ECE1783H Assignment 2 Report

Shengxiang Ji - 1002451232

Liangjing Xie - 1009684619

Weiyu Zhang - 1009736706

Part 1. Multiple Reference Frames

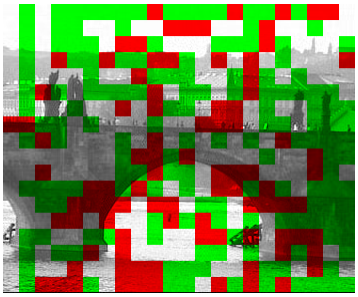
1.1 Visualization

We conducted an experiment focusing on multiple reference frame support with the parameters set as follows: i=16, QP=4, nRefFrames=4, I_Period=10. The following are the first ten frames of the synthetic.yuv file. According to our configuration, the first frame is an I-frame, and the subsequent nine frames are P-frames.

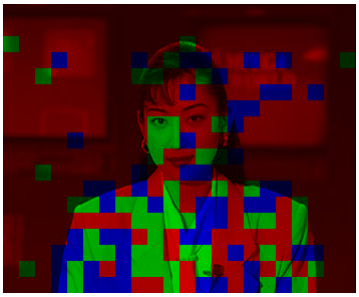
1

2

3



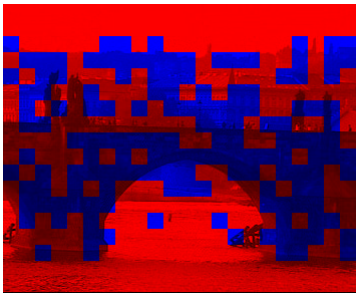
4



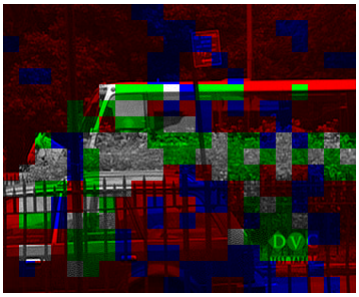
5

6

7



8



9

10

Blocks that are not marked indicate a reference frame index value of 0 (i.e. the previous frame) or that the block is located in an I-frame. Blocks marked in red indicate a reference frame index value of 1 (i.e. two frames before), while blocks marked in green indicate a reference frame index value of 2 (i.e. three frames before). Blocks marked in blue indicate a reference frame index value of 3 (i.e. four frames before).

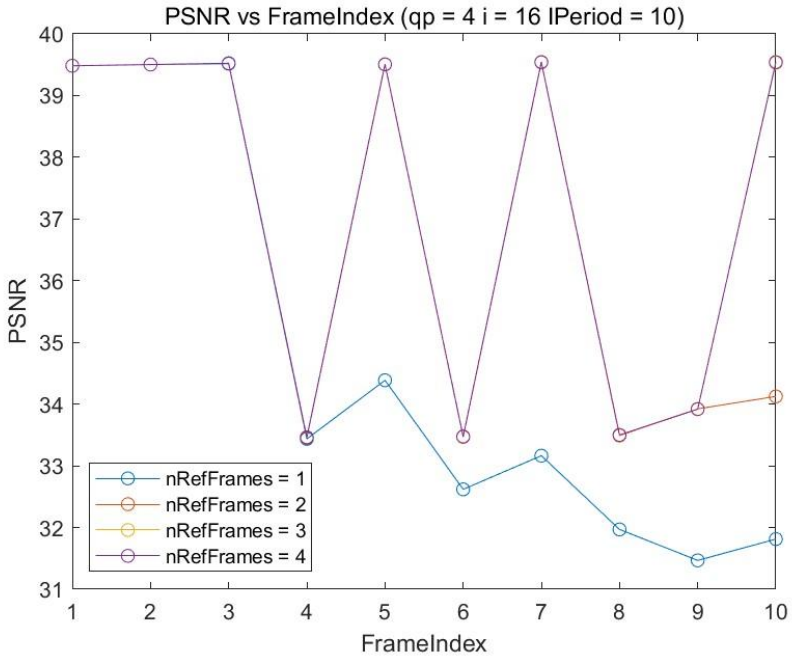
Now, let's analyze the visualization results frame by frame:

- **Frame 1**: I-frame.
- **Frame 2**: Following the I-frame, the reference frame index has a maximum value of 0, so all blocks on Frame 2 have a reference frame index of 0.
- **Frame 3**: The reference frame index can be either 0 or 1. Due to the loss of information in reconstructed frames used as reference frames compared to the original frame, blocks with richer details take a reference frame index of 1 (marked in red), while less detailed blocks take a reference frame index of 0 (not color-marked).
- **Frame 4**: The reference frame index can be 0, 1, or 2. Due to a sudden change in the video on Frame 4, blocks of three colors are evenly distributed within the frame.
- **Frame 5**: The reference frame index can be 0, 1, 2, or 3. Frame 5 is more similar to Frames 1, 2, and 3 than to Frame 4, so all blocks on Frame 5 do not choose a reference frame index value of 0.
- **Frame 6**: The reference frame index can be 0, 1, 2, or 3. However, only Frame 4 has the highest similarity to Frame 6 among all previous frames, so all blocks on Frame 6 choose Frame 4 as the reference frame.
- **Frame 7**: The reference frame index can be 0, 1, 2, or 3. Among the available reference frames for Frame 7, Frame 3 has high similarity with Frames 5 and 7, while Frame 4 has low similarity. Therefore, blocks on Frame 7 have reference frame index values of 1 and 3.
- **Frame 8**: Similar to Frame 7, blocks on Frame 8 have reference frame index values of 1 and 3.
- **Frame 9**: Similar to Frame 4, blocks on Frame 9 have reference frame index values of 0, 1, 2, and 3, with blocks of different reference frame index values evenly distributed within the frame.
- **Frame 10**: The reference frame index can be 0, 1, 2, or 3. However, only Frame 7 has the highest similarity to Frame 10 among all previous frames, so all blocks on Frame 10 choose Frame 7 as the reference frame.

1.2 Plots

In this section, we conducted two sets of experiments to investigate the impact of different values of nRefFrames on frame distortion and BitCount. In the experiment on frame distortion, to eliminate the influence of I-frames on the reference frame index values, we initially set the value of I_Period to 10. Subsequently, to verify the impact of I-frames on the reference frame index values, we repeated the experiments by setting the value of I_Period to 3.

1.2.1 PSNR vs FrameIndex



The initial three frames of the original video show minimal differences, hence the impact of nRefFrames on frame distortion is not pronounced.

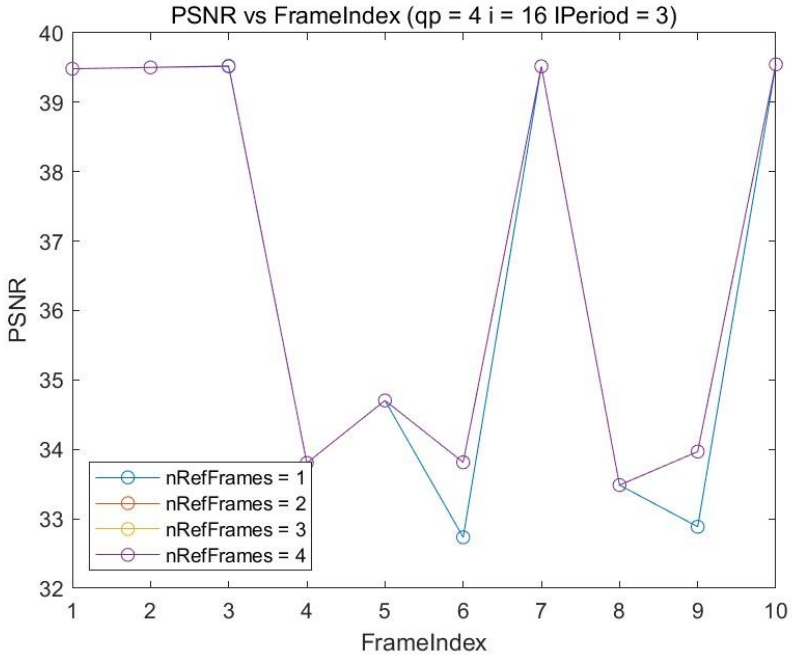
The original video undergoes a significant change starting from the fourth frame, causing poor PSNR for the fourth frame in frame interprediction as there is no suitable reference frame. Therefore, regardless of the value of nRefFrames, the PSNR for the fourth frame is consistently low.

Starting from the fifth frame, the value of nRefFrames begins to noticeably affect the PSNR. Specifically, the image quality for the fifth frame is significantly worse when nRefFrames is set to 1. This is because when nRefFrames is 1, the fifth frame can only choose the fourth frame, which has a significant difference, as the reference frame for frame interprediction. When nRefFrames is greater than 1, it can choose frames located further back as references.

The sixth frame has the highest similarity to the fourth frame. When nRefFrames is not equal to 1, the reference frame index takes the value 1; otherwise, it takes the value 0. While the PSNR is higher when the reference frame index is 1, indicating a better image quality, the distortion is still relatively high. This is due to the lack of a suitable reference frame for the fourth frame in the frame interprediction process, resulting in high distortion for the reconstructed fourth frame. Consequently, using the fourth frame as a reference frame for the sixth frame leads to similarly high distortion.

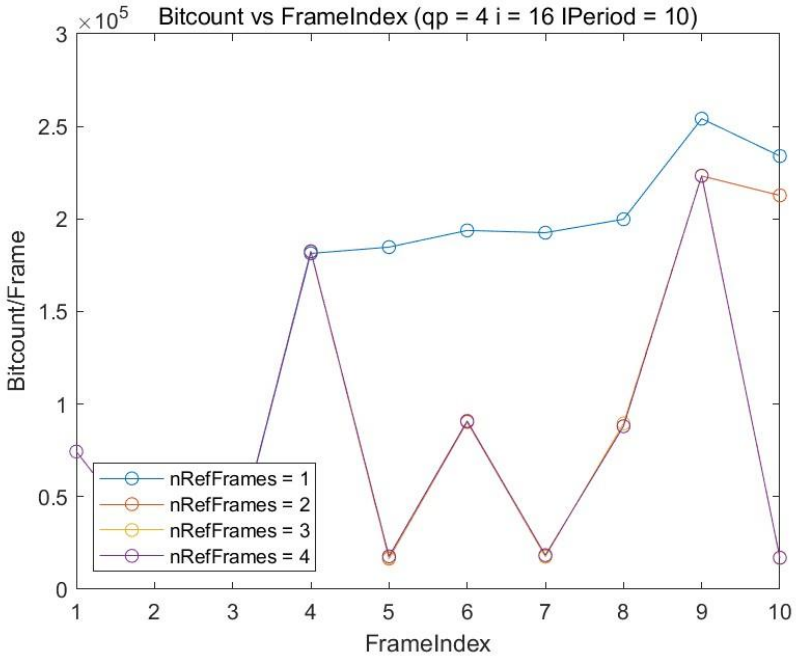
For the tenth frame, when nRefFrames takes values of 1, 2, and 4, there is a significant difference in frame distortion. When nRefFrames is 1, all blocks on the tenth frame can only use the ninth frame as a reference, which has a large difference. When nRefFrames is 2, blocks on the tenth frame can choose either the eighth or ninth frame as a reference, resulting in a higher PSNR compared to when nRefFrames is 1. When nRefFrames is 3 or 4, blocks on the tenth frame will choose the seventh frame as a reference, and since the seventh frame is highly similar to the tenth frame, the frame quality is the highest.

