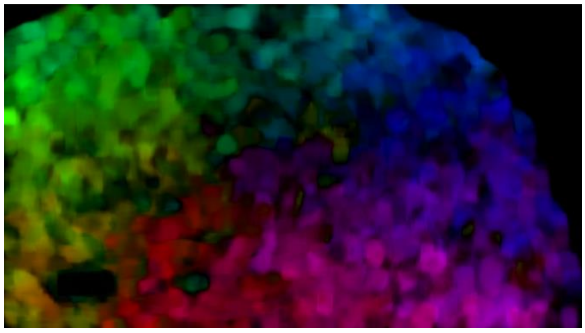
DIGITAL IMAGE PROCESSING

PROJECT REPORT

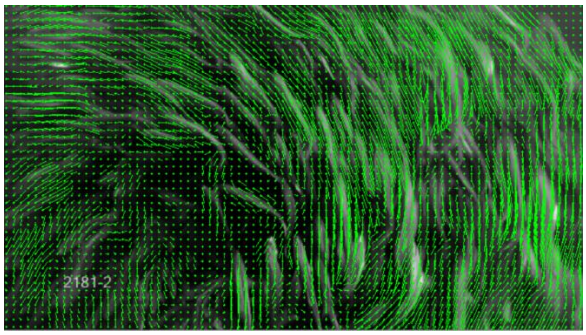
SAI DEEPAK REDDY K	19/11/EE/023
CHOKKARI DINESH	19/11/EC/062
MOHIT KHUSHLANI	19/11/EE/026
HARISH PETKAR	19/11/EE/038
KUMAR PRATYAY	19/11/EC/063

