INFOMCV Assignment 3

Olga Capmany Dalmàs and Julian Verouden

Summary

For this assignment it was decided to use the code base of the assignment 2 that was provided to us. As for how the voxel space was established, for each camera, we iterate over each voxel of the 3D grid and obtain the corresponding index of this voxel, which will correspond to the index that was previously used to create the look_up_table. Next, the lookup_table is used, which contains information about the projection of the voxel (given the specific index) in the current camera view and obtains the projection coordinates of the voxel in the image.

It then checks if the projection coordinates are within the boundaries of the foreground_image and if the corresponding pixel is marked as foreground in the foreground picture. If the projection coordinates are outside the foreground image boundaries or the corresponding pixel in the foreground image is not active, disable the voxel.

Once we have located the 4 persons in the grid, we used K-means to identify the 4 clusters, only taking into account the x and z coordinates. For K-means we do 100 iterations to increase our chances of finding better clusters.

After the clusters are identified, the colour models are made for each one. In our case, this was based on the colour histograms. For each cluster, 3 identifying histograms (H, S and V) were extracted from the pixels that correspond to the 2D projection of the voxels of each cluster from a specific view. The head and bottom half of a person was not taken into account for this.
The number of bins was chosen by trial and error until optimal and efficient results were found. For each colour channel we use a histogram with 20 bins.

To perform the matching between clusters when viewing different frames, the 3 identifying histograms for each cluster were calculated and then compared respectively using the Bhattacharyya distance with the histograms of the already labeled clusters. The difference between the histograms was summed. This calculation was performed for each camera view used and also summed, yielding a value for each cluster. The Hungarian algorithm was then used to match the clusters based on these distances.

Link to video

We have three videos, since we opted to do choice task 3 and our final run only has one version.

Without tracking based on the previous position, version 1: https://youtu.be/-kpdcq0rumQ
With tracking based on the previous position, version 1: https://youtu.be/hkd6F3MzEtE
With tracking based on the previous position, version 2: https://youtu.be/EG1rOEKEhAI

Trajectory image

We have two types of images, a plot with arrows to show the paths between each step, and one with only the dots where centers are observed.

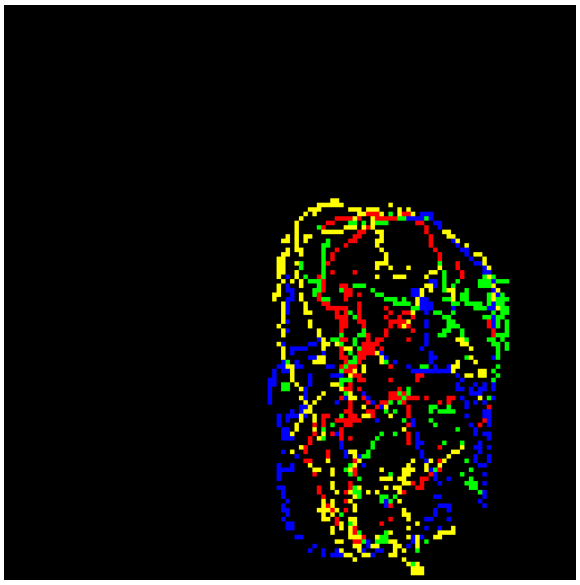
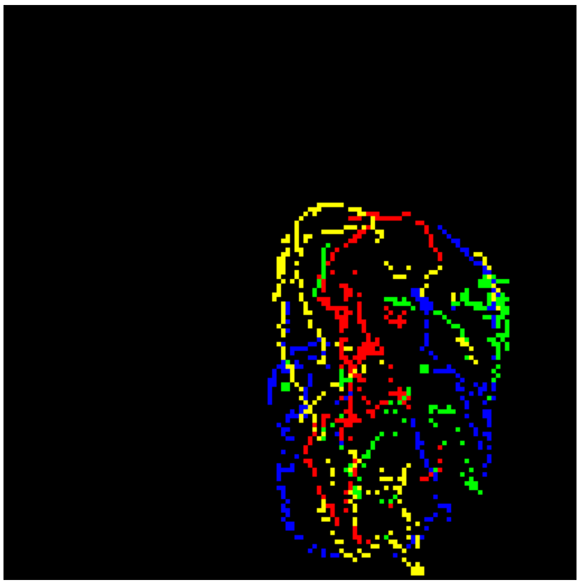
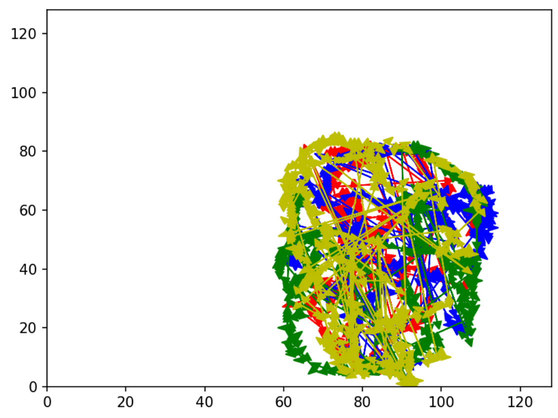
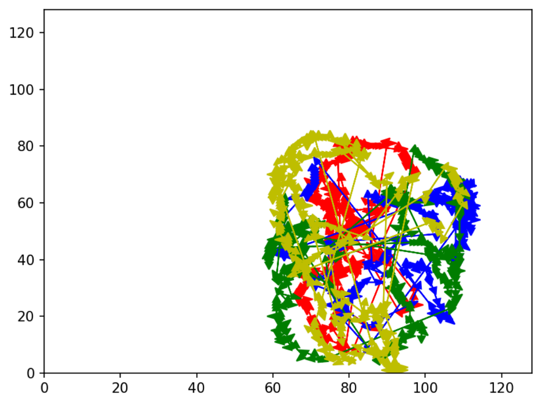
	