To validate the influence of I-frames on reference frame index values and verify the above analysis, we repeated the experiments with I_Period set to 3 and plotted the following graph:



Compared to the case where I_Period is set to 10, when nRefFrames is greater than 1, the PSNR values for the fifth and eighth frames remain the same as when nRefFrames is equal to 1. This is because, at this point, the fourth and seventh frames are I-frames, and the reference frame index for the fifth and eighth frames can only take a maximum value of 0. The tenth frame also becomes an I-frame at this point, resulting in the same performance for frame distortion regardless of the value of nRefFrames.

1.2.2 BitCount vs FrameIndex

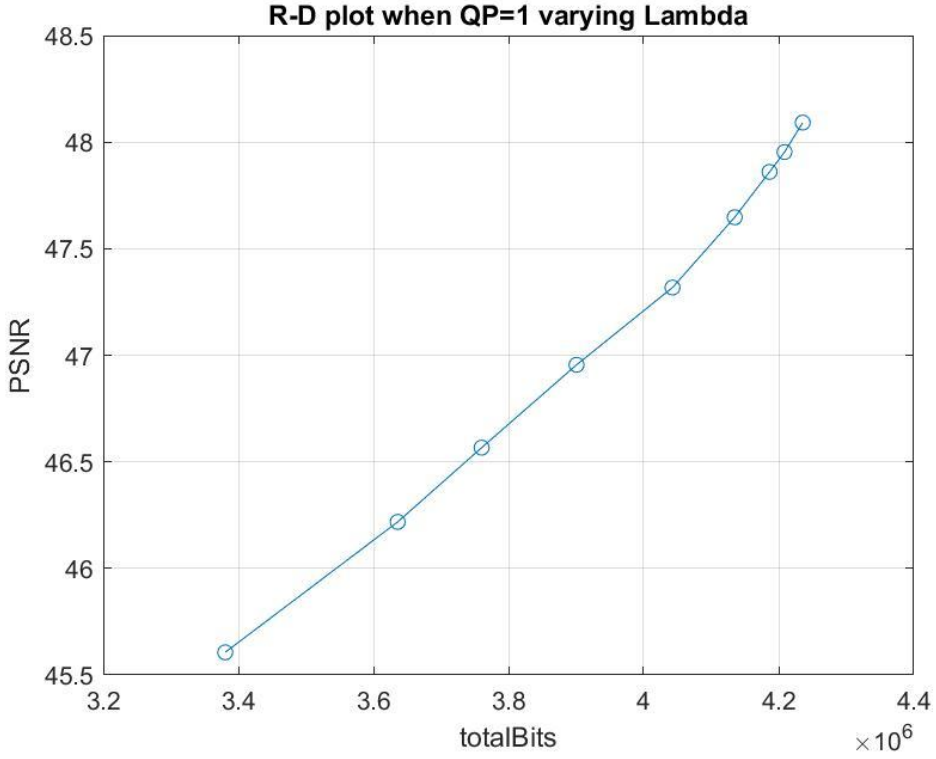


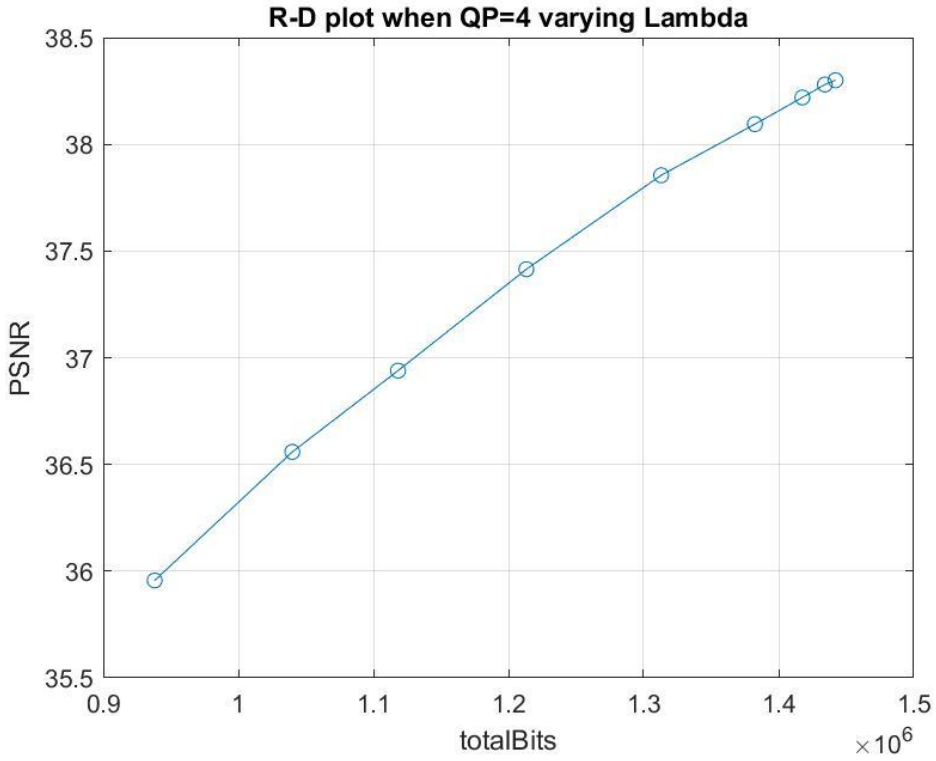
It is evident that an increase in the value of nRefFrames correlates with a reduction in Bitcount. This outcome stems from the selection of more appropriate reference frames, which results in smaller values within the residual frame and, consequently, more efficient compression of the encoded file.

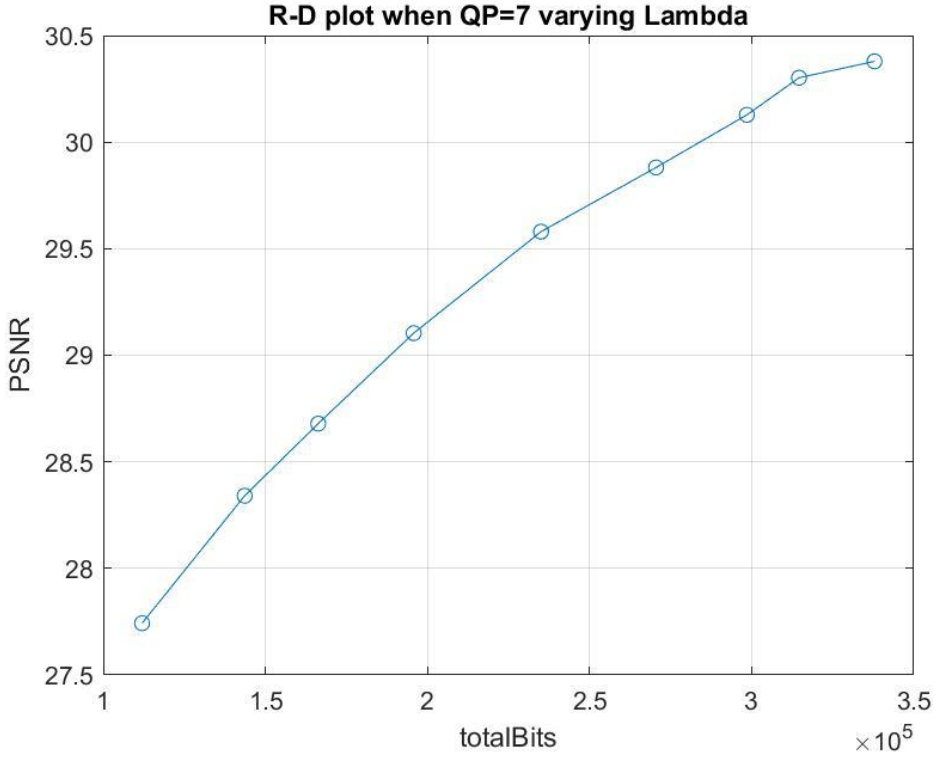
Additionally, noteworthy is the observation that instances of abrupt changes in the frame sequence coincide with a substantial increase in Bitcount (frames 4, 6, 9). This further underscores the efficacy of opting for suitable reference frames in minimizing Bitcount. The results make it apparent how the application of Multiple Reference Frames search enhances Motion Estimation in our experiments.

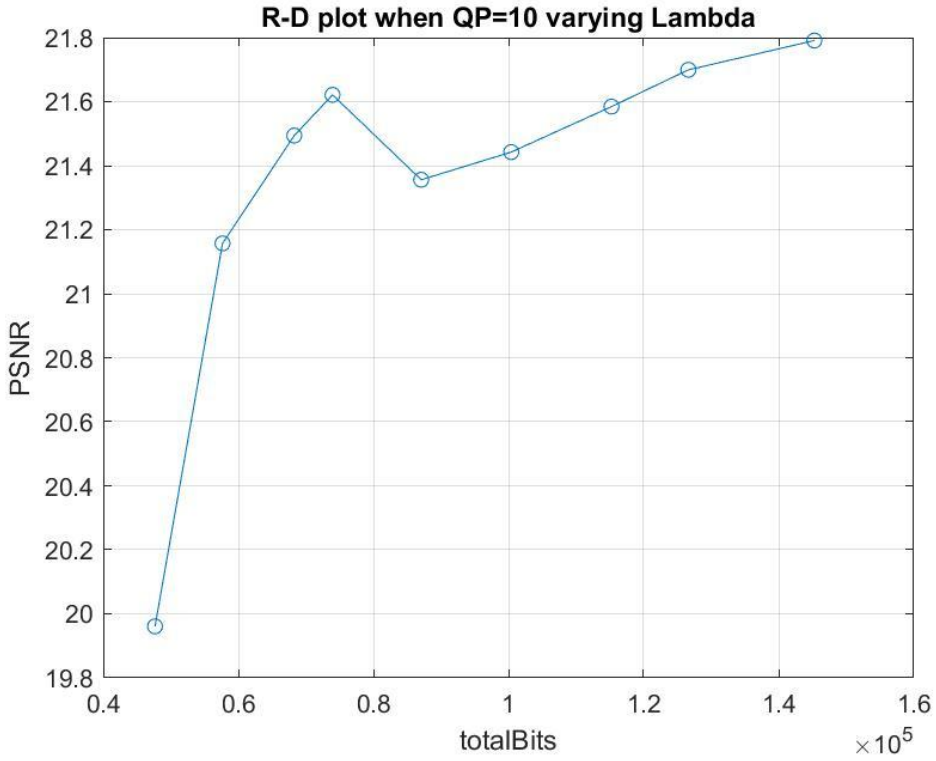
Part 2. Variable Block Size

The variable block size feature allows a large block to be split into four smaller blocks, possibly making the residual values smaller, hence improving the quality of the video. To make the decision on whether to split a block, the R-D cost will be evaluated for both the options to split or not split the block. The R-D cost J is defined to be $J=D+\lambda R$, where D is the SAD of the predicted block, R is the total bits required to encode the residuals and the MVs (for P frames) or modes (for I frames). λ is the Lagrange Parameter, and a set of experiments are performed to find its value based on QP.