FLOWCHART

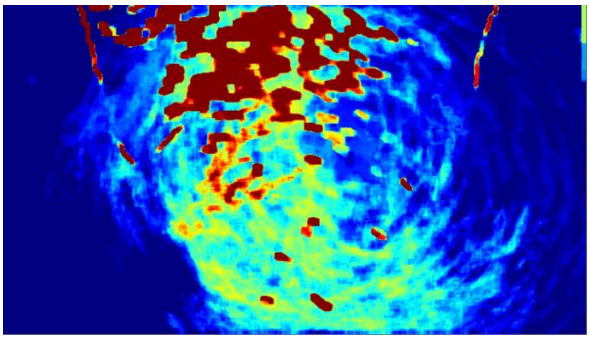
Original Video Frame



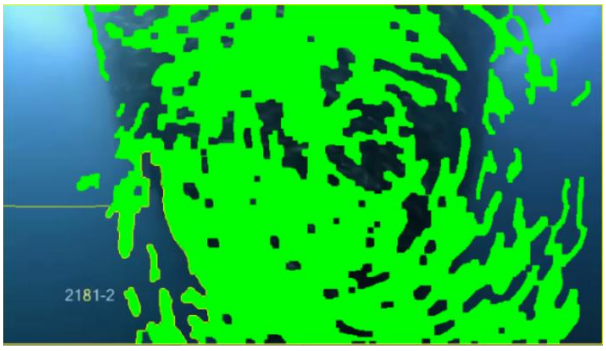
Optical Flow Frame



StreakFlow Frame



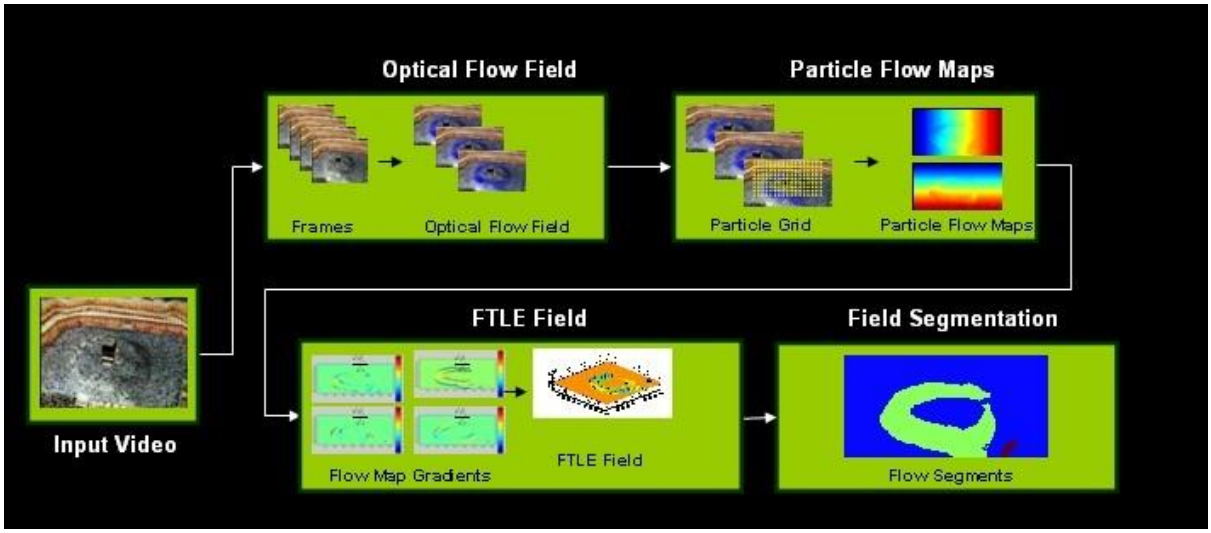
Heatmap Frame



Watershed Frame

DOMINANT CROWD FLOW SEGMENTATION

Dominant crowd flow segmentation is the first step towards building an automated monitoring system for high density crowd scenes. In computer vision, optical flow is widely used to compute pixel wise instantaneous motion between consecutive frames, and numerous methods are reported to efficiently compute accurate optical flow. However, optical flow does not capture long-range temporal dependencies, since it is based on just two frames. In order to achieve an accurate representation of flow from crowd motion, we use the streaklines to compute a new motion field which we refer to as streak flow. To compute streak flow, streaklines are computed by temporally integrating optical flow.



Block Diagram

Algorithmic Steps to compute Crowd Flow Segmentation:

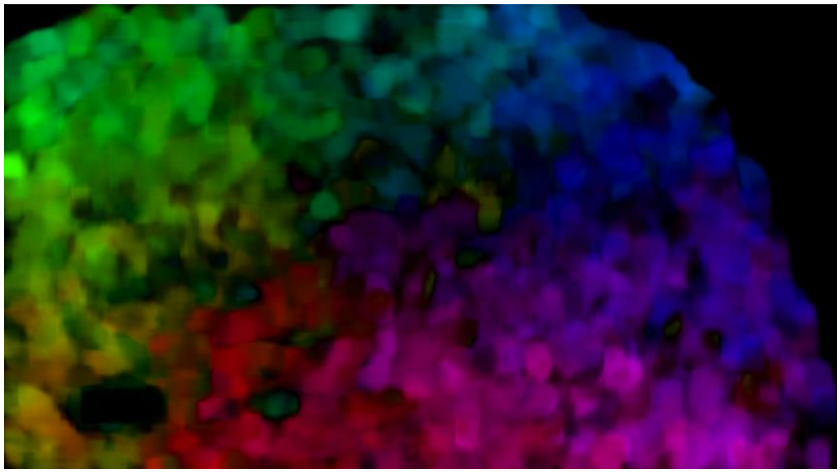
DENSE OPTICAL FLOW:

The motion of video which is within the set of frames, basically the objects might be moving. Now when the objects are moving, they get displaced from its original location. If you simply compute the intensity differences between the frames that also gives you the notion of motion. It tries to capture the displacement of the moving objects. And when we try to represent it in a color coded way we can see different intensity values and what you see in the optical flow or the maximum flow or maximum movement is happening.It is a 2D vector field where each vector is a displacement vector showing the movement of points from first frame to second. In our python program the code we use for visualizing Optical flow in our dataset is based on the farneback algorithm

which needs a 1D input image, so here we convert BRG image to Grayscale. Then use the function cv.calcOpticalFlowFarneback(). And we color code the result for better visualization. The Direction corresponds to Hue value of the image and the Magnitude corresponds to Value plane.

