A CONSTRAINED ADAPTIVE SCAN ORDER APPROACH TO TRANSFORM COEFFICIENT ENTROPY CODING

Ching-Han Chiang, Jingning Han, Yaowu Xu
Google
Coding of Transform Block Coefficients

- Use a scan order to code transform block coefficients sequentially.
- Code an end-of-coefficient token after last non-zero coefficient in the scan order.
- The rest of zero coefficients can be skipped.

```
72 -> 24 -> 30 -> 0 -> 4 -> 2 -> eob
```
Motivation

- Arrange most zero coefficients to the tail.
  - The end-of-coefficient token will allow us to ignore zero coefficients in the tail.
- Transform block statistics are different across different video clips.
  - Using a predefined scan order will waste bits in coding zero coefficients.
  - An adaptive approach is needed.
Context Dependency

- How about we design an adaptive scan order merely by sorting the non-zero probabilities of positions in transform blocks?
- The above/left coefficient context dependency is applied in entropy coding to exploit the remaining inter-coefficient correlations.
- The scan order obtained from sorting the non-zero probabilities may have conflict with the context dependency.
Topological Sort - Example

- By applying topological sort, one can generate a scan order that mostly follows the descending order of non-zero probabilities without violating the context dependency.

![Diagram showing topological sort example](image)

- Sort $\text{Pe}[i][r][c]$
- Draft scan index of the draft scan order
- DAG of coefficient context dependencies
- Scan index of adaptive scan order
- Apply topological sort with draft scan order and context dependencies
Topological Sort - Resolve Context Conflict

estimating non-zero probability

scan order

without context conflict

with context conflict

0 1 5

2 3 6

4 7 8
Estimation of Non-zero Probabilities of Transform Coefficients

- The cost of transmitting non-zero probabilities or the adaptive scan order from encoder to decoder is impractically large

- Moving window estimation
  - Estimate non-zero probabilities of transform block coefficients for $i$-th frame.
  - Non-zero coefficient counts and number of transform blocks
    - $C[i-1][r][c]$  
    - $M$ → number of transform blocks
  - Observed non-zero probabilities
    - $P_c[i-1][r][c] = C[i-1][r][c]/M$
    - per-frame update
  - Estimating non-zero probabilities
    - $P_e[i][r][c] = (1 - k) \times P_e[i-1][r][c] + k \times P_c[i-1][r][c]$
    - per-frame update
Performance

- Coding gains (BDRate) over predefined scan order scheme on VP9
  - Low-resolution dataset
    - 40 videos with resolutions of 240p, SIF or CIF
    - 1.04%
  - Mid-resolution dataset
    - 23 videos with resolutions of 480p or 4CIF
    - 0.93%
  - High-resolution dataset
    - 38 videos with resolutions 720p, 1080p or XGA
    - 1.13%

- Encoder/decoder Time
  - Encoder: non-observable
  - Decoder: +1%
Q & A
Context Dependency - A Sparse but Strict Constraint
Algorithm 1 Modify Scan Order by Topological Sorting

Parameter:
- len \(\triangleright\) side length of the transform block
- N \(\triangleright\) len \(\times\) len
- c_{idx} \(\triangleright\) coefficient index defined by \(c_{idx} = r \ast len + c\)
- ds_{idx} \(\triangleright\) draft scan order index
- s_{idx} \(\triangleright\) scan order index
- visit[c_{idx}] \(\triangleright\) table of coefficient scanned indicators

Input:
- draft_scan_order[ds_{idx}] \(\triangleright\) obtained by sorting \(P_r[i][r][c]\).
- ctx_dep[c_{idx}] \(\triangleright\) coefficient context dependencies

Output:
- scan_order[s_{idx}] \(\triangleright\) adaptive scan order

Procedure: TopologicalSort
- for \(c_{idx} = 0\) to \(N-1\) do
  - visit[c_{idx}] = False
- end for
- s_{idx} = 0
- for \(ds_{idx} = 0\) to \(N-1\) do
  - \(c_{idx} = \text{draft_scan_order}[ds_{idx}]\)
  - ContextConflictSolver(c_{idx}, ctx_dep, scan_order, visit, s_{idx})
- end for

Algorithm 2 Recursive Context Conflict Solver

Input:
- s_{idx} \(\triangleright\) to-be-assigned scan order index
- c_{idx} \(\triangleright\) coefficient index
- ctx_dep[c_{idx}] \(\triangleright\) coefficient context dependencies
- visit[c_{idx}] \(\triangleright\) table of coefficient scanned indicators

Output:
- s_{idx} \(\triangleright\) increment it by one after it is assigned
- scan_order[s_{idx}] \(\triangleright\) adaptive scan order

Procedure: ContextConflictSolver
- for each ctx_c_{idx} in ctx_dep[c_{idx}] do
  - if visit[ctx_c_{idx}] is False then
    - ContextConflictSolver(nb_c_{idx}, ctx_dep, scan_order, s_{idx})
  - end if
- end for
- s_{idx} = s_{idx} + 1
- scan_order[s_{idx}] = c_{idx}
- visit[c_{idx}] = True
- s_{idx} = s_{idx} + 1