R-D plot when QP=1 varying Lambda
PSNR
totalBits
×10^6
48.5
48
47.5
47
46.5
46
45.5
3.2
3.4
3.6
3.8
4
4.2
4.4


R-D plot when QP=4 varying Lambda
PSNR
totalBits
×10^6
38.5
38
37.5
37
36.5
36
35.5
0.9
1
1.1
1.2
1.3
1.4
1.5



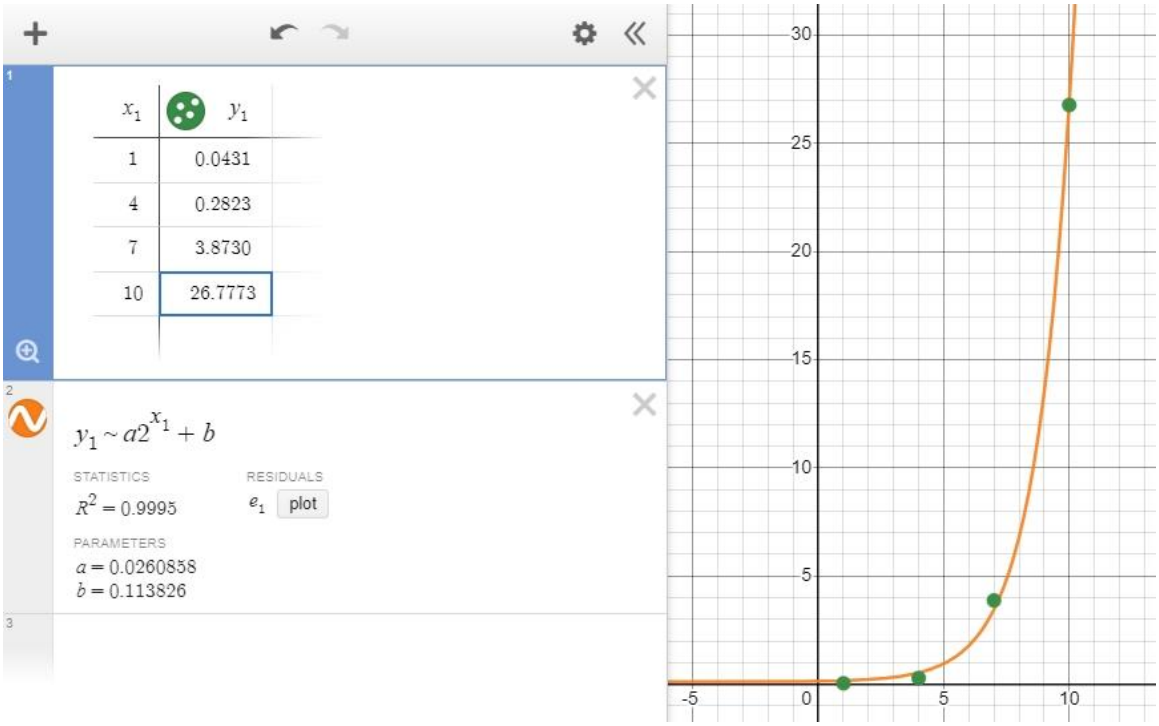


As shown in the plots above, the R-D plots are shown for QP values 1, 4, 7 and 10, and each point in the plots represents a different lambda value. The top-right point represents setting λ=0, meaning the encoding will split the block whenever the SAD after splitting is lower. The

bottom-left point represents disabling the variable block size feature, meaning no block is ever split. The optimal lambda for each QP value is selected subjectively by looking at the R-D plots, where the points closer to the top-left corner are selected. The optimal lambda values for the QP values are:

QP	Optimal Lambda
1	0.0431
4	0.2823
7	3.8730
10	26.7773

For QP values that are not explicitly tested, we use exponential regression methods to approximate the best lambda value. The optimal lambda value is approximated by λ = 0.0261 * (2^QP) + 0.1138, as shown in the regression plot below.

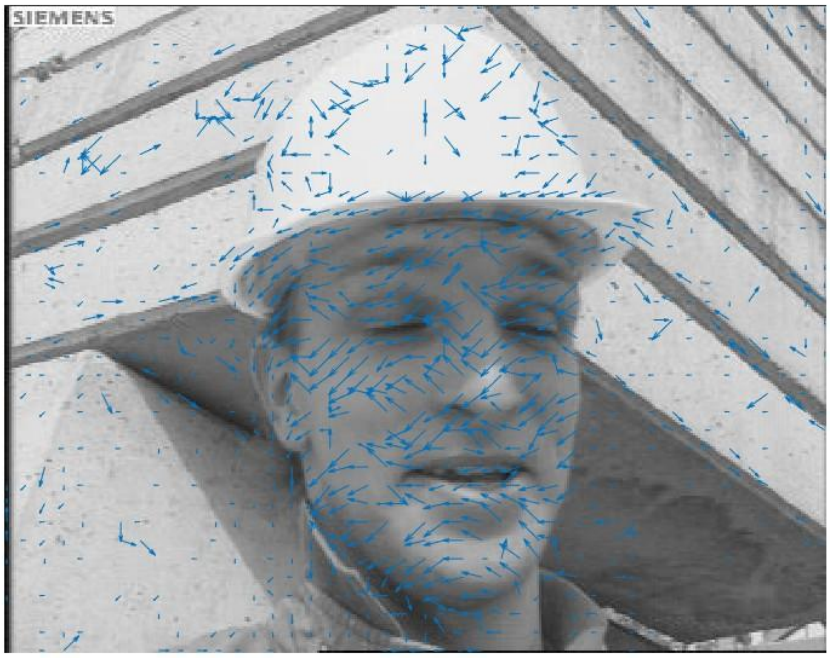


2.1 Visualization

The following are the first four frames of the foreman420.yuv under the parameter settings: blockSize = 16; r = 4; QP = 4; I_Period = 10; nRefFrames = 4; VBSEnable = true; FMEEnable = false; FastME = false. In the I-frame (first frame), horizontal arrows indicate intra-prediction mode as horizontal, vertical arrows indicate intra-prediction mode as vertical, and the arrow length represents the current block size. In the P-frames (frames 2, 3, 4), arrows represent motion

vectors (MV).

1



2

3

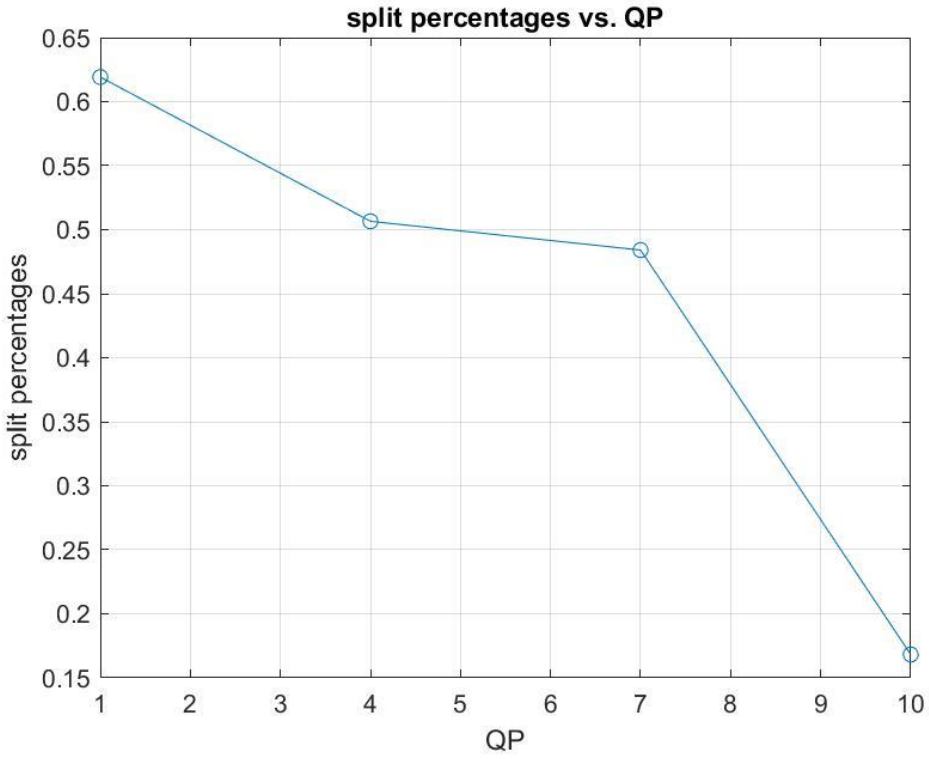
4

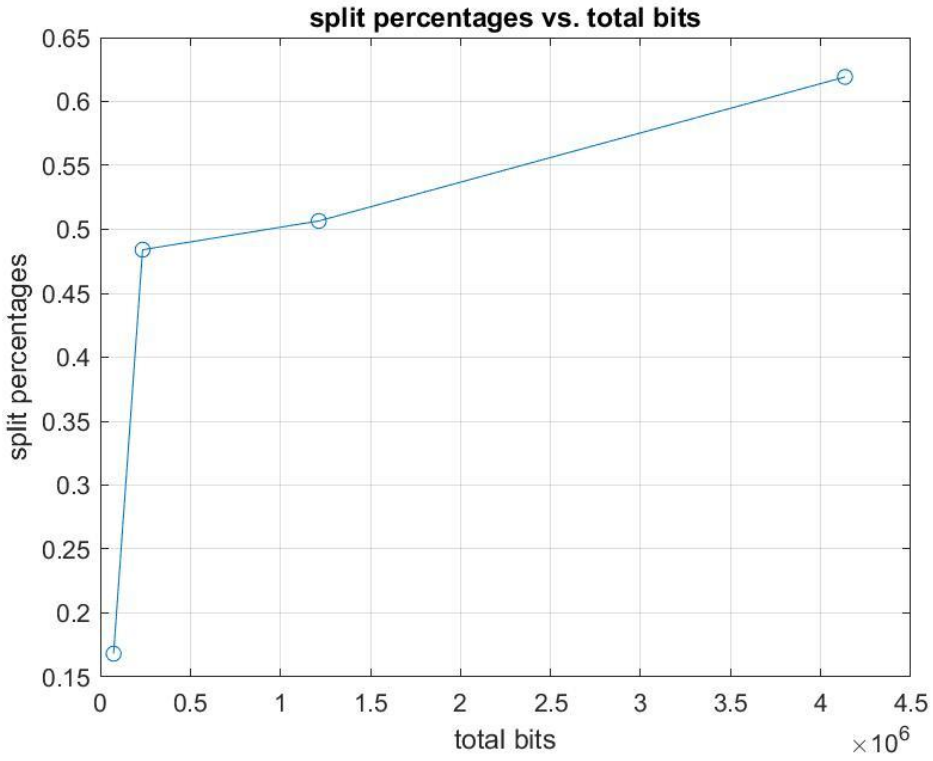
In the images, it is evident that more blocks are split in regions with rich image details. In areas featuring moving characters, the MV vectors have longer

lengths, while the background areas exhibit shorter MV vector lengths.