Dots, without tracking	Dots, with tracking
	
Arrows, without tracking	Arrows, with tracking

Figure 1: Result of the first run, with time skips and formed by taking every third frame. Note that the arrows that are green should be blue, and vice-versa.

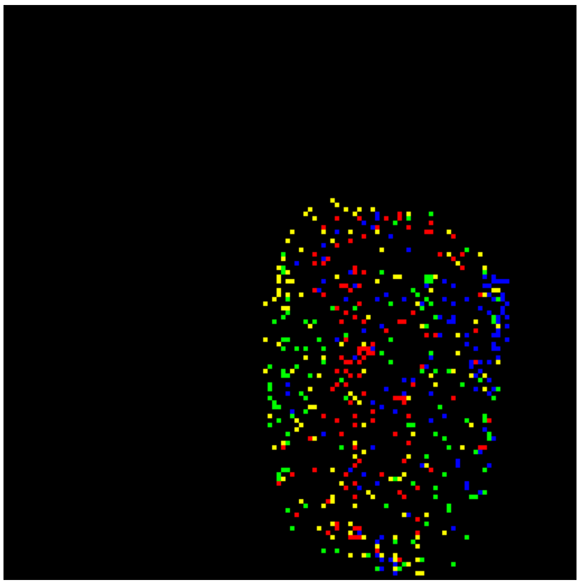
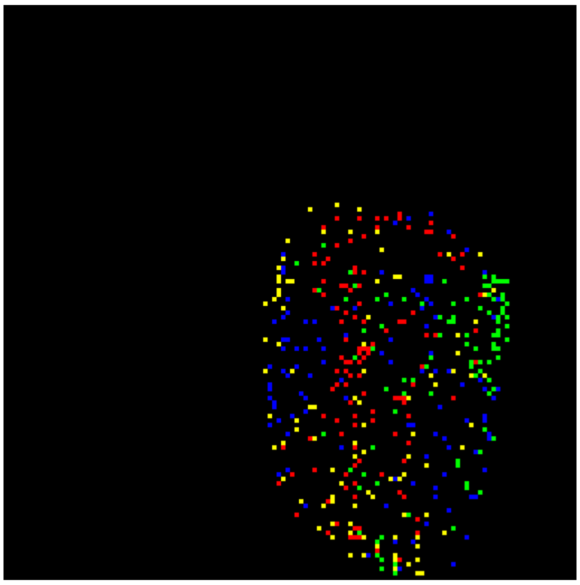
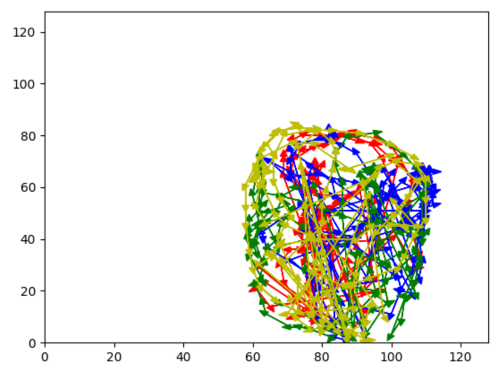
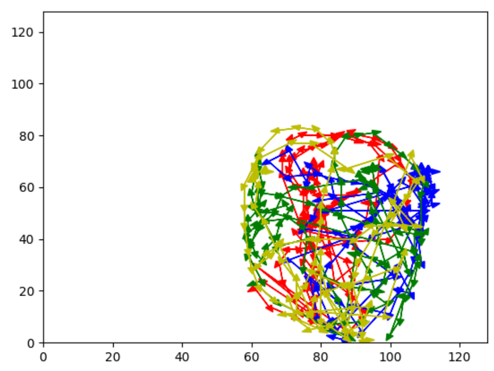
	
