COP290-A1

Dhananjay Sapawat 2019CS10345

Sachin 2018CS50418

1 Metrics

a)Utility

The error is defined as the average of percentage error.
D1[n] = Baseline Density at a certain frame n
D2[n] = Method Density at a certain frame n
Error = (sum of (D1[n]+D2[n])/D1[n]) at all frame)*100/no of frame

a)RunTime

RunTime is equal to the Time taken by our code to complete.we use std:chrono to find runtime.

2 Methods

a)Sub-sampling frames

Main idea behind sub-sampling is to process every x th frame i.e. if the last frame was N , the next frame to be processed will be the one at position N + x. In our implementation, we skip the processing of next x − 1 frames by using a for loop to go through the frames in the designated video.

b)Resolution reduction

Resolution reduction is performed to decrease the processing time of various functions called on the frame matrix by reducing it size while trying to keep the error minimum. In our implementation, we reduce the resolution of each dimension of the video by a factor of x and process it.

c)Spatial threading

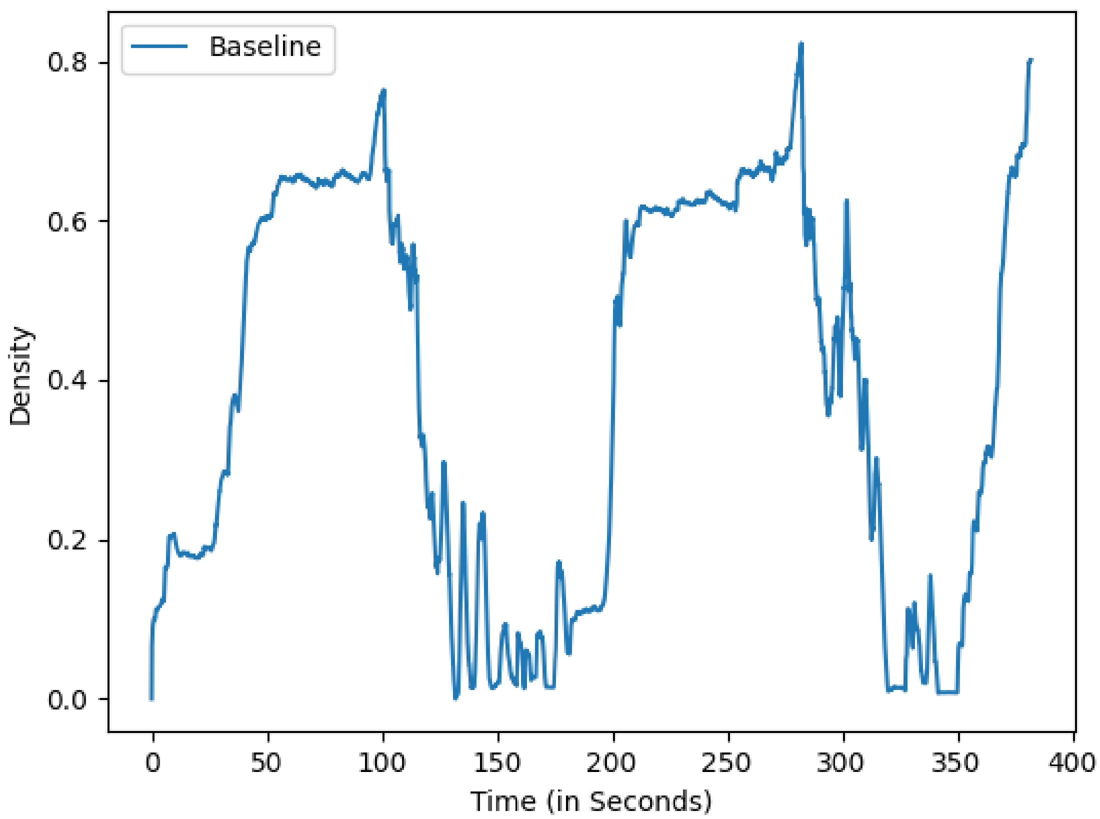
Splitting work spatially by allotting different sections of every frame matrix to different threads is a lucrative method to spread the processing load. The reference frame is read and broken into x segments which is stored in a struct. During the processing of each frame, the frame is directly passed to the thread functions, where based on the thread id, the corresponding segment is obtained and sum of the “white pixels” obtained. In the same loop, the threads are joined and the values obtained are added and the result reported for that thread.

d)Temporal threading

This method involves reading multiple frames one after the other and then processing all of them at the same time. In a single iteration of the loop, x frames are read and corresponding threads created and the frame passed to it. In the same iteration, the threads are combined so that they can be reused in the next iteration.

