**Basic Setup**

**Intrinsics & Extrinsics**

Since cameras remain common from Assignment 2, intrinsics were carried over from previous generation. The code still exists and can be run again if necessary. Similarly, extrinsincs were generated in the exact same manner as the previous assignment but using the new images from the new camera angles. Checkerboard corners were picked manually, using the same code framework as Assignment 2.

**Background Subtraction**

The background image was created by taking every single frame in background.avi and combining them by taking a mean of all frames in Photoshop. This generated a final png file that was less noisy and more likely to yield better results during background subtraction. The png was subjected to light Gaussian Blurring, to reduce artefacts in the image, specifically at certain edges.

The foreground was then extracted using the same code from Assignment 2, built upon the existing basic subtraction, as well as a number of additional image-processing functions which were previously implemented to further clean up the image and isolate the foreground more precisely (erosion, dilation, and blob detection). The foreground images from each camera are then used to create the 3D scene reconstruction of visible voxels.

**Offline Colour Models**

In order to generate colour models, two steps are necessary. Firstly, the list of visible voxels must be clustered in such a way that each cluster corresponds to a separate person in the video (out of four). Following this, each voxel’s projection onto the camera contributes a colour to the colour model of its cluster. In this way, a colour model is generated for each person, and can be used during runtime to match clusters to that person and track them accordingly.

For generating the initial colour models, which will be used as references during the online segment, the most important thing to consider is separation of the people in the frame. This ensures that clusters are very clearly defined and reduces occlusion (which can skew colour models) as much as possible. Since this part of the assignment is offline, 4 frames were hand-picked (one for each camera) at various points in the video, prioritising separation of the subjects in that camera viewpoint.



Figure 1 - Left to Right the frames picked for Cameras 1,2,3,4

**Clustering**

For each of these frames, the 3D scene is reconstructed. The visible voxels are then projected onto the 2D floor (height is ignored), and clustered together using k-means. The initial centres for the k-means iterations are found using the k-means++ algorithm (by David Arthur and Sergei Vassilvitskii), and k-means is run 10 times, with the best result being taken. This is done in order to (as much as possible) avoid cluster centres being stuck in local minima. To carry out k-means, the OpenCV function kmeans() was used, with a repetition parameter of 10.

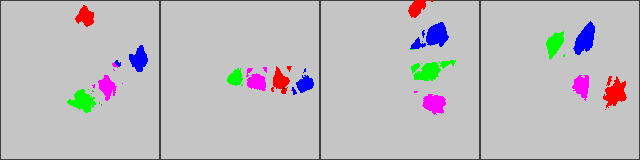


Figure 2 - 2D Projections of Clusters of initial frames (scaled down by voxel step size)

The basic process is simple. First, iterate through each visible voxel in the reconstructed scene, and create a 2D point corresponding to the x and y coordinates of that voxel. The list of 2D points is then passed into k-means, which generates a list of labels (one per voxel) that assigns voxels to their cluster, and a list of four centres.

1. **for** (**int** i = 0; i < m\_visible\_voxels.size(); i++)
2. {
3. data.push\_back(cv::Point2f(m\_visible\_voxels[i]->x, m\_visible\_voxels[i]->y));
4. }
6. Mat labels, centers;
7. cv::kmeans(data, cluster\_num, labels, TermCriteria(TermCriteria::COUNT + TermCriteria::EPS, 4, 0.1), 10, KMEANS\_PP\_CENTERS, centers);

After clustering, the list of labels is then passed into the colour model generation functions, which generate the four offline colour models, that will later be used as reference points to calculate distance and identify people.

**Offline Colour Models**

\*ADD THIS TO COLOUR MODEL BIT\* One crucial factor to consider is that, during the initial colour model generation, it is impossible to know which cluster belongs to which person for either frame. This causes issues if simply taking cluster centres in order, because the generated models might be using clusters that belong to other people. As such, the problem was solved by manually defining which cluster belongs to whom for the first four frames, since cluster centres were found to be deterministic for these initial reference points.