Dots, without tracking	Dots, with tracking
	
Arrows, without tracking	Arrows, with tracking

Figure 2: Result of the second run. No time skips and using every 20th frame. Note that the arrows that are green should be blue, and vice-versa.

Choice tasks

<u>**CHOICE 1:** Use multiple cameras to increase the robustness: 15. You need to combine the information in the matching step</u>.
As explained above, our matching step is based on a distance value obtained by summing the distances between the 3 histograms respectively (H, S and V). Thus, to perform this step we simply summed the distances obtained for each camera used. In this way, the differences with the histograms obtained from different perspectives are taken into account, and in the case that two clusters overlap in that frame, not only that difference will be taken into account, which in this case will not be realistic, but also those of the other perspectives.

CHOICE 5: Get rid of outliers and ghost voxels: 10 Based on the clustering, find out which voxels should be removed altogether so they don't influence the calculation of the center.

In our case we have decided to do a preprocessing of the voxels marked as person before doing the clustering. To do this we analyze all (x,z) coordinates where a voxel is present somewhere along that column. This results in a binary image where all white pixels have a voxel in 3D space along the y-axis at that coordinate. We perform erosion on this image to separate clusters more and to remove small outliers/ghost voxels. We then use cv.connectedComponentsWithStats to find all connected areas. We discard all connected areas where the area is less than 15, which takes care of extra ghost voxels and outliers.

CHOICE 3: Implement tracking based on previous position and/or movement: 5 Only eligible if you show the results with and without tracking in your trajectory image and video.

To make this choice task the main idea is based on taking into account the center of the previous position to know if the new center makes sense and to mark it as part of the trajectory or not. For this we subtract the x and y coordinates between the possible new point and the point of the previous trajectory respectively. The absolute value of this is used and a threshold is set to determine that if it is further away from this it will not be taken into account as part of the trajectory as it is probably an outlier. As seen in figure 1 this results in a much cleaner trajectory where outliers are skipped and not as many arrows go from one side of the voxel grid to the other.
This works better when the amount of frames skipped between samples is smaller, since the difference between changing from cluster A to cluster B is comparatively larger than the difference a person would have moved between samples.

Discussion:
As seen in our first two videos there are large jumps where a large portion of frames is skipped. We had this issue on one of our computers, but not on the other.
Since we did not have time to make the videos we needed after making some improvements we included all three videos.
Some other small improvements were also made in this later version, like the number of iterations in k-means. We sadly did not have time to run the video twice and with the same frame rate again. We therefore only have a version with fewer samples, and only one video of this newer version.

We think that our result would have improved greatly if we had tried to handle occlusion. From what we observed, most times when the tracking goes poorly it happens when two people are standing in front of each other in line with one of the cameras.

In choice task 3 we could have tried using cv::KMEANS_USE_INITIAL_LABELS as a flag in the cv.kmeans function. We are not sure if this would have worked, but this way you should be able to use the previous position of the centers in the next step.