3 Trade-off Analysis

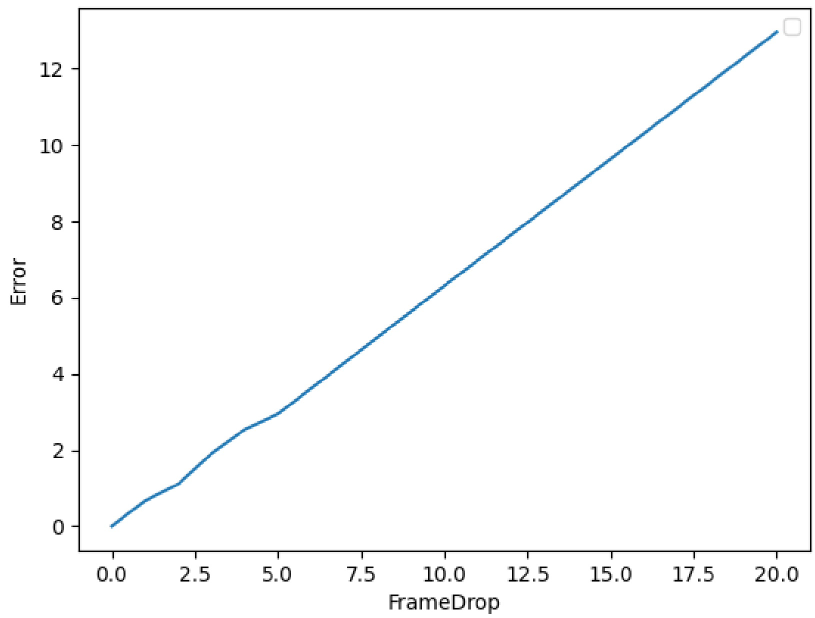
After implementation of all methods, we compare value of queue density of these method with Baseline value.
The result by baseline :
Runtime : 55 seconds

