In Figures 9.11 and 9.12 there are some examples for the object functionalities mode. Images have been obtained from the *Akiyo* sequence (QCIF@10Hz). Both the base and enhancement layer have been encoded with a budget bit rate of 64 kbps. In the figure, odd rows represent, from left to right, the original image, the base layer and the enhancement layer. Even rows show the final partitions for the base and enhancement layers.

Object functionalities mode - Summary of results: In object functionalities mode, enhancement layer partitions are determined by the need to include all the contours of the semantic object. In this case, the number of regions in the enhancement layer partition is heavily influenced by the level in the Partition Tree at which all the necessary contours can be found. For instance, in Figure 9.2, the first splitting level in the base layer Partition Tree does already contain all the necessary contours, so that the decision algorithm in this case can choose to form the enhancement layer partition with few regions more than in the base layer. In other cases the wanted contours appear at levels where the number of regions is high and, therefore, the decision algorithm is forced to form the enhancement layer partition with many regions. Figures 9.11 and 9.12 are examples of this behavior.

For texture coding, layer intra techniques are much more important than in full frame mode. In this case there is a high number of regions in the enhancement layer forced by the necessity to include the contours of the object. The cost of the motion vectors for inter-frame techniques makes them less efficient for small regions.
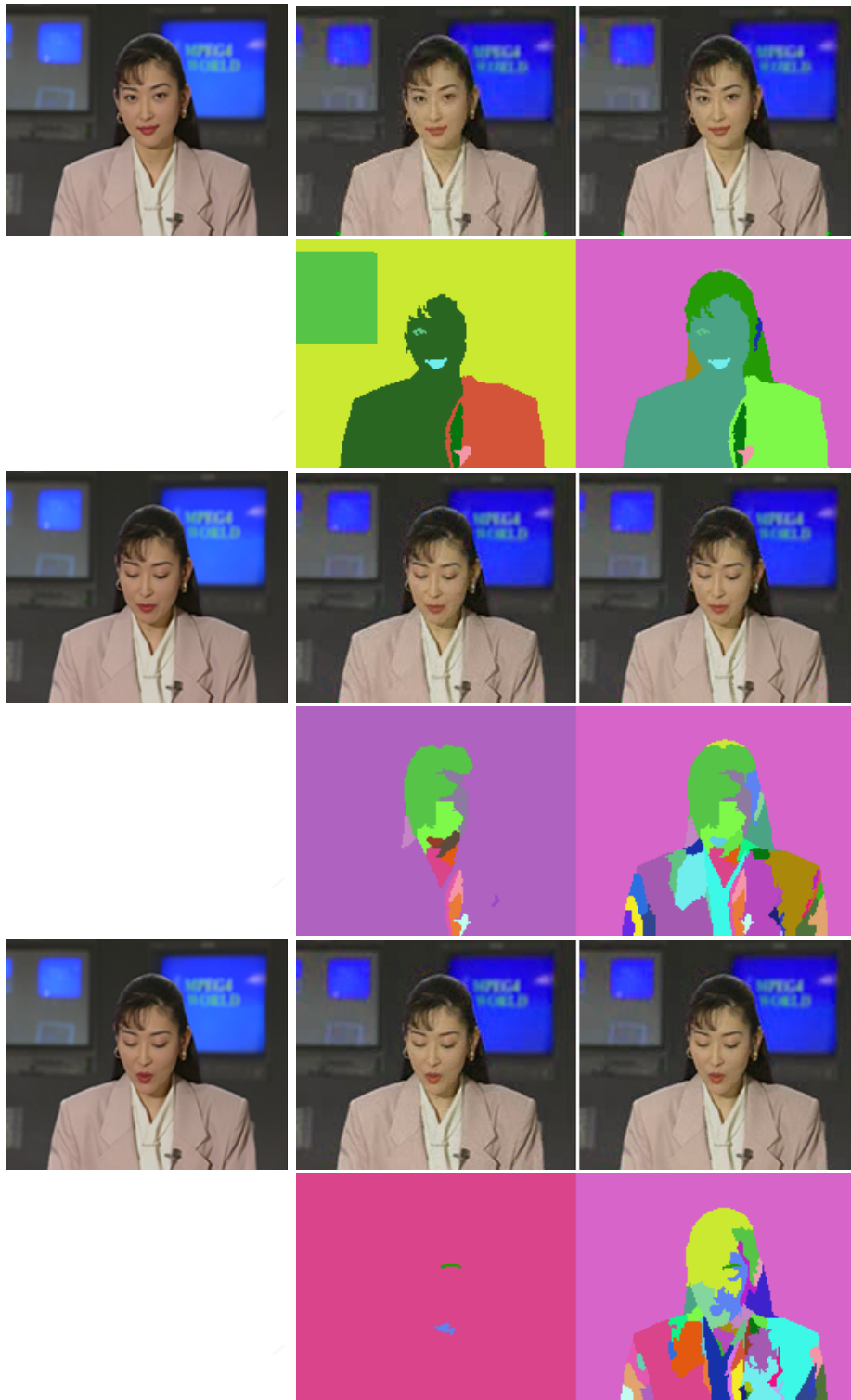
Figure 9.11: Example of Object functionalities PSNR scalability: Akiyo sequence (QCIF@10Hz), 64+64 kbps. From left to right, the figure shows the original image, the base layer, the enhancement layer and the corresponding partitions (below the images) for frames #0 (intra), #60 and #120 (inter)
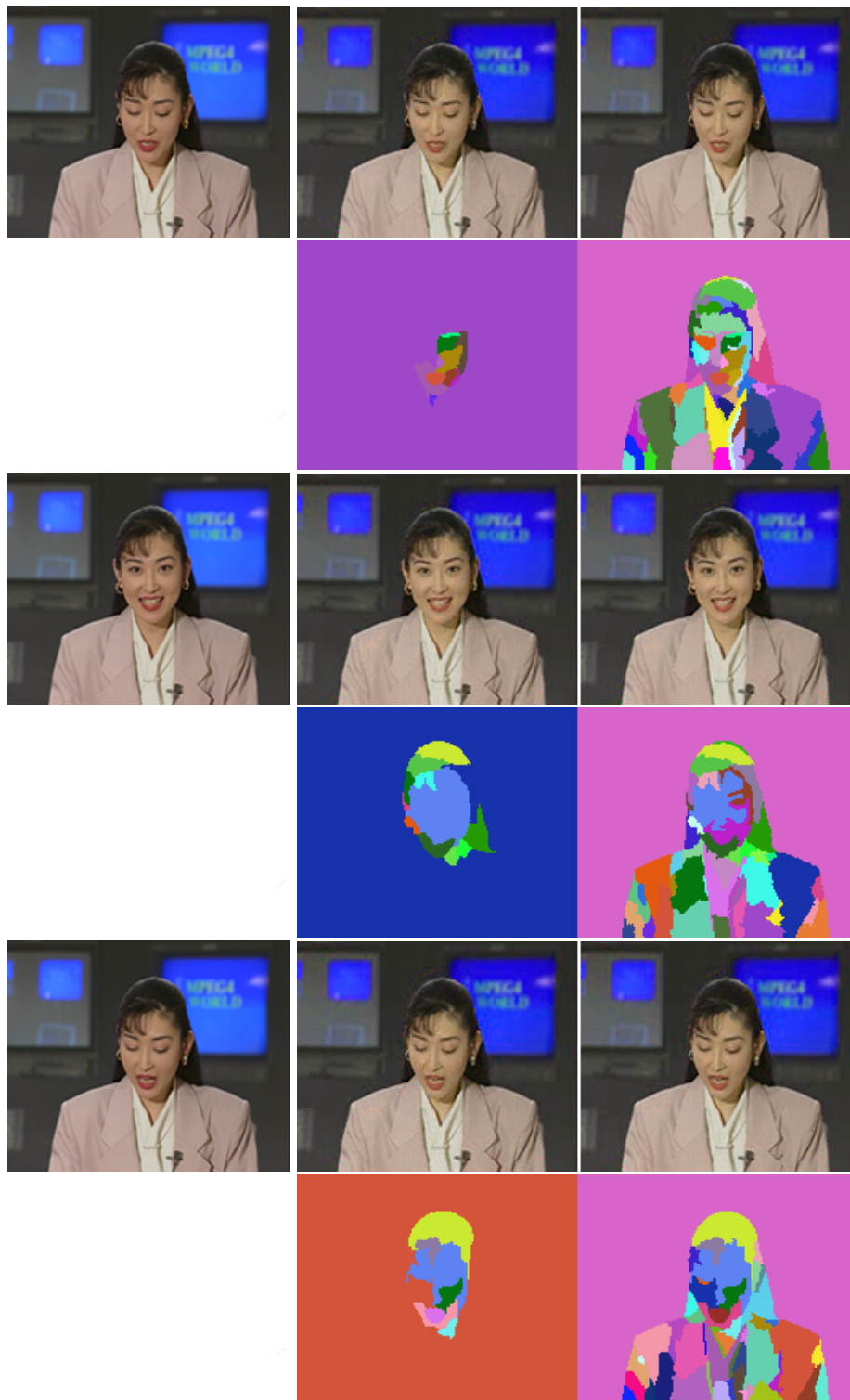
Figure 9.12: Example of Object functionalities PSNR scalability: Akiyo sequence (QCIF@10Hz), 64+64 kbps. From left to right, figure shows the original image, the base layer, the enhancement layer and the corresponding partitions (below the images) for frames #180, #240 and #297 (inter)