2.2 Plots

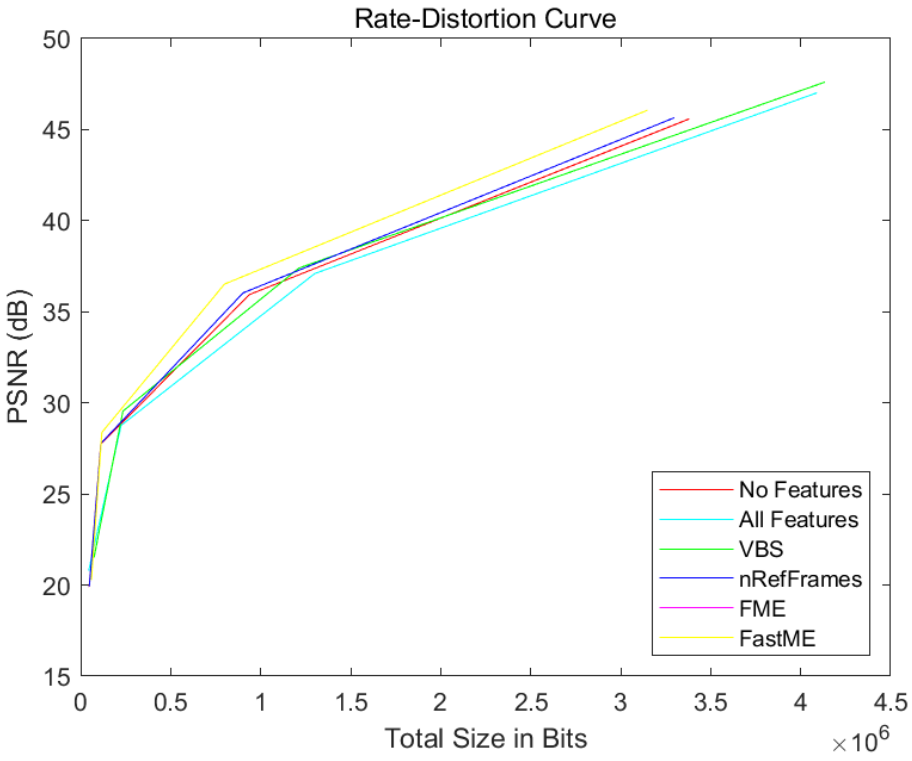
The percentage of blocks that were chosen to be split into sub-blocks is reported in the plots below. For QP values of 1, 4, 7, and 10, the optimal lambda values are used as reported in the previous section. As we can see from the plots, as QP increases, the percentage of blocks that were chosen to be split decreases. On the other hand, as the total number of bits increases, the percentage of blocks that were chosen to be split increases. Splitting the blocks can help increase the quality of the encoder, at the cost of a higher bit rate. Therefore at high bit rates, splitting can help with saving bits in residuals, therefore more blocks are split. When the bit rate is low, lots of details are lost, splitting does not help with the overall quality but increases the bit rate, therefore fewer blocks are split.

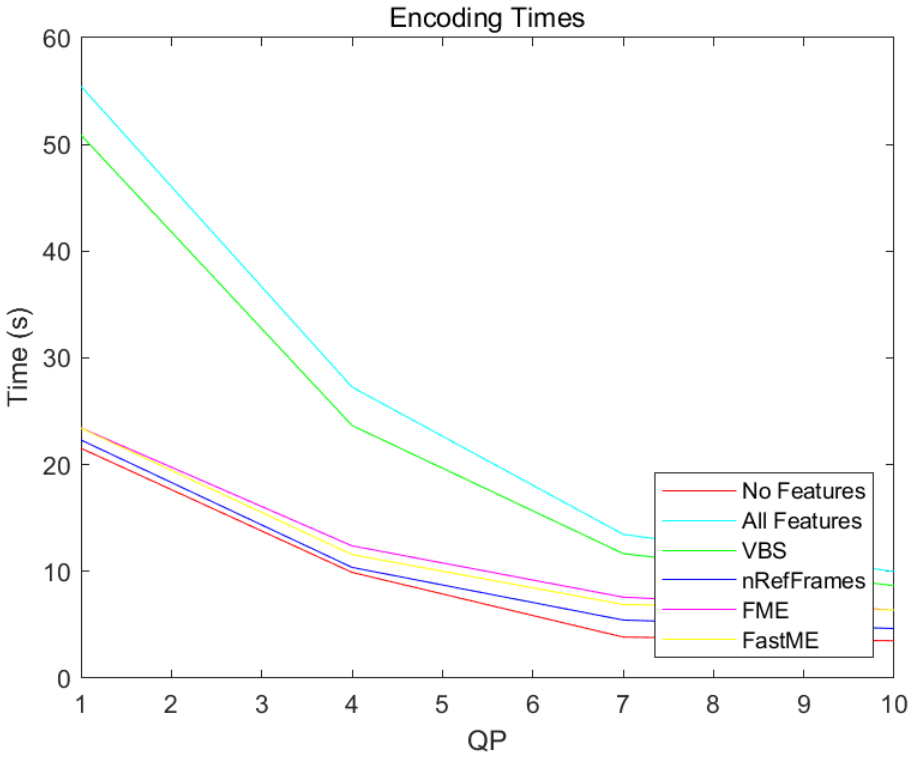




Part 3. Overall RD plots and Execution Times

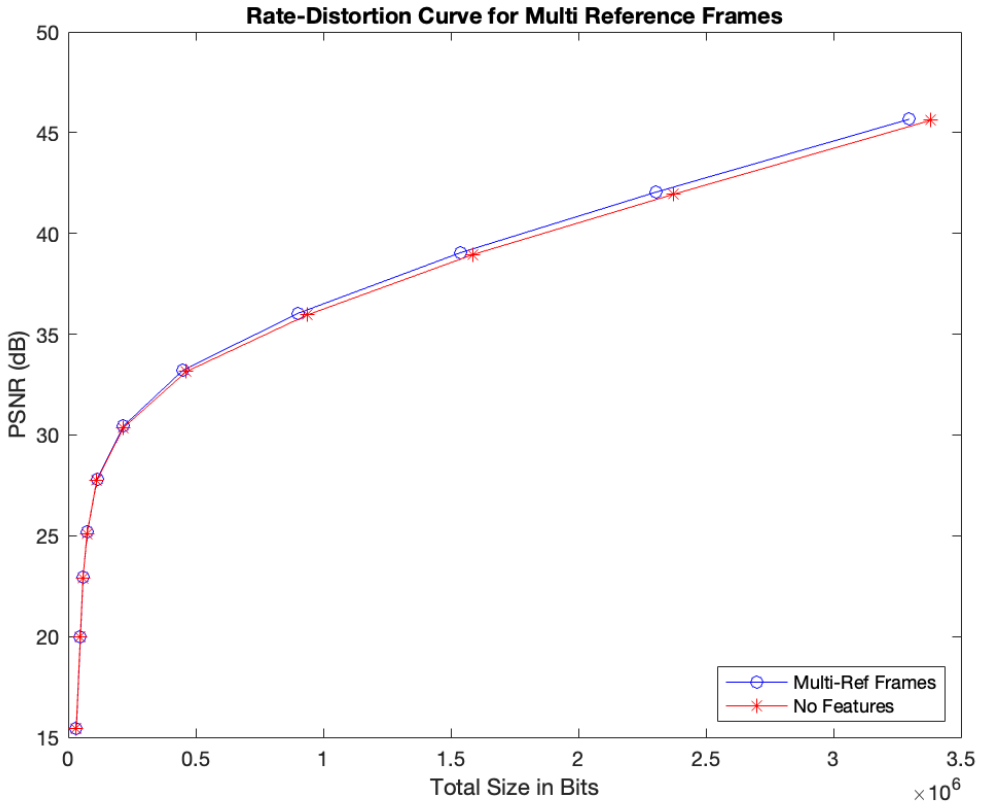
In this section, we analyze the impact of each feature by contrasting the Rate-Distortion plots generated with the feature enabled versus disabled, under a fixed set of parameters (block size = 16, search range = 4, I_Period = 8). We include encoding times and decoding times as well in the comparisons. We use the first 10 frames of Foreman sequence for testing.





Above are the results obtained when we take QP=1,4,7,10 to run the experiments. In the following, we will compare and analyze the execution time and the result of the four features.