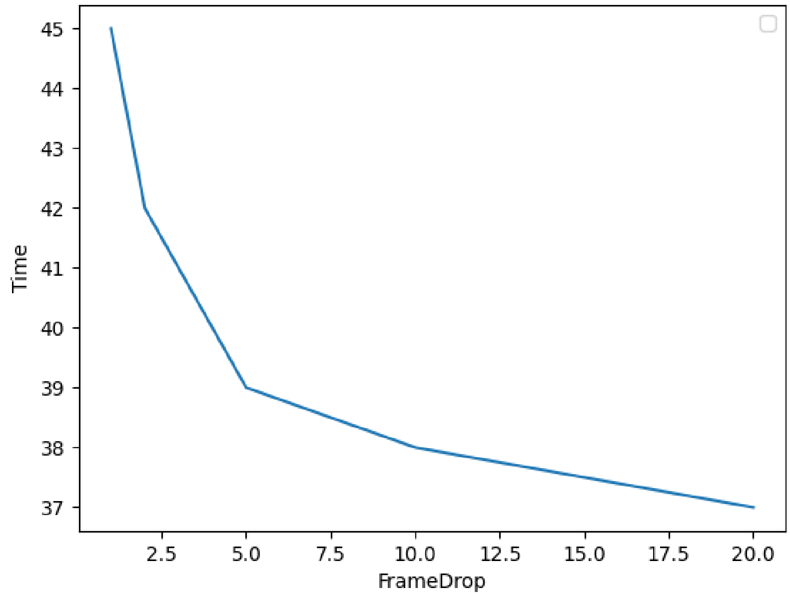


3.1 Sub-Sampling

Frame Skipped	RunTime	Percentage Error
1	45	0.664757
2	42	1.10428

3	41	1.9034
4	40	2.53531
5	39	2.94398
10	38	6.30294
20	37	12.9652

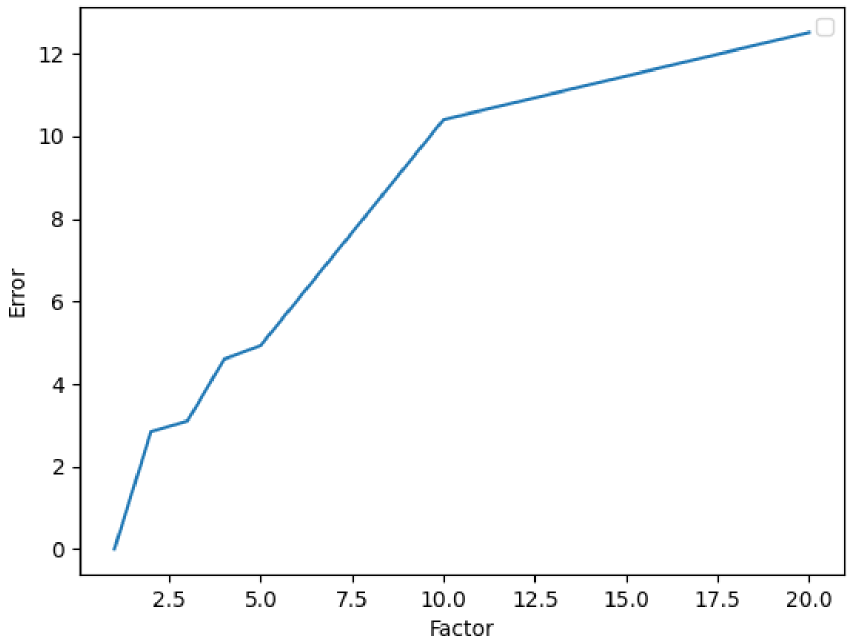


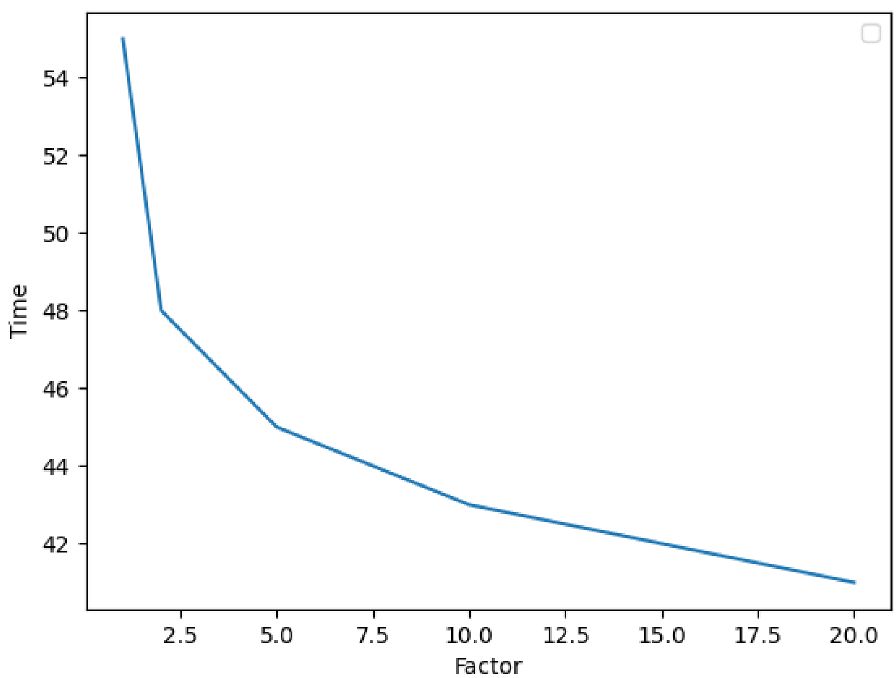


For this method, the parameter is no. of frames skipped. As the parameter increases ,more information loss due which more increase in error. As we are skipping frames , so our runtime will decrease .After 5 frames skipped the decrease in runtime is not much but error is increasing linearly so skipping frames more than 5 is not effective.

