

Motion Estimation

Four Clip Types were used:

- **LOW** (static), **MODERATE** (steady motion), **HIGH** (camera move/pan), **FLICKER** (illumination).
- Per category: **decision counts** and **visuals**; **method comparisons** are discussed.

LOW - Decision Counts

- **Full :-** LOW = 260 , MEDIUM = 38 , HIGH = 124
- **Diamond:-** LOW = 260 , MEDIUM = 38 , HIGH = 124



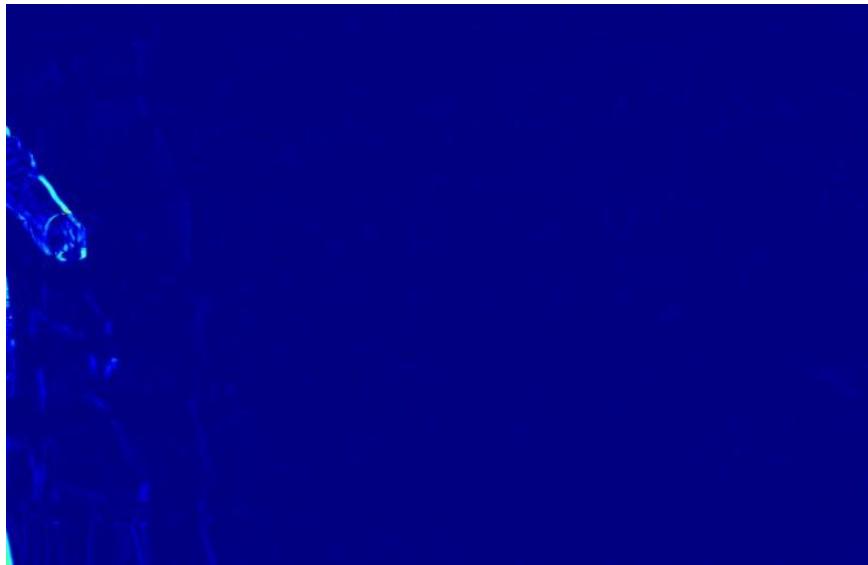
LOW – Differential Heatmap



Low - Motion Vectors

MODERATE - Decision Counts

- **Full** :- LOW = 94 , MEDIUM = 26 , HIGH=124
- **Diamond**:- LOW = 94 , MEDIUM = 26 , HIGH=124



MODERATE - Differential Heatmap



Moderate – Motion vectors

HIGH - Decision Counts

- **Full** :- LOW = 115 , MEDIUM=15, HIGH=198
- **Diamond**:- LOW= 115 , MEDIUM=15, HIGH=198



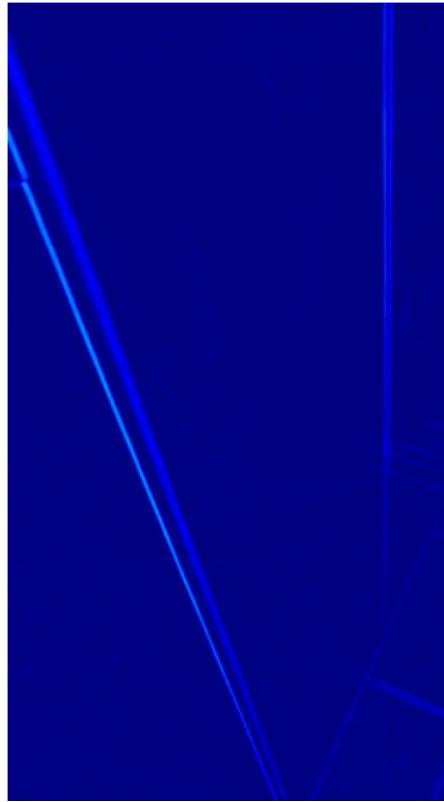
HIGH – Differential Heatmap



HIGH – Motion Vectors

FLICKER - Decision Counts

- **Full**:- LOW = 256 , MEDIUM = 14 , HIGH = 7
- **Diamond**:- LOW = 256 , MEDIUM = 14 , HIGH = 7



FLICKER – Differential Heatmap



FLICKER – Motion Vectors

Discussion of Results

Overall, the pipeline behaved the way a simple codec should. First, note that the LOW / MEDIUM / HIGH decision is made before any motion-estimation method is chosen. That's why the frame counts per class are the same regardless of using Full, Diamond, or TSS. Only the speed and the compensated quality on MEDIUM frames can differ between methods.

Low / static motion.

Most frames were classified LOW, which is good. There is no point doing heavy ME when almost nothing moves. On the few MEDIUM frames, Full search was ~seconds per frame, while Diamond and TSS finished in hundreds of ms. Despite that huge runtime gap, the PSNR of the compensated frames was basically the same (~20 dB) for all three. The heatmaps showed small edges; motion vectors were short and sparse. Takeaway: in calm scenes, fast searches recover the same answer at a fraction of the time.

Moderate motion (person entering/exiting the frame).

This is where ME helps most. We saw a steady stream of MEDIUM frames, and again Diamond/TSS were ~10–20× faster than Full search, with very similar PSNR (roughly 19.6–20.1 dB in my runs). The vector overlays looked clean and consistent with the subject's movement. Takeaway: for "normal" motion, Diamond or TSS give you Full-search quality but much faster.

High motion (camera shake/pan).

Many frames went straight to HIGH, skipping ME. This matches reality, because simple block motion struggles when everything moves a lot at once. When ME did run, quality across methods was similar (around 20–21 dB), but Full was still very slow. Vectors were longer and less stable. Takeaway: for big camera moves, it's usually smarter to use I-frames rather than burn time on ME.

Lighting flicker.

Brightness changed without real geometric motion. The difference heatmap lit up, but vectors stayed short or scattered. The classifier didn't over-trigger ME here, and when ME ran, all methods gave very similar PSNR (~22 dB).

It makes sense because flicker is not motion. Our simple pixel cost treats it as change, but the routing still avoided wasting work.

Method comparison summary.

- Full (exhaustive) = solid baseline but too slow to be practical.
- Diamond = fastest overall with PSNR \approx Full on MEDIUM frames.
- TSS = also fast; usually close to Diamond in time and quality.

What this means.

The classification step is doing the heavy lifting (cheap for LOW, skip for HIGH, ME for MEDIUM). When we do apply ME, Diamond/TSS keep the same quality as Full while cutting runtime dramatically. Visually, the results make sense: sparse/short vectors for low motion, coherent vectors tracking the subject for moderate motion, noisy/long vectors during high motion, and small vectors during flicker where only intensity changes.

Limitations.

This is a learning-level pipeline, not a full codec: no bitstream or entropy coding, thresholds are simple, and searches are integer-pixel. But for the assignment goal, the results are consistent and support the conclusions above.