3.1 Multiple Reference Frames

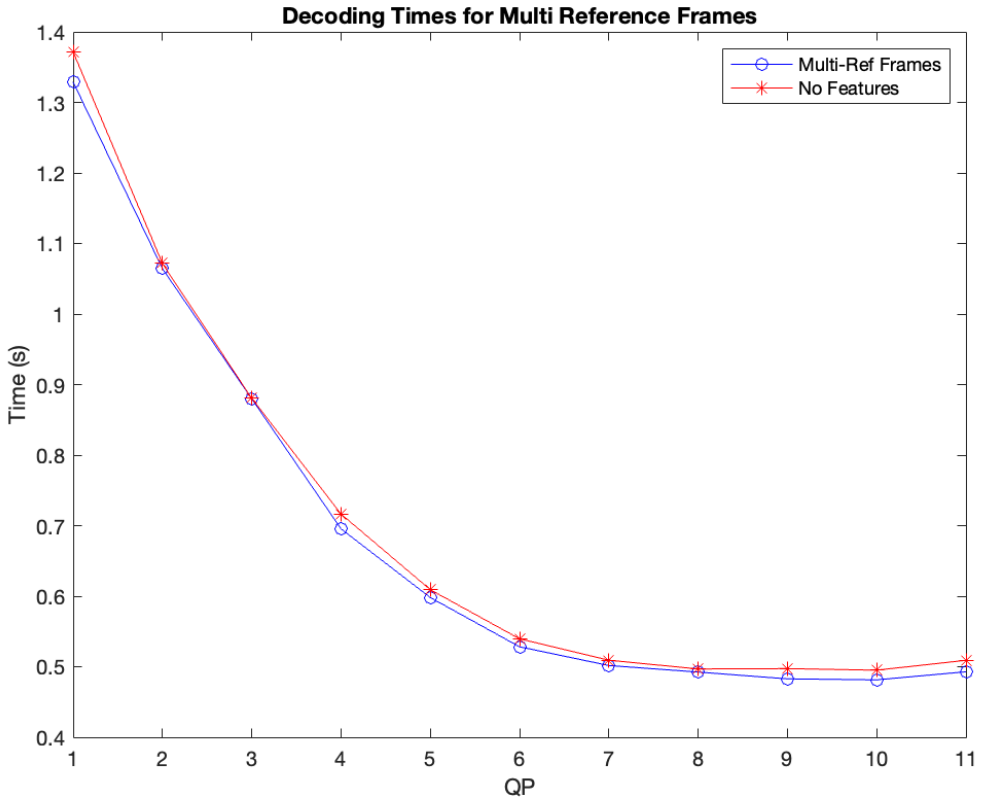


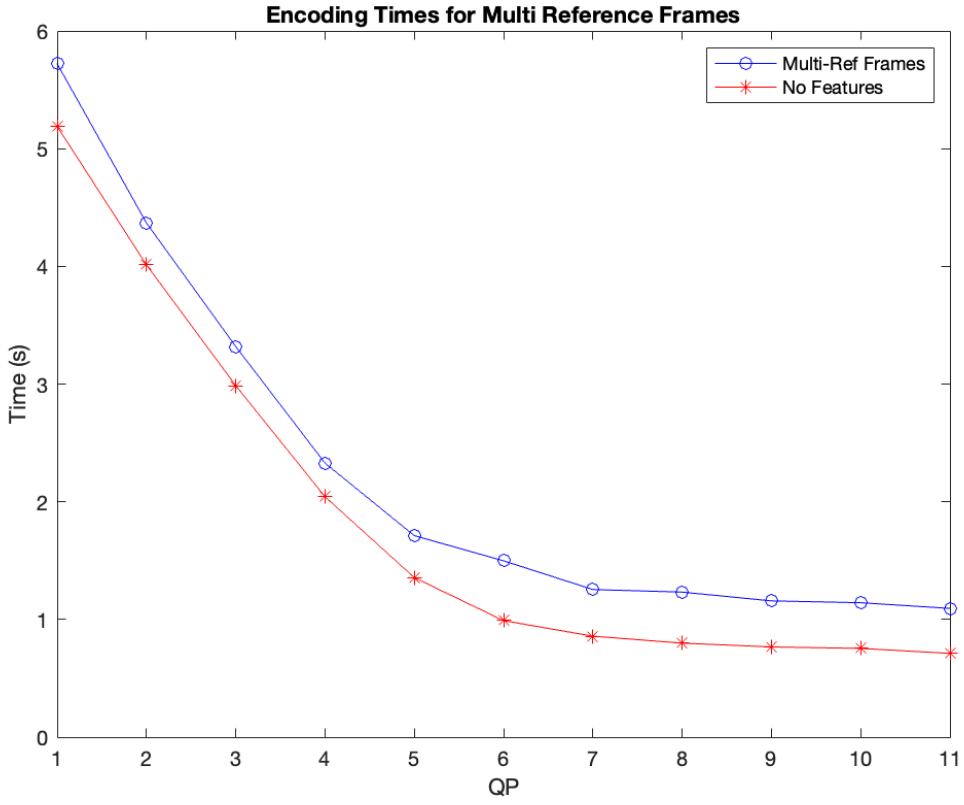
In the above comparison, we configured the number of reference frames to 4 to activate the feature, and set it to 1 for the baseline scenario without the feature.

For the same total size, the use of multiple reference frames seems to consistently yield a higher PSNR compared to when no features are used. This indicates that enabling multiple reference frames helps to preserve video quality at a given bitrate or, conversely, to achieve a certain level of quality with fewer bits.

At lower bitrates, the two curves appear to converge, implying that the advantage of multiple reference frames diminishes as the bitrate decreases. This is because at lower bitrates, the amount of data available to represent each frame is reduced.

The difference between the two curves becomes higher when total size is bigger, suggesting that the benefit of using multiple reference frames is more significant when the bitrate is

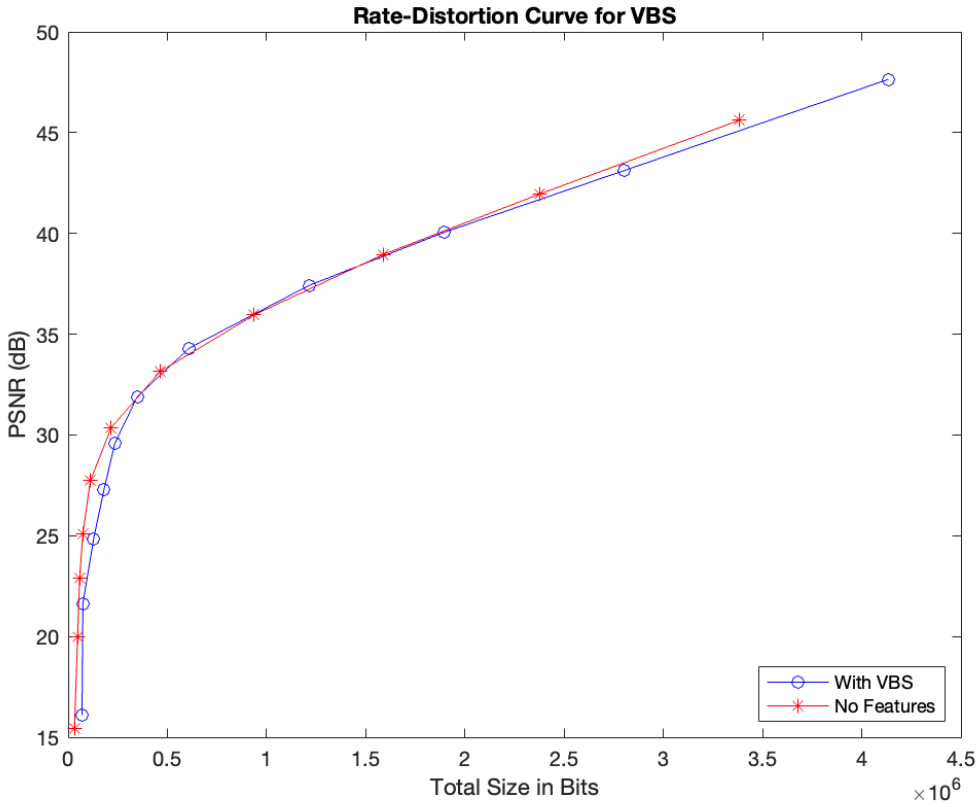

Decoding Times for Multi Reference Frames
Multi-Ref Frames
No Features
Time (s)
QP
1.4
1.3
1.2
1.1
1
0.9
0.8
0.7
0.6
0.5
0.4
1
2
3
4
5
6
7
8
9
10
11


Encoding Times for Multi Reference Frames
Multi-Ref Frames
No Features
Time (s)
QP
6
5
4
3
2
1
0
1
2
3
4
5
6
7
8
9
10
11

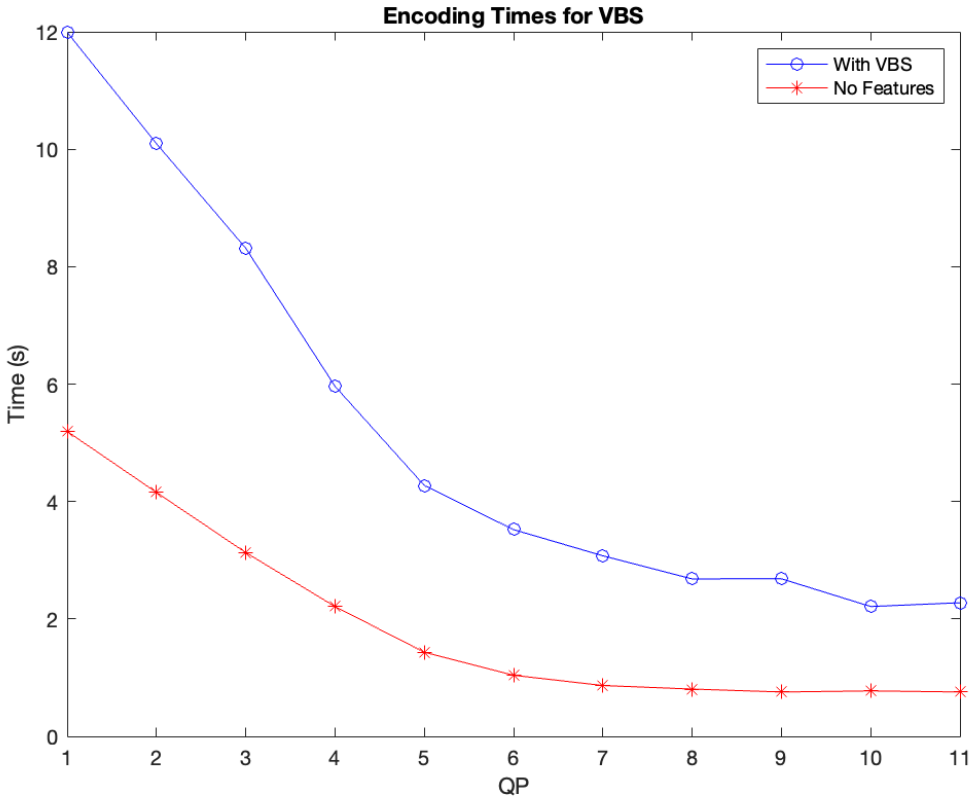
sufficient.