3.2 Reduce Resolution

Factor	RunTime	Percentage Error
1	55	0
2	48	3.10473
3	47	2.84394
4	46	4.59764
5	45	4.93068
10	43	10.4044
20	41	12.5174

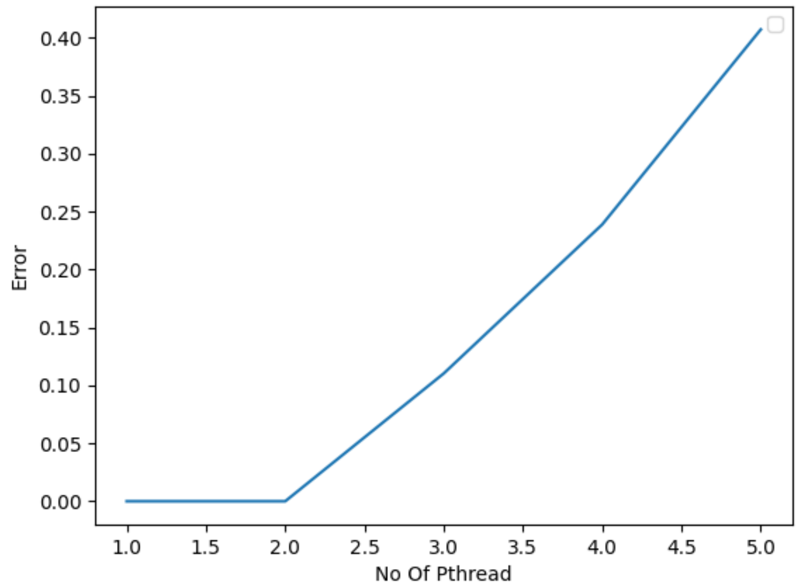


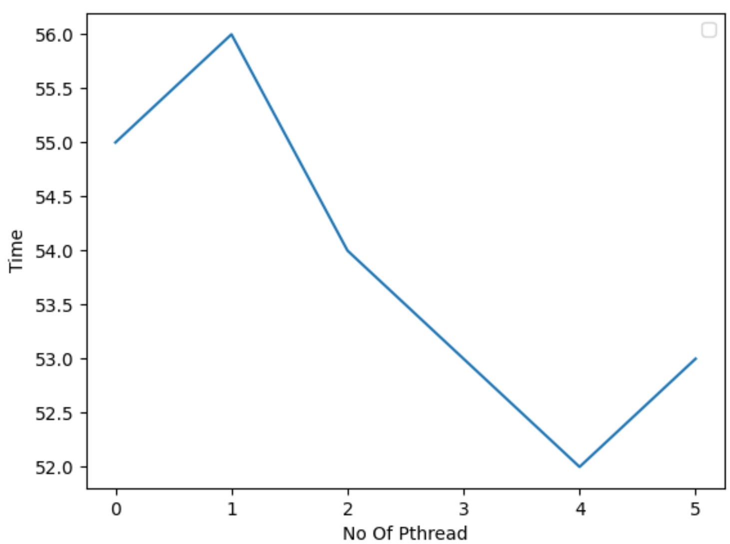


For this method, the parameter is the factor by which we reduce the resolution. To reduce resolution we resize our frame by our parameter. As the frame size decreases processing of frames takes less time due to which our runtime decreases ,as we reduce size we lose some information.

3.3 Spatial Threading

No of Threads	RunTime	Percentage Error
0	55	0
1	56	0
2	54	0
3	53	0.110326
4	52	0.238953
5	53	0.407241

