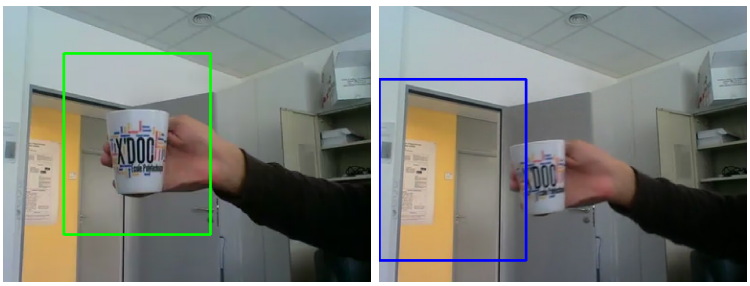

Figure 2: Test 2, with focus lost at the beginning of the video

The advantages of this algorithm are that it is easy to implement (few LoC using OpenCV), and the computational time is anecdotic.

1.2 Q2 ANALYSE

Here are our results:

Figure 3: Original (left), hue component (center) and back projection (right) visualization

The hue component image represents the color type of each pixel independent of lighting intensity and saturation. This property makes hue a robust feature for tracking objects under varying lighting conditions and is particularly useful when the object of interest has a distinct color compared to its surroundings, as seems to be the case with the red ball in the original image.

In the hue image, one would expect the ball to stand out due to its distinct color. However, the actual hue image shows that while the ball is noticeable, other areas in the scene also have similar hue values. This indicates that there are other objects in the scene or reflections that share a similar hue to the ball, which could potentially interfere with tracking. These objects are obviously legs playing with the ball, and the camera movement induces that objects in the background are moving as well, causing a lot of noise.

The back-projection image is a representation of the likelihood of each pixel belonging to the object, based on the hue histogram of the object model. Ideally, in the back-projection image, the object being tracked would have high values (bright areas) while the background would be dark. However, the provided back-projection image shows a lot of bright spots scattered throughout the scene, not just where the ball is. This suggests that there are several regions in the scene with hue values similar to those of the ball, which then result in a less accurate tracking. As said before, it is because of legs and camera movements.

Dynamic Histogram Update Mechanism improvements
After every update_rate frames, our script updates the histogram model of the object. It does this by recalculating the histogram for the current RoI and then blending this new histogram with the original histogram to accommodate appearance changes.

This dynamic updating approach helps the tracking algorithm stay adaptive to changes in the object's appearance, potentially improving tracking accuracy over time. However, it's important to balance the update frequency and the blending weights to avoid rapid fluctuations in the model histogram that could destabilize tracking, especially in noisy or dynamic environments.

2 Hough Transform

2.1 Q3 CALCULATE

Here are our results:

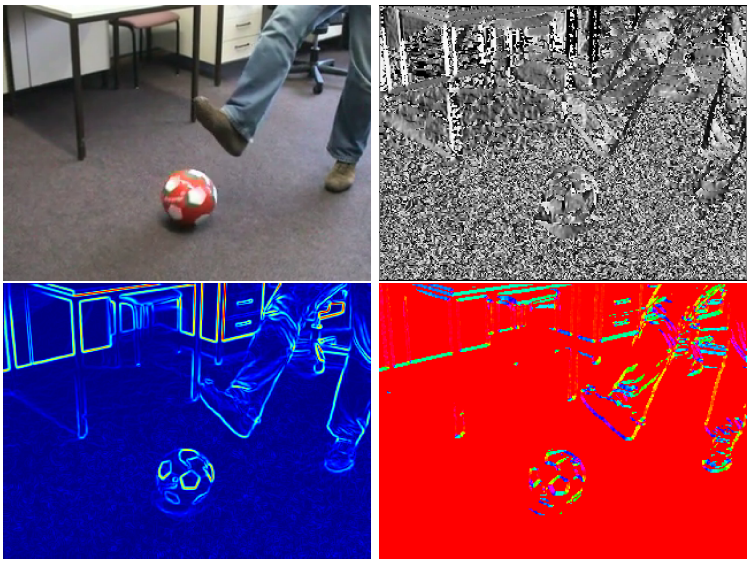

Figure 4: Computing the index of the vote (gradient orientation), and selection of the voting pixels (using the gradient norm).

Our results are pretty accurate with what is waited in the practical's PDF. Our gradient orientation is just a few more noisy, but that is because it is not the same exact frame as the one in the PDF. Same standings can be statued for other pictures.

2.2 Q4 Build

Here are our results:

Figure 5: Test n°1

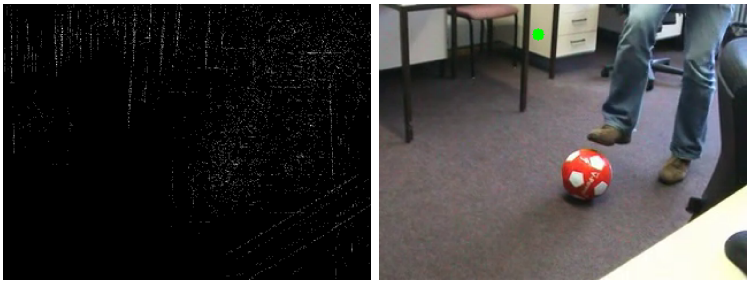

Figure 6: Test n°2

Comparing our results to the ones obtained in the PDF, we can say that at the frame when the screenshot was taken, the detection managed to work similarly. We defined the window initialization in the same shape and size as the one in the PDF. Regarding Hough transform now, it seems that our solution shows the same lighting dots, even though the picture of the PDF is smoother.

Now, the other test on the ball video shows a misplaced detection. On the Hough transform, there are some pixels representing the ball, but not enough to surpass the noise caused by the camera movement and the objects in the room which are moving indeed.

3 Deep Features

3.1 Q6 EXPLAIN

First of all, we consider selecting a well-known and widely used network like VGG-16 or ResNet-50. These networks are trained on large datasets like ImageNet and have proven to be effective in extracting rich feature hierarchies suitable for a wide range of tasks, including object recognition and localization.

For the purpose of tracking, intermediate layers of the chosen network are the most beneficial. These layers tend to capture more complex features than the initial layers but are not as highly abstracted as the final layers. For instance, in VGG-16, layers like 'conv3_3' or 'conv4_3' could be a good starting point. These layers capture complex textures and patterns that can be more distinctive and robust for tracking objects under various transformations and occlusions.

For the channel selection, we propose the use of dimensionality reduction techniques or feature selection methods like PCA to identify the most informative channels. Alternatively, channels could be selected based on prior knowledge of the type of features that are most relevant for the objects being tracked in our specific application (mug or ball in our case).