9.2 Spatial scalability

As in the previous Section, we will start giving some details of the system operation and then, some experimental results will be provided.

Analysis of Projection and Partition Tree construction

Figure 9.13 shows an example of the partition up-sampling algorithm presented in Section 8.7.2. The result is compared with a simpler technique consisting on nearest neighbor interpolation. While the simpler technique produces staircased contours, the presented technique allows up-sampling a given partition and to match the resulting contours with those of a reference high resolution partition, so both partitions can be merged without errors.
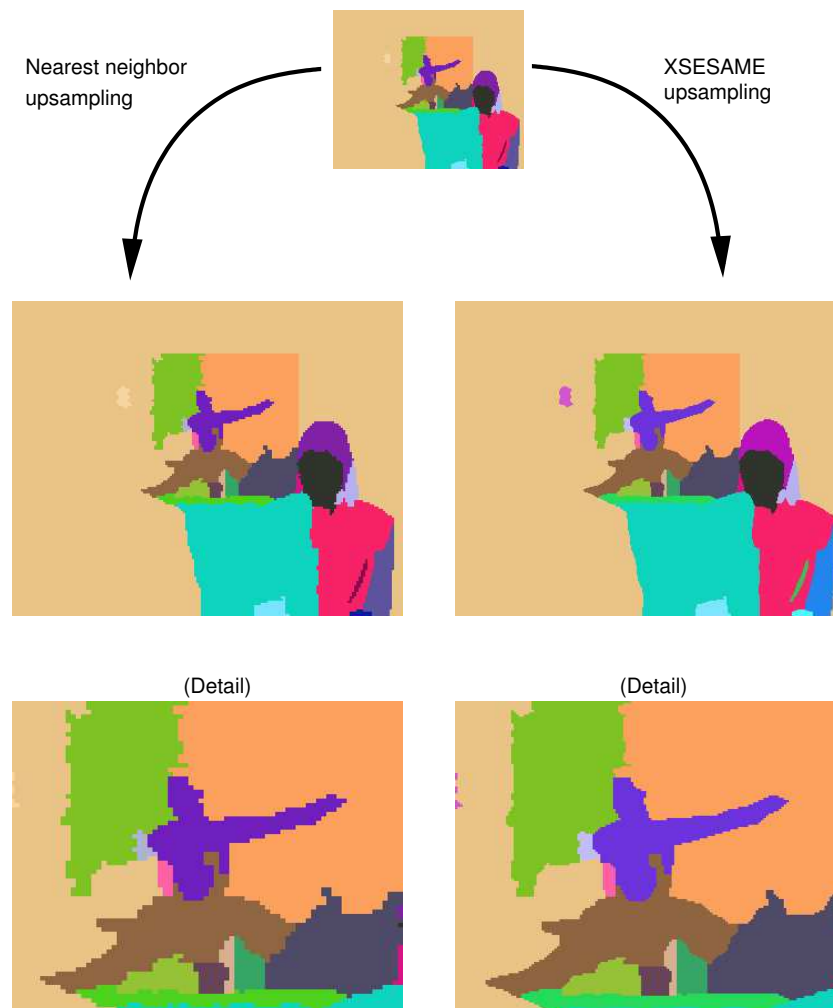

Figure 9.13: Example of partition up-sampling