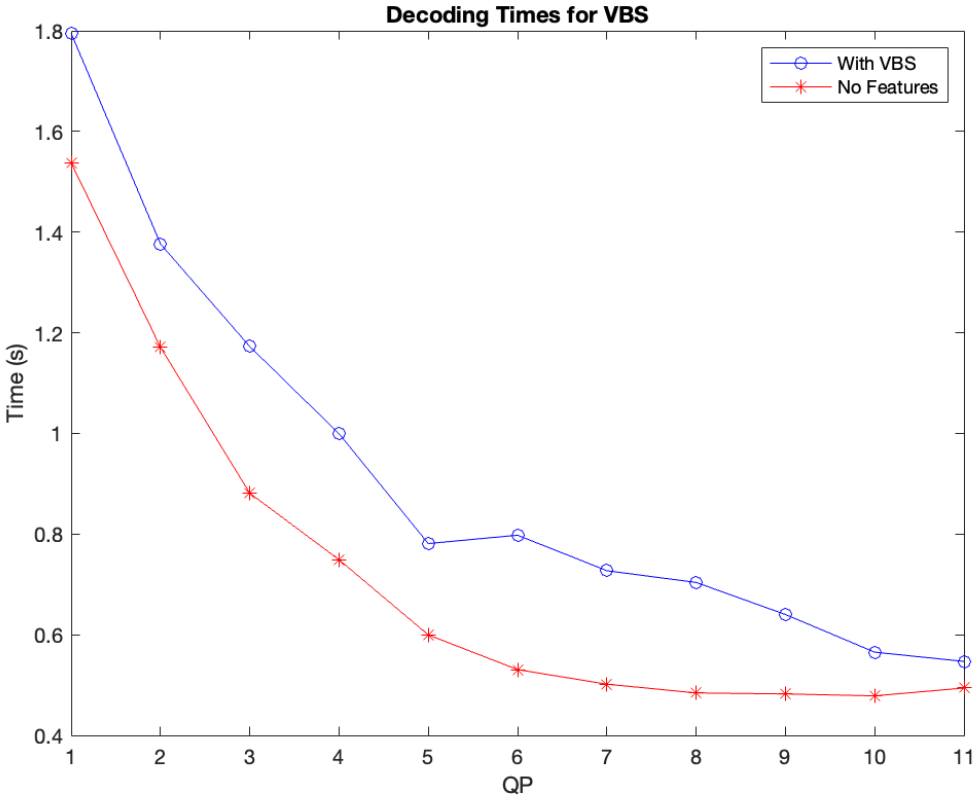
From the two figures, we can observe the following:
For the encoding times and decoding times, both with multi-reference frames enabled and with no features, the execution time decreases as the QP increases. This is expected since higher QP results in more compression, thus less data to process, leading to faster decoding.
The decoding time with multi-reference frames enabled is very close to the time with no features, suggesting that the impact of this feature on decoding time is minimal.
For the encoding times, there is a bigger difference between having multi-reference frames enabled and disabled. The encoding time is consistently higher when multi-reference frames are used at all QPs. This indicates that while multi-reference frames may improve the rate-distortion performance, they also increase the complexity of the encoding process.

3.2 Variable Block Size



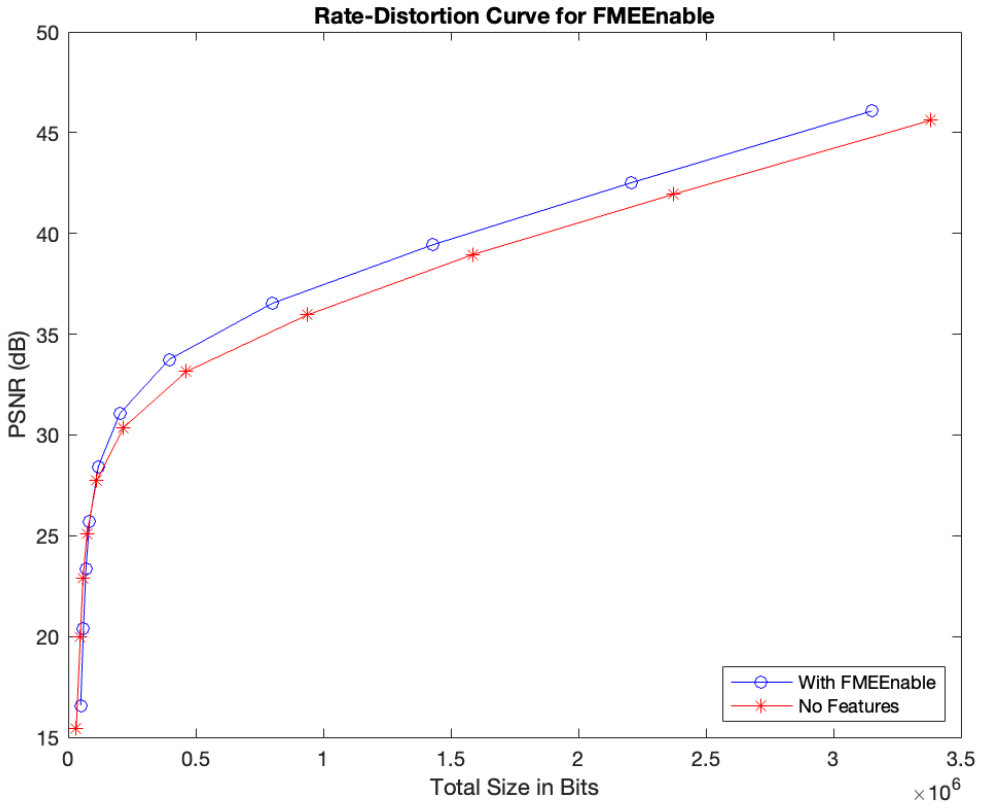
The VBS feature provides a slight advantage as shown in the Rate-Distortion curve being higher with VBS enabled compared to without, suggesting that enabling VBS improves the PSNR for a given bitrate.


Encoding Times for VBS
With VBS
No Features
Time (s)
QP
0
2
4
6
8
10
12
1
2
3
4
5
6
7
8
9
10
11


Decoding Times for VBS
With VBS
No Features
Time (s)
QP
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
1
2
3
4
5
6
7
8
9
10
11

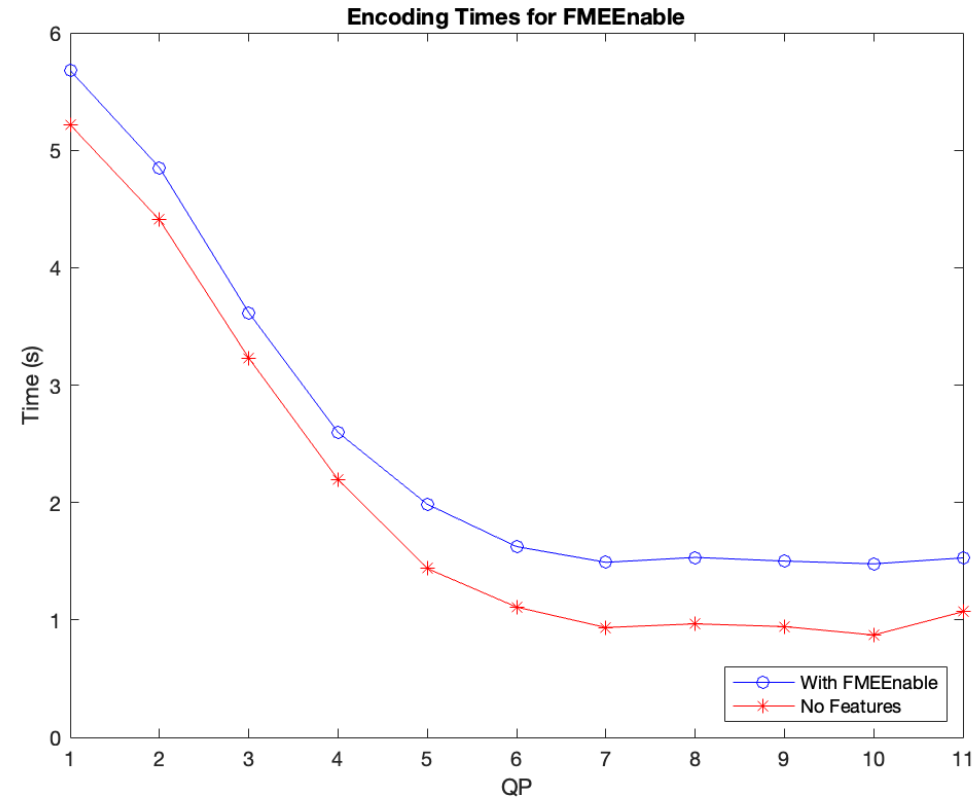
From the above two figures we can see that enabling variable block size (VBS) increases both encoding and decoding time.
In encoding, VBS requires evaluating whether to split each block into smaller ones for better motion compensation, involving additional computations for motion estimation, quantization, and encoding for each sub-block. In decoding, enabling VBS needs additional steps to interpret variable-sized blocks and reconstruct the frame from these segments. These added computational requirements for handling variable-sized blocks lead to increased processing times during both encoding and decoding.

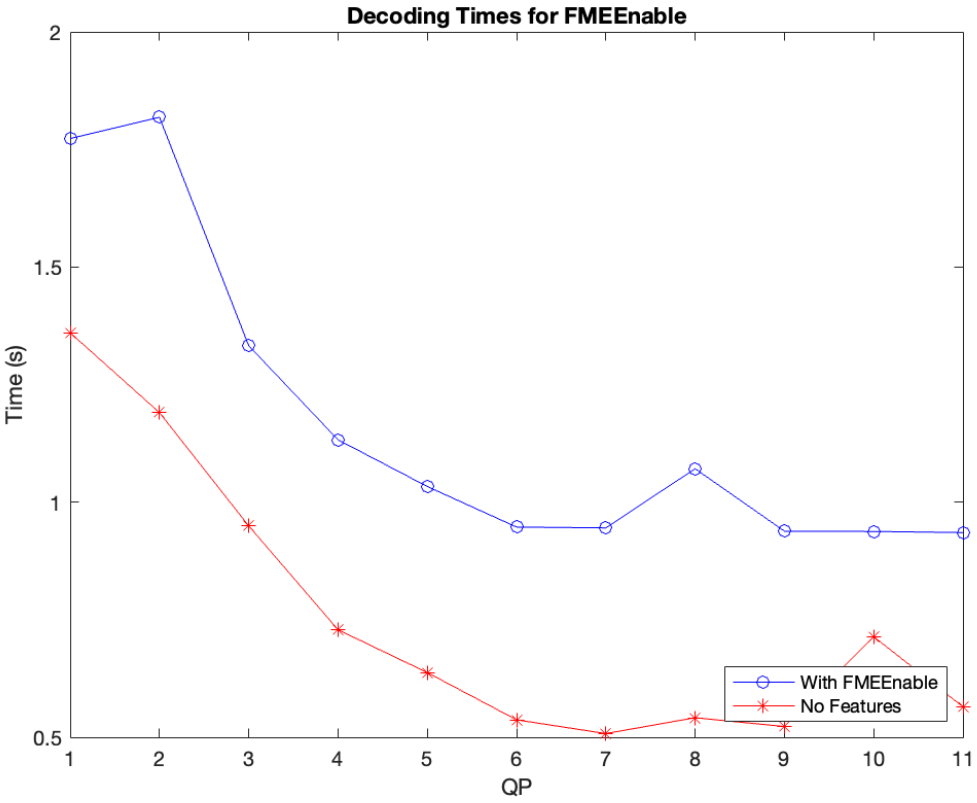
3.3 Fractional Motion Estimation



From the above figure we can see the FMEEnable feature significantly improves the reconstructed video quality, it makes greater improvement than enabling multiple reference frames and variable block size.