Optical flow works on several assumptions:

1. The pixel intensities of an object do not change between consecutive frames
2. Neighbouring pixels have similar motion.



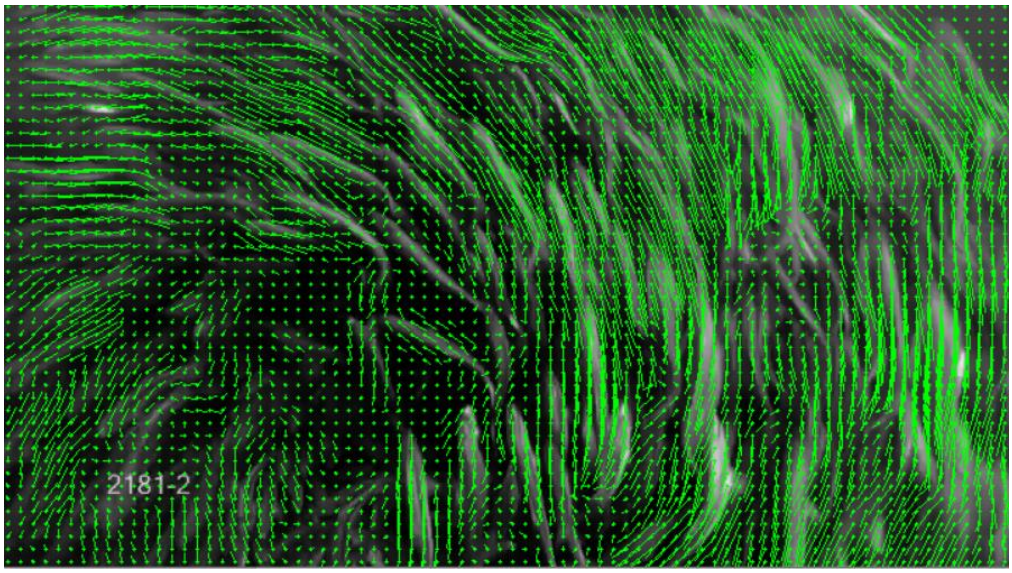
Optical Flow Frame

STREAKLINES:

Streaklines are obtained by repeatedly initializing a fixed grid of particles at each frame, then moving both current and past particles using optical flow. This leads to a representation of the flow that more accurately recognizes spatial and temporal changes in the scene, compared with other commonly used flow representations. It gives the directionality of the motion. The displacement, which is happening between two consecutive frames and taken over the set of frames. This is nothing but an orientation histogram which has been plotted. It previews directionality shown with small small arrows. Streaklines are instantaneous and capture the continuity of motion.It is an instantaneous vector field which represents the accumulative motion of the scene. It resembles the temporal average of optical flow but it is more imminent.

Streakflow relates to group motion, instantaneously reacts to changes and has far less noise than optical flow.In our python program we first change the image from grayscale from previous optical flow to BGR. Then we show Streakflow using the following functions namely cv.polylines() and cv.circle().

