On the Architecture of H.264 to H.264 Homogeneous Transcoding Platform

(Invited Paper)


Centre for Signal Processing, Department of Electronic and Information Engineering
The Hong Kong Polytechnic University
Hung Hom, Kowloon, Hong Kong
Tel: (852) 27666229 Fax: (852) 23624741 email: enwcsiu@polyu.edu.hk

Abstract - For homogeneous transcoding, we usually transfer a compressed video into a lower bit-rate and/or video with a smaller size. In this paper we introduce the general architecture of a downing sizing transcoder and propose some novel techniques for its practical realization, including motion vector re-estimation, sub-pixel motion re-estimation, mode re-decision, etc. We then generalize the idea of transcoding to video enlargement and propose a simple framework for its re-encoding. The paper ends with some useful remarks on the formation of super-resolution videos via the transcoding frame work.

Keywords - video coding, transcoding, motion estimation, H.264 and transcoding architecture.

I. INTRODUCTION

Video transcoding is a process of converting a previously compressed video bitstream into a different video format, size and/or transmission rate. In order to achieve bitrate reduction during the transcoding process, there are three common approaches[1-4]: 1) video downsizing, 2) frame rate reduction and 3) requantization of DCT coefficients for quality reduction. Video downsizing is to achieve a lower bit rate and to downscale an encoded video [1-4] produced by current video compression standards which employ motion compensated prediction. The conventional approach needs to decompress the video and perform downsclaling in the pixel domain. Recent useful transcoding techniques[5-10] include also Direct Addition Formulation for Frame-Skipping Transcoder[5] and efficient methods for transcoding directly in the transform domain[6-8]. These algorithms usually are able to transform coding quickly without sacrificing the quality.

The process to re-encode a high-resolution (SR) video from a compressed low-resolution (LR) video can also be considered as a kind of transcoding. Instead of downsizing we perform up-scaling and/or frame interpolation. These techniques form a good reference for constructing videos with with super-resolution quality and for SR video re-encoding.

II. TRANSCODING ARCHITECTURE

SDTV, HDTV, mobile videos and videos for other handheld devices will enter into their significant development in the coming 10 years. A video may just have been compressed into one format, and has yet to be transmitted and received by a device with different display capabilities. Video downsize-transcoding is a significant research topic of the previous years. However, most of the techniques concentrated on down-sizing with dyadic grids, such as the division by 2, 4, or 8. It is always desirable to transcode a compressed HDTV to SDTV, or a compressed video with a size of 1920x1080 to 1280x720. This requires a transcoding ratio of 3→2, which in a non-dyadic conversion, as shown on fig 1.

The easiest way to do transcoding is to decode the compressed video and re-encode it from the spatial domain. The most sophisticated way is to transcode it into the new format directly from transform domain[5,10]. However, it suffers from drift, for a long picture group. Hence, in this paper we just discuss the case on decoding and then re-encoding with fast algorithms. Extremely fast algorithms are allowable, since in the decoded video we have the original motion vectors, residual error signals, quantization parameters, etc. to be used as reference for the re-encoding.

Let us consider, for example, the case if motion vectors of the previously decoded frames are to re-used for the re-encoding (transcoding). Each new macroblock relates to four macroblocks in the corresponding frame of the previously encoded video. The spatial area of each contributing block is different as shown in fig 2. Furthermore the size of the image is reduced by a factor of 2/3 in each of the y- and x-directions, therefore the new motion vector has also to be reduced by the same amount. In general the motion vector of these four adjacent macroblocks may not be well-aligned, i.e. all motion vectors have different magnitudes and directions. In this case the re-estimation of the motion vector is more complicated. One simple approach to estimate the new motion vector is to take the average of the associated motion vectors of the four macroblocks and downscale it by 2/3 so that a resampled motion vector for the downscaled version of the video can be obtained. This is a simple approach, but the motion vectors obtained in this manner are not optimal. Equivalently, the approaches using i) align-to-average weighing, ii) align-to-best weighting, iii) align-to-worst weighting, and iv) adaptive motion vector re-sampling can be used. The latter
be used by any 4x4 blocks in the original MB will form a set of candidate directions. The prediction directions which were used by any 4x4 blocks in the original MB will form 4x4 blocks in the current MB will try these four directions to find out the best direction of each block.