Fractional motion estimation (FME) enhances video quality by enabling sub-pixel precision in motion compensation, allowing for more accurate inter-frame motion predictions through interpolation.





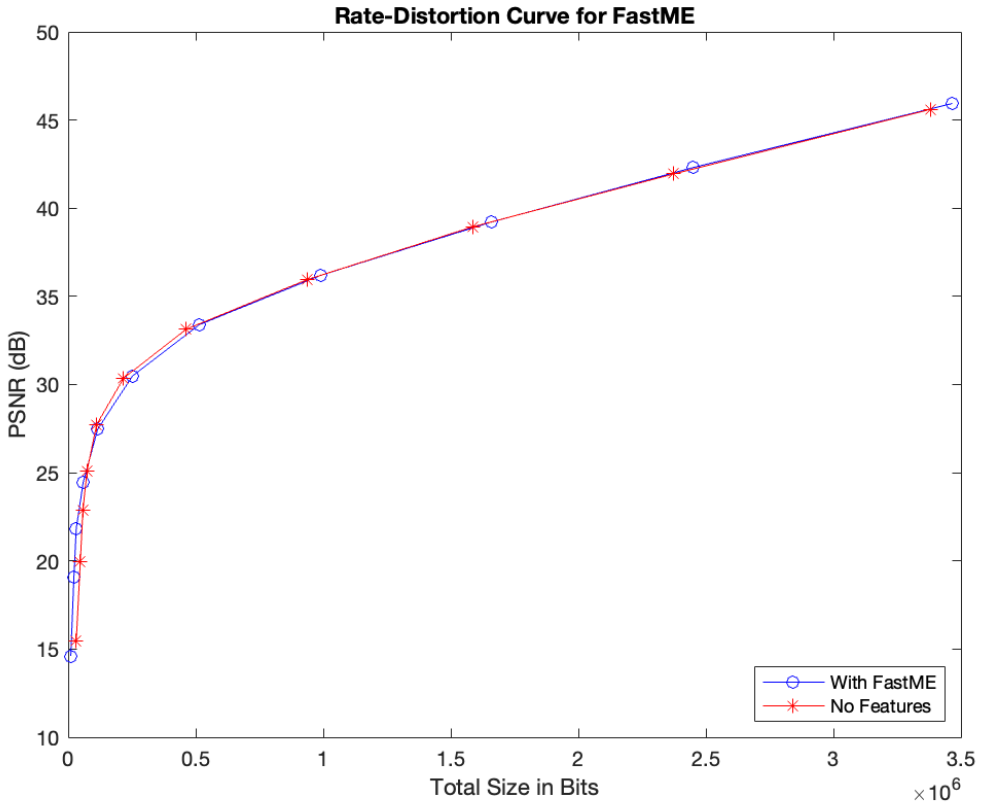
In both figures we can see that with FME enabled, the encoding and decoding time remains

fairly consistent across the range of QPs,suggesting that additional time is invested in the interpolation process to enhance the motion estimation accuracy.

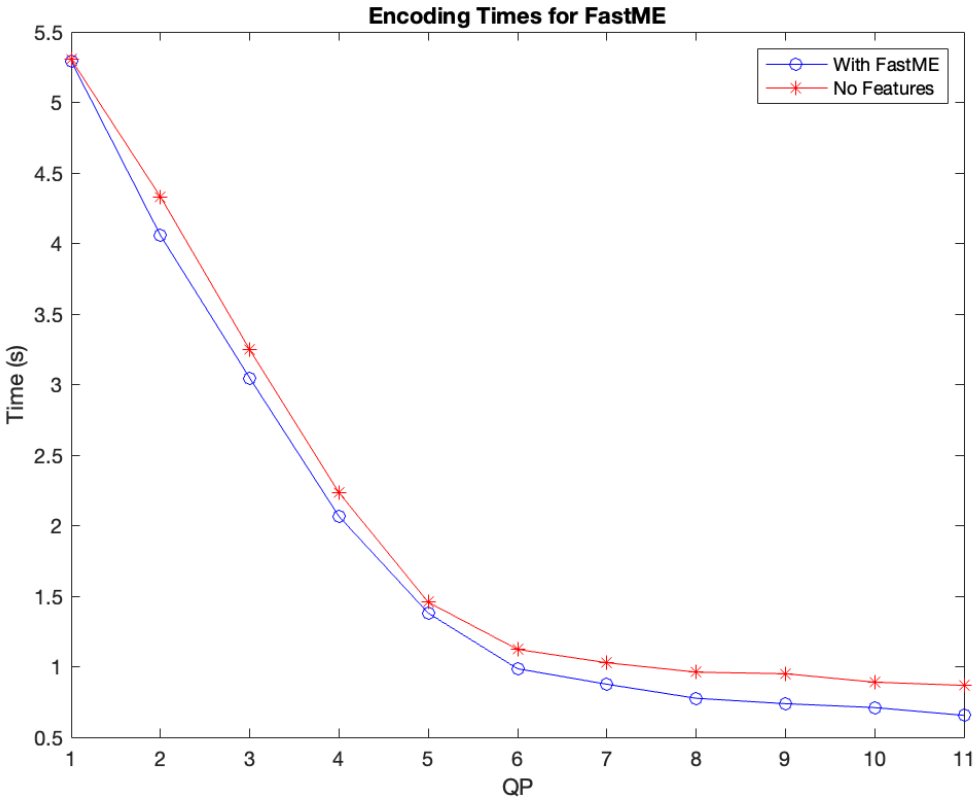
FME increases encoding time because it requires additional computations to find the best match for the fractional pixel positions.

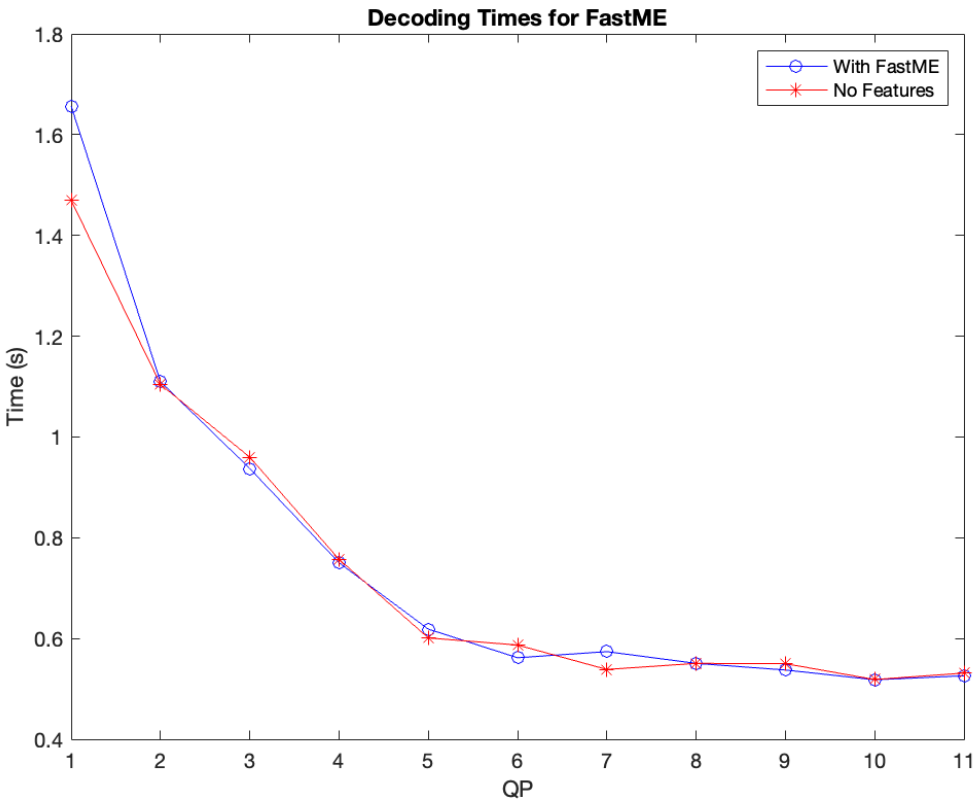
The decoding time is also affected because the decoder must perform the inverse operations to reconstruct the video from the compressed bitstream. If FME has been used in encoding, the decoder must also apply the same sub-pixel interpolation to accurately reconstruct the original frame. This requirement to handle sub-pixel accuracy increases the complexity of the decoding process as well, leading to longer decoding times.

3.4 Fast Motion Estimation



From the above figure we can see the two curves are almost the same. Enabling FastME almost does not affect the quality of the reconstructed sequence.

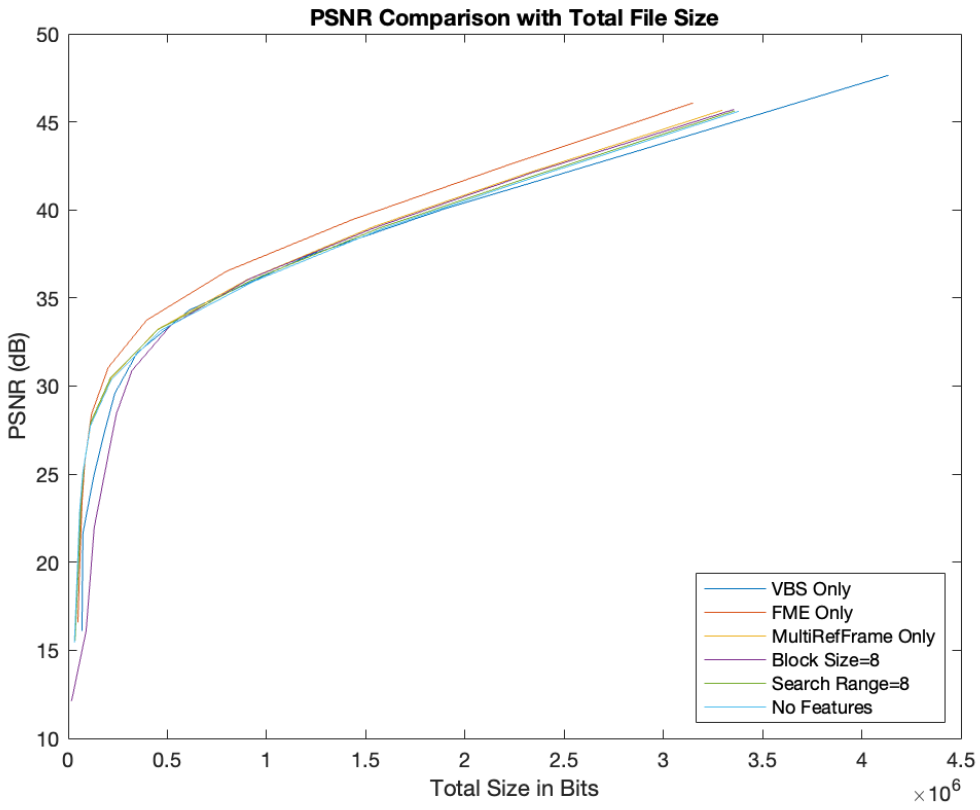




As the QP increases, the time taken for both encoding and decoding generally decreases. This is because higher QP values lead to greater compression and less data to process, which speeds up both encoding and decoding.
The first graph shows a significant decrease in encoding times when FastME is enabled compared to when no features are used. This suggests that FastME greatly accelerates the encoding process, as it reduces the number of blacks to be searched.
Decoding times, on the other hand, are less affected by motion estimation complexity. The decoding process primarily involves reading and reconstructing frames based on the motion vectors and other data encoded into the bitstream. Since FastME doesn't change the amount of data to be decoded (it only affects the selection process of what data to encode), the decoding process doesn't benefit as much from FastME as the encoding process does.