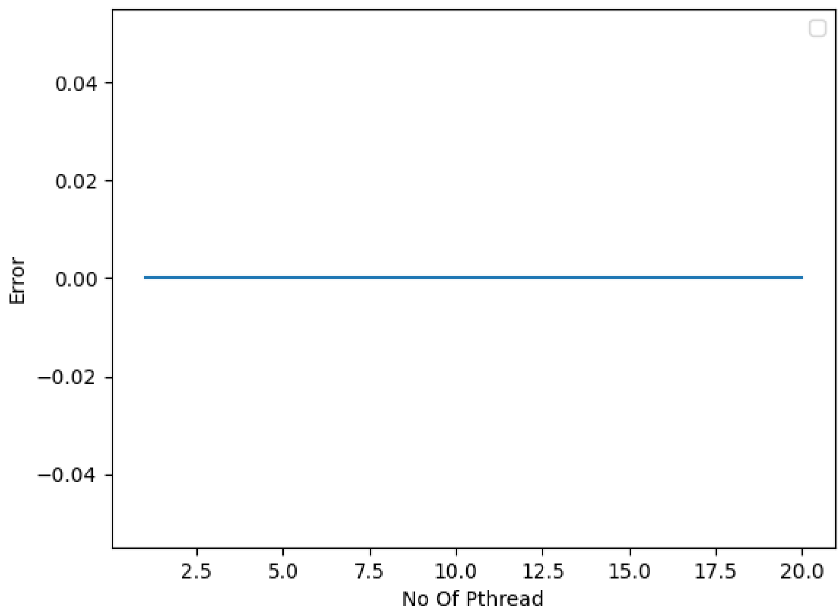


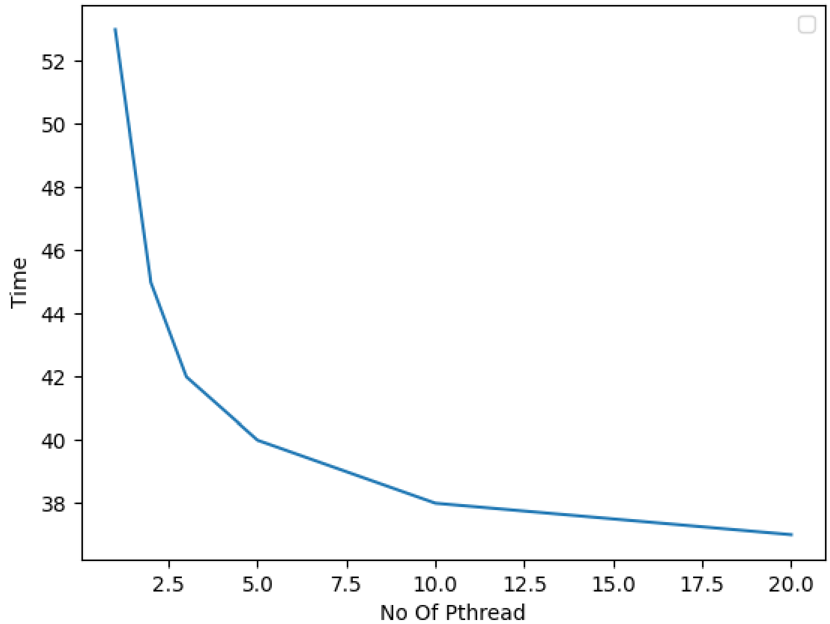


In this method, the parameter is No. of threads. In this we process each frame by giving a part of the frame to each thread due to which runtime decreases . There is no loss of information in this method but due synchronization of Pthread there is error in queue density .

3.4 Temporal Threading

No of Threads	Runtime	Percentage Error
1	53	0
2	45	0
3	42	0
4	41	0
5	40	0
10	38	0
20	37	0