B: Inter-mode Decision: If the target MB is decided to be inter-coded, the concept of majority mode again can be used. The majority mode is the mode that occupies the largest total area within overlapped MBs. If better quality is required, a scheme selecting the first three priority modes is used.

C: Fast MV re-estimation: In downsizing transcoding, the coding information of the original video can be reused to determine the MVs of image blocks in the transcoded video. Hence, a complete motion estimation[13] process is no longer necessary in a transcoding process. All MVs of overlapped original blocks will be used to help determine the MV of an image block of the downsampled video. In our approach, the original MVs of overlapped inter-blocks will be put into a list of candidate MVs for the downsampled block. These MVs are needed to be downsampled since the picture size is changed after transcoding. In addition to these candidate MVs, the list also contains a median MV for the sake of increasing the accuracy. Since integer-pixel scale is applied in this process, all candidate MVs are going to be rounded up to their nearest integer-pixel values. Instead of searching for the best match from a search region in a reference frame, only the positions pointed by the rounded candidate MVs will be considered. The candidate MV which gives the smallest RD cost will be

### Table 1: Coding Analysis of the Crowd Run sequence, (a) without B-frame and (b) with 2 B-frames

<table>
<thead>
<tr>
<th>Number of B-frames = 0</th>
<th>Total encoding time</th>
<th>763.07s</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Reading frames</td>
<td>11.64s (1.53%)</td>
</tr>
<tr>
<td></td>
<td>Integer-pixel ME (EPZS)</td>
<td>153.98s (20.18%)</td>
</tr>
<tr>
<td></td>
<td>Sub-pixel ME (EPZS)</td>
<td>227.23s (29.78%)</td>
</tr>
<tr>
<td></td>
<td>Other ME time</td>
<td>39.28s (5.15%)</td>
</tr>
<tr>
<td></td>
<td>Interpolation</td>
<td>88.19s (11.56%)</td>
</tr>
<tr>
<td></td>
<td>Preprocessing</td>
<td>1.34s (0.18%)</td>
</tr>
<tr>
<td></td>
<td>Intra prediction</td>
<td>131.78s (17.27%)</td>
</tr>
<tr>
<td></td>
<td>Luma residue coding</td>
<td>11.43s (1.50%)</td>
</tr>
<tr>
<td></td>
<td>Chroma residue coding</td>
<td>16.76s (2.20%)</td>
</tr>
<tr>
<td></td>
<td>Setting parameters</td>
<td>1.02s (0.13%)</td>
</tr>
<tr>
<td></td>
<td>Entropy coding</td>
<td>12.55s (1.64%)</td>
</tr>
<tr>
<td></td>
<td>Deblocking filtering</td>
<td>9.00s (1.18%)</td>
</tr>
<tr>
<td></td>
<td>Others</td>
<td>58.88s (7.72%)</td>
</tr>
<tr>
<td></td>
<td>PSNR</td>
<td>30.27dB</td>
</tr>
<tr>
<td></td>
<td>Bit-rate</td>
<td>23377.14kbps@50Hz</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Number of B-frames = 2</th>
<th>Total encoding time</th>
<th>1440.73s</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Reading frames</td>
<td>15.80s (1.10%)</td>
</tr>
<tr>
<td></td>
<td>Integer-pixel ME (EPZS)</td>
<td>305.05s (21.17%)</td>
</tr>
<tr>
<td></td>
<td>Sub-pixel ME (EPZS)</td>
<td>369.44s (25.64%)</td>
</tr>
<tr>
<td></td>
<td>Other ME time</td>
<td>349.97s (24.29%)</td>
</tr>
<tr>
<td></td>
<td>Interpolation</td>
<td>29.27s (2.03%)</td>
</tr>
<tr>
<td></td>
<td>Prediction for the direct mode</td>
<td>1.79s (0.12%)</td>
</tr>
<tr>
<td></td>
<td>Preprocessing</td>
<td>1.88s (0.13%)</td>
</tr>
<tr>
<td></td>
<td>Intra prediction</td>
<td>131.05s (9.10%)</td>
</tr>
<tr>
<td></td>
<td>Luma residue coding</td>
<td>10.32s (0.72%)</td>
</tr>
<tr>
<td></td>
<td>Chroma residue coding</td>
<td>17.80s (1.24%)</td>
</tr>
<tr>
<td></td>
<td>Setting parameters</td>
<td>1.35s (0.09%)</td>
</tr>
<tr>
<td></td>
<td>Entropy coding</td>
<td>10.89s (0.76%)</td>
</tr>
<tr>
<td></td>
<td>Deblocking filtering</td>
<td>8.46s (0.59%)</td>
</tr>
<tr>
<td></td>
<td>Others</td>
<td>187.68s (13.03%)</td>
</tr>
<tr>
<td></td>
<td>PSNR</td>
<td>30.04dB</td>
</tr>
<tr>
<td></td>
<td>Bit-rate</td>
<td>20788.35kbps@50Hz</td>
</tr>
</tbody>
</table>
chosen as the best match. In order to speed up the calculation of RD costs, the concept of Partial Distortion Search (PDS) is used[14]. If the current block size is 16x16, the calculation of the RD cost of the current candidate position will be early terminated when the partial cost value is checked to be larger than the previously computed smallest RD cost of the current block.