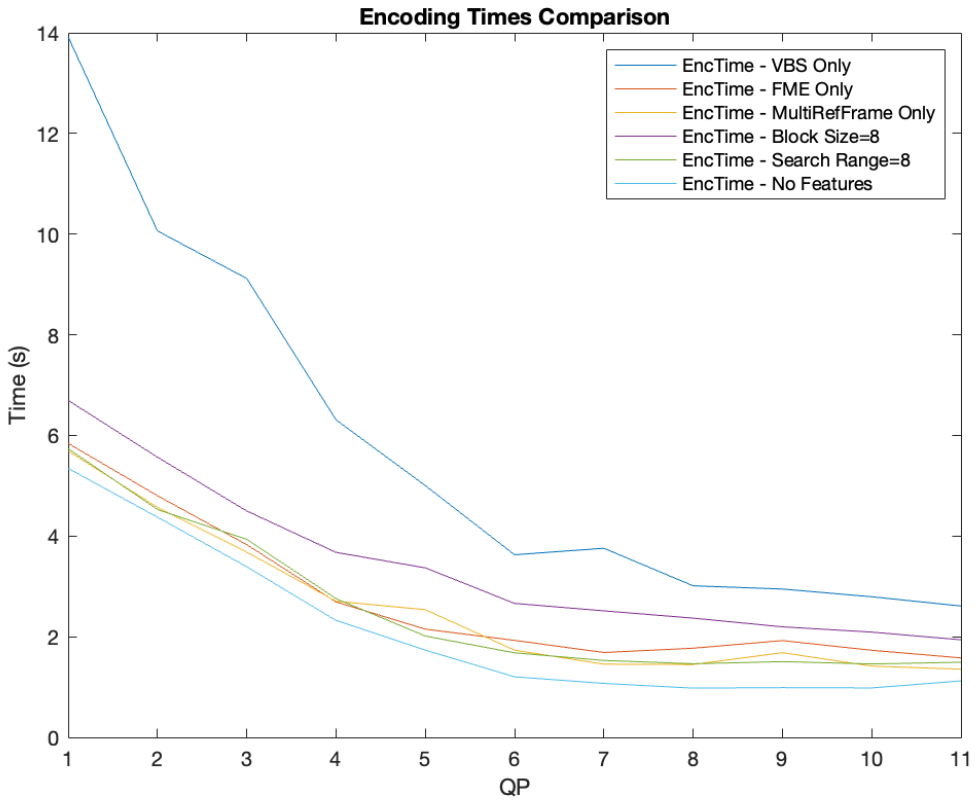
Part 4. Questions

4.1 What effect do Multiple Reference Frames, Variable Block Size, and Fractional Motion Estimation have on encoding speed and quality? How do they compare to using a smaller block size and/or a larger search range?



From the above figure we can see that enabling FME yields the greatest improvement on encoding quality, enabling VBS or multiple reference frames also improves the encoding quality compared to enabling no features, but not as much as FME. FME increases PSNR most because it allows for sub-pixel accuracy in motion compensation, which can result in a more precise representation of the moving objects

Decreasing the block size from 16 to 8 or increasing the search range from 4 to 8 also slightly improves encoding quality, but not as much as enabling multiple reference frames. Smaller blocks can match more detailed motion, and a larger search range can find better matches over a broader area.



These features generally increase encoding time because they add computational complexity to the encoding process. VBS requires additional decisions on block partitioning, FME involves sub-pixel interpolations and more refined searches, and multiple reference frames increase the number of comparisons that need to be made.
This suggests that while those features provide quality improvements, they also require more computation and increases encoding time, so we need to consider the trade-off between these factors.
Smaller block size increases encoding time due to the higher number of blocks that need to be processed and the more complex motion estimations required for each block.
Larger search range also increases encoding time since a larger area must be searched for each block to find the best match, leading to more computations.

4.2 Could the effect of Multiple Reference Frames depend on content type? Can you think about what kind of non-artificial video could benefit the most from it?

The effect of reference frames can indeed depend on the content type, and certain types of videos can benefit more from MRF.

A content type that stands to gain substantial benefits from Multiple Reference Frames is fast repetitive motion. A non-artificial example is a high-speed camera capturing the motion of a hummingbird's wings. In such videos, birds' wings maintain high-speed movements, leading to significant variations between adjacent frames. But two frames, separated by a certain

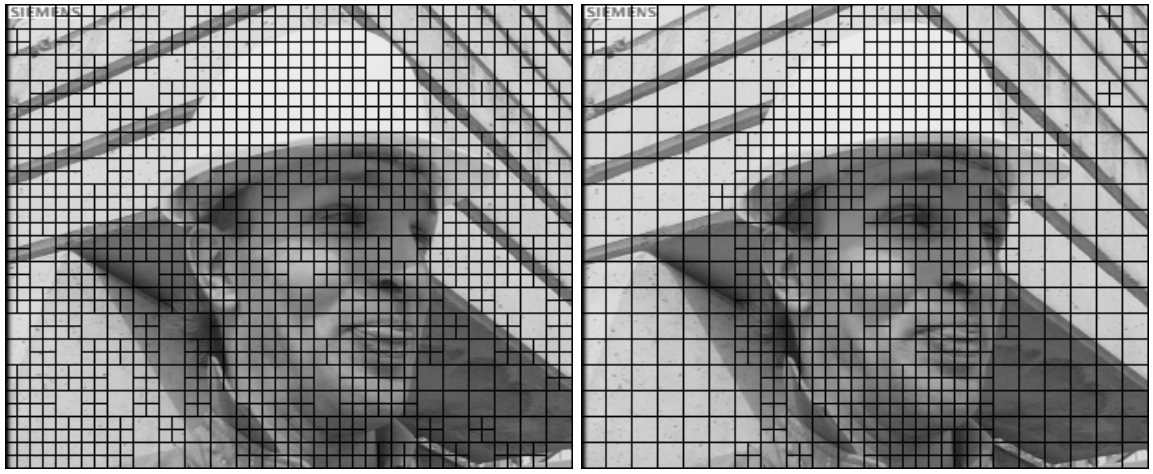
distance, are of a high similarity.

On the other hand, videos with static or slow-motion scenes may not see as much benefit from Multiple Reference Frames. In these cases, the content doesn't change as rapidly from frame to frame, and the use of additional reference frames may not provide a significant improvement in compression efficiency.

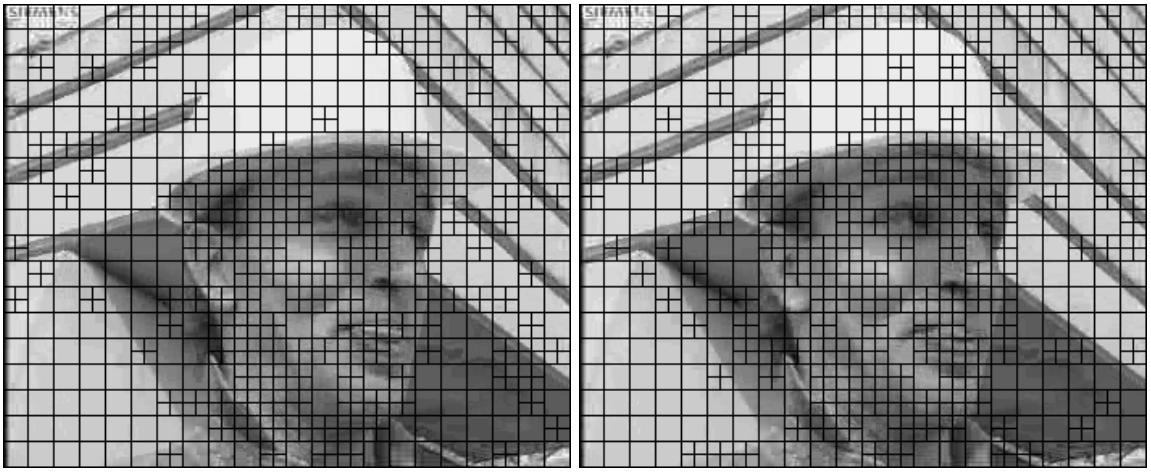
4.3 What trade-offs are involved in using a feature like Fractional Motion Estimation, other than performance? Some video standards support up to 1/8-pixel ME, would it be a good idea to use such a feature unconditionally? Why?

First trade-off is the increase in computational complexity. Higher precision in motion estimation requires more processing computational resources. This can be particularly challenging for real-time applications like video conferencing or live streaming. Moreover, the motion vectors themselves need to be encoded and transmitted so FME can also result in larger file sizes in some cases.
Given these trade-offs, unconditionally adopting such a feature isn't advisable. The choice to employ 1/8-pixel motion estimation should hinge on the unique use case and priorities at play.

4.4 For Variable Block Size, what kind of areas get larger block sizes in Intra frames? Are they the same in Inter frames? Why?



Frame 8 (Intra frame), QP=1 Frame 7 (Inter frame), QP=1



Frame 8 (Intra frame), QP=7 Frame 7 (Inter frame), QP=7

For Intra frames, almost no block gets the larger block size when QP is small. This is because splitting can make most of the spatial redundancies within each large block, because the top-right, bottom-left and bottom-right sub-blocks can utilize the reconstructed top-left block as the reference to reduce the bitstream size for residuals. When QP is large, some background blocks with repetitive values within the large block start to get the larger block size, because when QP is large, lots of details are lost so splitting the block cannot get a large saving in the bitstream size for residuals.

For Inter frames, the larger block sizes are in the background where there is little motion between frames, regardless of the QP value. This is because when the content of the block does not change much between frames, it is easy to find a very good reference, and using a single MV for the block can help with saving the bitstream size. The blocks where the content changes a lot are often split because it helps to find a better reference, therefore saving the bitstream on residuals.