Original Video Frame → Optical Video Frame → Orientation Histogram

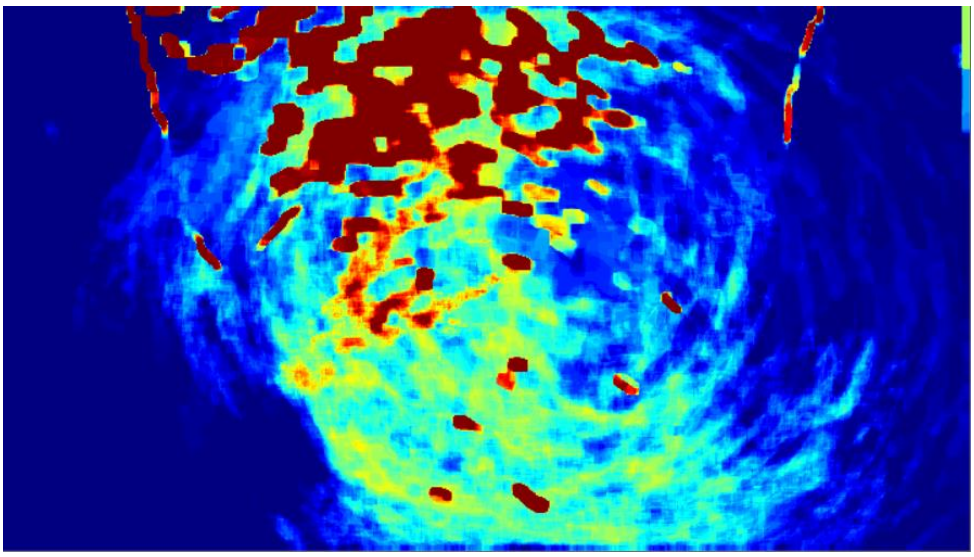


Streakflow Frame

SIMILARITIES:

We find a factors that whether these directionality patterns or some kind of patterns, in the motion, using the peak lines. It is showing the maximum of the flow, which is happening. We compare the similarities between adjacent frames of the video and get a heatmap as an output to show the flow segments in the sequence. Here we show dynamic potential flows with color coded regions for divergent and convergent regions.we use with the help of cv.dilate() and cv.erode() and color coded ColorMap() and then cv.drawContours().

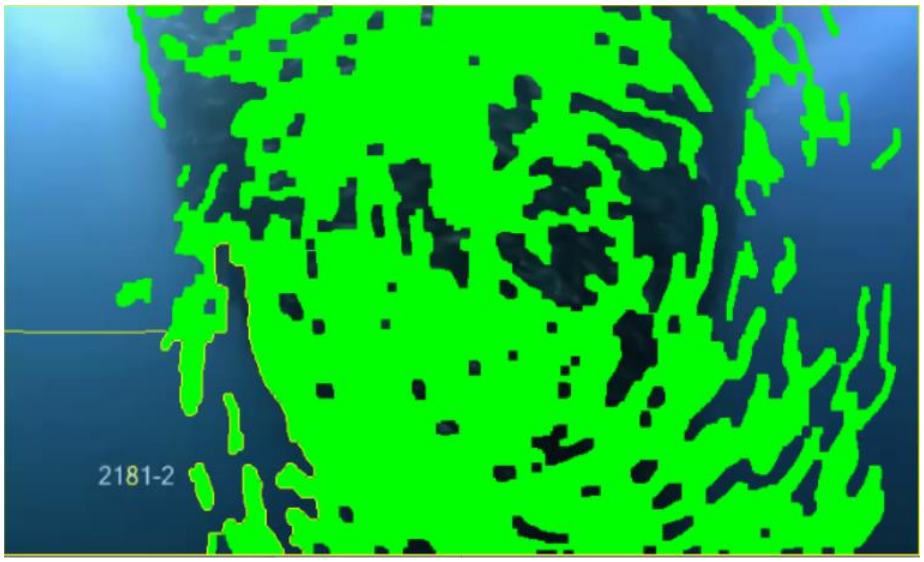
Original Video Frame → Optical Video Frame → Streakflow Frame→HeatMap Frame



HeatMap Video Frame

WATERSHED:

It shows the segregating of the flow pattern which we have captured in the similarities. In every frame of our sequence, we give different labels for our object we know. Label the region which we are sure of being the foreground or object with one color, label the region which we are sure of being background or non-object with another color and finally the region which we are not sure of anything, label it with 0. That is our marker. Then apply watershed algorithm. Then our marker will be updated with the labels we gave, and the boundaries of objects will have a value of -1.Morphological opening and closing is used to remove white noise and holes respectively. The functions we have used in our python program are Function cv.dilate() is used to increase object boundary to the background and cv.watershed() function to get the segmented result.



Watershed Video Frame

Conclusion:

Using crowd segmentation we identify the flow estimation and has its own and real time applications in crowd gathering places like events, traffic etc.

References:

R. Mehran, B. E. Moore, and M. Shah, “A streakline representation of flow in crowded scenes,” in Computer Vision—ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5–11, 2010, Proceedings, Part III, vol. 6313 of Lecture Notes in Computer Science, pp. 439–452, Springer, Berlin, Germany, 2010.

https://www.crcv.ucf.edu/projects/streakline_eccv