D. Sub-pixel Motion Re-estimation: After choosing the best integer candidate for the current MB, a refinement process using sub-pixel accuracy is performed to improve the quality. This step will be skipped if the RD cost of the chosen integer MV is smaller than a predefined threshold value. In this case, the chosen MV is good enough and no refinement is necessary.

A sub-pixel MV refinement comprises two stages: half-pixel refinement and quarter-pixel refinement. In the half-pixel refinement, the RD costs of one integer-pixel position and three half-pixel positions will be checked. The integer-pixel position is the position pointed by the rounded MV chosen in the previous motion re-estimation stage. The RD cost of this position is already known. The three candidate half-pixel positions are positions around the candidate integer-pixel position. There are 8 possible sets of candidate half-pixel positions. Without referring to the obvious 4 directions (horizontal right and left, vertical up and down), fig.3 gives the other four directions as indicated by the dotted and solid line triangles, corresponding to top-left, top-right, bottom-left and bottom-right directions from the central integer-pixel position (the black dot). The selection of a set depends on the location pointed by the MV chosen in the previous motion re-estimation process before rounding up. If the un-rounded location is in the up-left direction from the rounded integer-pixel position, positions 1, 2 and 8 will be the candidate checking points, etc. The RD costs of three selected half-pixel positions will be computed. The calculation of each RD cost will be early-terminated making use of again the PDS concept. The candidate position with the smallest RD cost among four candidates will be chosen at the end.

After finishing the half-pixel refinement, the RD cost of the chosen position will be compared to a predefined threshold value to decide whether the quarter-pixel refinement should proceed. If the RD cost is smaller than the threshold value, further refinement is skipped.

The quarter-pixel refinement process consists of 2 sub-stages. The positions to be considered in the first sub-stage depend on which position has been chosen in the half-pixel refinement. Fig.4 shows the search points for 9 different cases. The point (triangles represent half-pixel points; black dots represent the rounded integer-pixel point) indicated by an arrow in each case is the point chosen in the half-pixel refinement; the squares shown in different cases are the quarter-pixel points to be considered. The grey areas correspond to the search regions in different cases.

In the second sub-stage, one or two more quarter-pixel positions will be checked for some cases. Which extra quarter-pixel positions will be checked depends on which point has been chosen in the first sub-stage. For cases 2, 4, 6 and 8 in the first sub-stage, if the chosen point is the central half-pixel position of the T-shaped search region, no extra positions are required. If point 1 of the T-shaped search region shown in the picture (all squares are quarter-pixel points) was chosen, point 4 is checked. If point 2 was chosen, point 5 is checked. If pint 3 was chosen, both points 4 and 5 are considered.