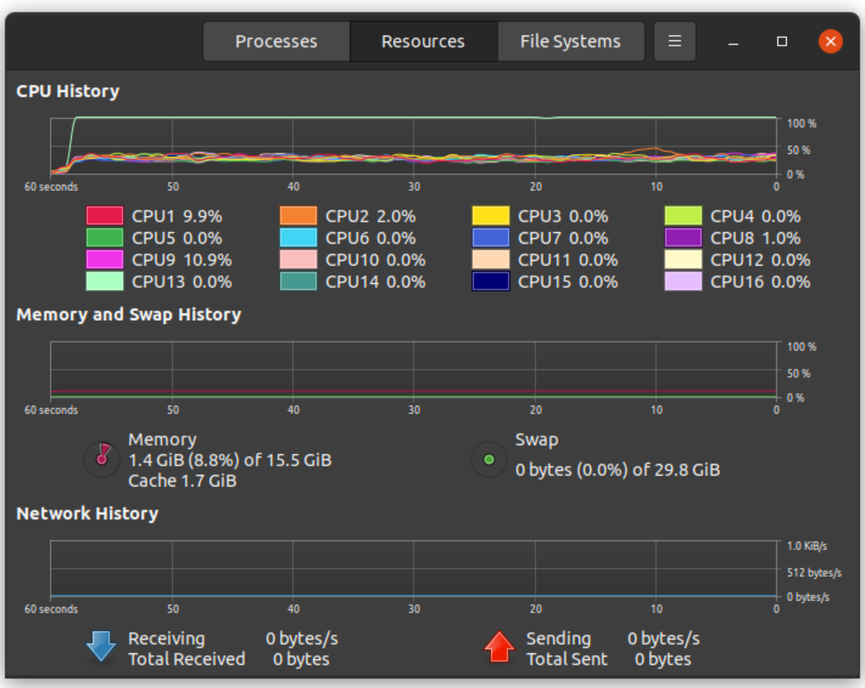




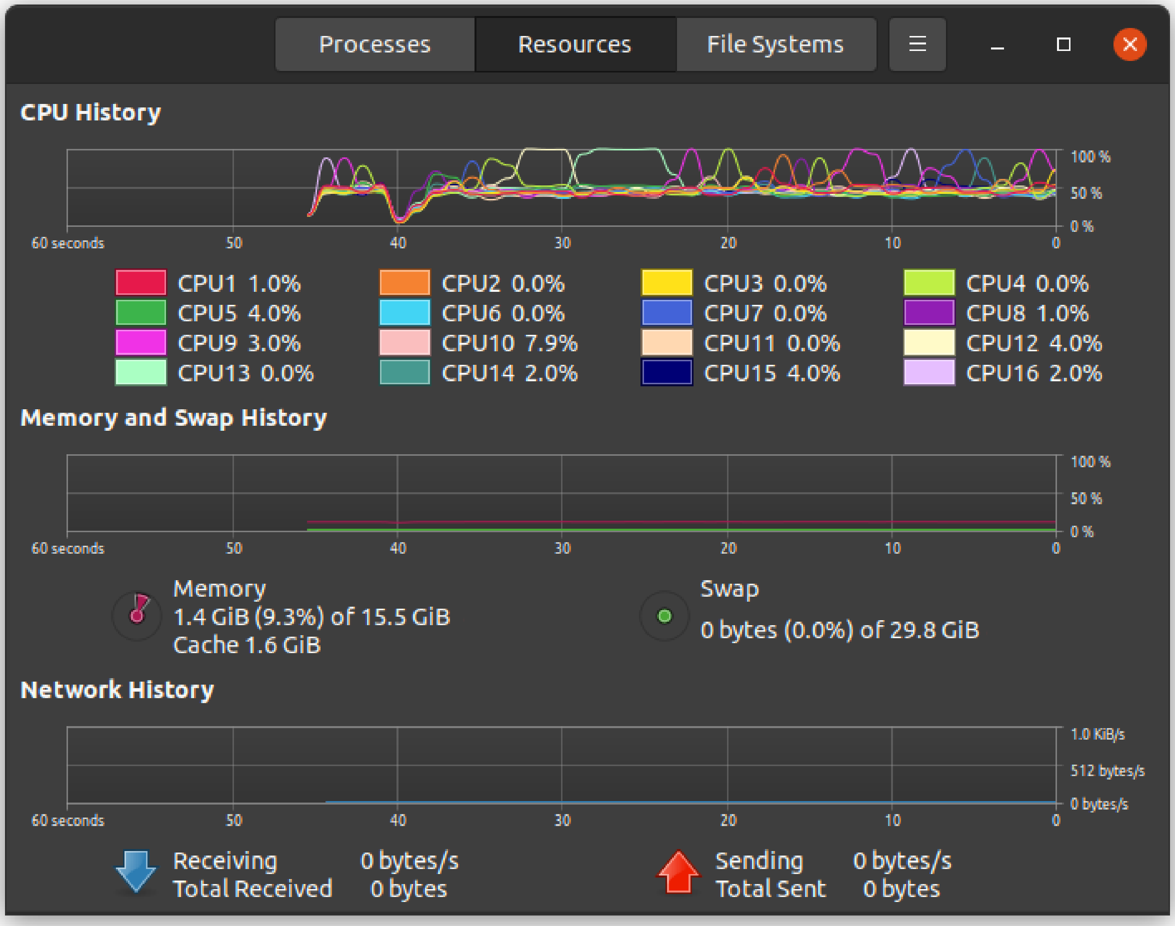
The parameter for this method is no. of threads.In this Method we give one frame to each thread and all threads are running parallely due to which our runtime decrease.As no frame was skipped and no change in frame ,there is zero loss of information .

4 CPU Usage

For a Single-threaded code, the CPU usage looks as follows:



For a multi-threaded code, the CPU usage looks as follows:



5 Conclusion

By Above runtime vs error trade-offs,We can conclude that Temporal Threading is the best method as there is no loss of information and no synchronization of threads. reduce resolution and subsampling methods are also good at certain parameter.To get best result we can use both temporal threading and subsampling .