IV. EXTENDING TO VIDEO AMPLIFICAITONS AND SUPER-RESOLUTION VIDEOS

With the development of visual communication and image processing, there is a high demand for high-resolution images such as video surveillance, remote sensing, medical imaging, HDTV and other entertainment applications. However, image resolution depends on the physical characteristics of the imaging devices. It is sometimes difficult to improve the image resolution by using better sensors because of the high cost or hardware physical limits. Super-resolution (SR) image reconstruction is a promising technique to increase the resolution of an image or sequence of images beyond the resolving power of the imaging system. A SR video may also require to be re-encoded for various reasons, including (i) to allow standard devices to view the SR videos without using additional conversion devices, (ii) to save SR video reconstruction time since the computing power of the viewing devices may not be sufficient and (iii) to avoid the unavailability of the SR video package at the viewing site. It is also true that broadcasting companies are looking for good technologies to convert videos between formats, different resolutions, and frame/bit rates. It is particularly difficult to do up-conversion of a compressed video, say for example from SDTV to HDTV, due to the missing data and blurring effect of edges by simple interpolation. Furthermore re-encoding of
these SR videos is required in many practical situations, since contents providers often have to standardize various video clips for uniform storage or transmission.

We propose an improved edge directed interpolation method by removing the accumulated interpolation error, and reducing correlation structure miss-match problem. Let us recall the transfer function of a Wiener filter,

$$Y(k) = \sum_{n=0}^{\infty} \alpha(n) x(k - n),$$

where $Y(k)$ is the predicted value, $x(k)$ is the original value, and $\alpha(n)$ is the weight of the filter.

(i) We have designed three parts of the system independently, such that the decoding will only produce LR video frame irrespective to interpolation. The interpolation is done initially within the decoded LR video frame without considering information from the temporal direction. The re-encoding part is done simply using the H.264 encoder, which requires relatively long encoding time. Fig.6 shows the result of a preliminary test on converting the “Rush Hour” sequence from the SD(1280x720) format to HD(1920x1080) format in the high profile of the H.264. The upper curve shows the quality and bit-rate of using fully decoding and encoding, with simple linear interpolation for magnification. The lowest curve shows the production of the compressed HR video by the simplest and quickest approach. In this approach we made use of the decoded motion vectors, decoded prediction modes, decoded mode sizes, etc. of the LR frame for the re-coding. This is done by some default arrangements. No motion estimation, no mode decision, etc. were required. It is about three times or more faster than that using the fully re-encoding mode, but it suffers from low PSNR and high bit rate. The middle curve shows a hypothetically case. This gives the best possible result that can be achieved if we do not perform full motion estimation, mode decision, etc. while the best parameters (MV, modes, etc.) were picked from the list of parameters decoded from the LR frame. This forms the target for our fast algorithm development.

(ii) A key part of this work is to design fast and accurate algorithms for obtaining encoding modes, motion vectors, or even transform coefficients without going through the heavy computational processes. The process is surprisingly close to downsize transcoding. We have to do (1) inter/inter mode re-decision, (2) intra mode re-decision (16x16 or 4x4), (3) inter mode re-prediction, (4) motion vector re-estimation, etc. The data and parameters available in originally encoded LR video are used to formulate the fast algorithms. The following strategies are used.

(a) Higher weights should to be given to parameters with larger areas. (b) All modes/MV (from LR frames) with the areas of LR blocks overlapping with the SR block should have a good priority to be checked. (c) The number of zero coefficients should be able to reflect the motion activities of the block. (d) Treat cases with different QP differently. (e) Refinement are made according to models built.

A: Interpolation Techniques: In order to remove the burring effect, edge enhancement is one of the best way to improve the quality of a super-resolution image/video sequence.

We propose an improved edge directed interpolation method by removing the accumulated interpolation error, and reducing correlation structure miss-match problem. Let us recall the transfer function of a Wiener filter,

$$Y(k) = \sum_{n=0}^{\infty} \alpha(n) x(k - n),$$

where $Y(k)$ is the predicted value, $x(k)$ is the original value, and $\alpha(n)$ is the weight of the filter.

In the previous years, most researchers, including us, just concentrated on downward conversion. Recently it is clear to us that there is a great need to develop techniques for upward conversion, including image/video up-sizing, frame interpolation, and super-video coding. This is a challenging topic but difficult one. Some works have been done by few researchers, but many technologies are still unavailable or premature. Hence this forms a fruitful direction for further research.

We have built an architecture which allows us to re-encode the SR video for either storage or transmission. We fully utilized the decoded data, statistics and parameters available from the previously encoded LR video to facilitate the super-resolution conversion. As shown in fig.5, a model has to be built for this investigation. The H.264 is our codec kernel. The model consists of three parts (a) “encoded bit-stream” decoding, (b) video interpolation and (c) re-encoding. We opt for a simple frame work as shown in fig.5, while many fundamental technologies are desperately needed for its practical realization.

Figure 5: Architecture of Transcoding Platform (Video Enlargement)

Figure 6: Video Enlargement

In order to remove the burring effect, edge enhancement is one of the best way to improve the quality of a super-resolution image/video sequence.
\( \alpha(k)'s \) are the linear prediction coefficients and \( x(n)'s \) are known samples. By optimizing the mean square error, MSE (\( \text{MSE} = \text{E}[e^2(n)] \)), we can come up with an equation for finding the coefficients of the Wiener filter for the interpolation,

\[
\mathbf{r}_{dx} = \alpha \mathbf{R}_{xx}
\]

where \( \mathbf{r}_{dx} = \text{E}[x(n)x(n-i)] \) is a cross-correlation function and \( \mathbf{R}_{xx} = \text{E}[x(n-k)x(n-i)] \) is an autocorrelation function.

The New Edge-Directed Interpolation (NEDI) scheme is to model a natural image as a second-order locally stationary Gaussian process which allows the interpolation using a simple linear prediction. The covariance of the image pixels in a local block (training window) can be used to obtain the prediction coefficients of the estimation problem. Consider the interpolation of an image \( X \) to a high-resolution image \( Y \).

\[
\begin{bmatrix}
0 & 1 & 2 & 3 & 4 & 5 \\
8 & 9 & 10 & 11 & 12 & 13 \\
16 & 17 & 18 & 19 & 20 & 21 \\
24 & 25 & 26 & 27 & 28 & 29 \\
32 & 33 & 34 & 35 & 36 & 37
\end{bmatrix}
\]

Figure 7: New Edge-Directed Interpolation (NEDI)

In fig.7, the numbers are used to represent the locations of the original low resolution pixel points. The solid point, entitled as \( y_i \), as shown in fig.7(a) is a high resolution point to be interpolated from four neighbor low-resolution pixels \( \{x_{18}, x_{19}, x_{26}, x_{27} \} \). In order to have the simplest formulations, one-D representation has been used as far as possible for explanation. The predicted pixel becomes,

\[
y'_i = \sum \alpha_i x_i = \sum \alpha_i x_i \quad \text{selected surrounding pixels}
\]

From eqn.1, we have

\[
\alpha = \mathbf{R}_{xx}^{-1} \mathbf{r}_{dx}
\]

The computation of \( \mathbf{r}_{dx} \) (cross-correlation between \( y_i \) and it’s interpolating points) and \( \mathbf{R}_{xx} \) (the auto-correlation among interpolating points) would require knowledge of statistics of \( y_i \) with its neighbors which are not available before the interpolation. This difficulty is overcome by the “geometric duality” property, as illustrated fig.7(b). The correlations between \( y_i \) in the high resolution domain and its neighbors

\[
\begin{bmatrix}
x_{18} \\
x_{19} \\
x_{26} \\
x_{27}
\end{bmatrix}
\]

\[
\begin{bmatrix}
x_9 \\
x_{11} \\
x_{25} \\
x_{27}
\end{bmatrix}
\]

where elements of \( y \) are the training points and the row of \( \mathbf{C} \) are the set of respective points to interpolate elements of \( y \). In this case we have \( \mathbf{r}_{dx} = \mathbf{C}^T y \) and \( \mathbf{R}_{xx} = \mathbf{C}^T \mathbf{C} \).

To interpolate a point between two vertical LR pixels (2nd step), the same procedure is used with a rotation by an angle \( \pi/4 \) as shown in figs.8(a) and (b). In fig.8, circles represent LR pixels and grey dots represent the interpolated points in the 1st step (fig.7) and small black dots represent HR points to be interpolated. To save computation, the NEDI adopted a hybrid approach, this correlation based interpolation is applied to edge pixels only and bilinear interpolation is applied to non-edge pixels (i.e. pixels in smooth regions).

However, the NEDI suffers from the prediction error propagation problem which limits the performance of the algorithm. NEDI is a two-step interpolation scheme, where the first step makes use of the original pixels for interpolation, whilst the second step makes use of the interpolation results obtained from the first step, i.e. gray pixels in Fig.8 to obtain the interpolation pixel (the small black dot). The interpolation error in the first step will be propagated to the second interpolation step, and thus causes the interpolation error propagation problem. At the same time, NEDI also suffers from covariance structure miss-match problem. The span of pixels does not represent the best coverage in the HR domain. Hence a different set of pixels could give a better interpolation of the edges. We resolve the problem by suggesting a new version. The first step is the same as before. In the second step, we propose to interpolate the unknown pixels by a sixth-order linear prediction with a training window as shown in figs.8(c) and (d) by using points on the original LR domain only. This completely eliminates the error propagation problem. To reduce the covariance miss-match problem, we may use multiple low-resolution training window candidates, i.e. a scheme to choose one from more than one low-resolution training windows to represent the covariance of the high-resolution block to perform the linear prediction, as shown in fig.8(d).

B. Super-Resolution Video: Since the interpolation from a frame to form an enlarged frame is restricted by the resolution and information available from the original image, it is very natural to use more frames (both in temporal and spacial domains) to construct the enlarged frame. An enlarged frame obtained from more then one orginal frame is defined as a super-resolution frame (video) in this paper. This can be achieved by both non-iterative and iterative approach. Due to
the computation time for motion estimation, mode decision, and size. It is seen that there is a substantial reduction in for converting the Crowd Run of size 1280x720 to 2/3 of this size. The table shows the results of our realization of the transcoding results using the H.264 JM12.2 and using our fast approaches.

### Table 2: Comparison of results using JM12.2 and our fast approach

<table>
<thead>
<tr>
<th></th>
<th>JM 12.2</th>
<th>After fast</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer-pel ME</td>
<td>153.98s (20.18%)</td>
<td>20.50s (7.21%)</td>
</tr>
<tr>
<td>Sub-pel ME</td>
<td>227.23s (29.78%)</td>
<td>30.25s (10.64%)</td>
</tr>
<tr>
<td>Other ME</td>
<td>39.28s (5.15%)</td>
<td>5.23s (1.84%)</td>
</tr>
<tr>
<td>Intra prediction</td>
<td>131.78s (17.27%)</td>
<td>17.54s (6.17%)</td>
</tr>
<tr>
<td>Others</td>
<td>210.80s (27.63%)</td>
<td>211.01s (74.14%)</td>
</tr>
<tr>
<td>Total time</td>
<td>763.07s</td>
<td>284.32s (2.68X)</td>
</tr>
</tbody>
</table>

Table 2: Comparison of results using JM12.2 and our fast approach

Figs.9 and 10 show that our approach on interpolation for the enlargement of an image and simulated SR video reconstruction. The reader may note the bar and connection parts above the wheel of fig.10, which look more smooth and sharper. The effect is more effective if we use some further level of amplifications.

**VI. CONCLUSION AND FURTHER DEVELOPMENT**

In this paper we have provided initially an analysis of the requirement for video downsize transcoding. We then provide a set of technologies for an efficient and simple architecture for the transcoding. These new technologies are able to speed up the individual parts of the process over ten to several tenth times, whilst a speedup of 2.7 times for the transcoding process has also be achieved. The architecture can be directly applied to video enlargement transcoding, and eventually leads to a systematic approach for a basic kernel for super-resolution video construction. This is a fruitful direct of research.

**ACKNOWLEDGMENT**

This work is supported by the Centre for Multimedia Signal Processing, Hong Kong Polytechnic University and the Research Grant Council of the Hong Kong SAR Government (PolyU 5268/06E), and in part by the Hong Kong Applied Science and Technology Research Institute (ASTRI).

**REFERENCES**


