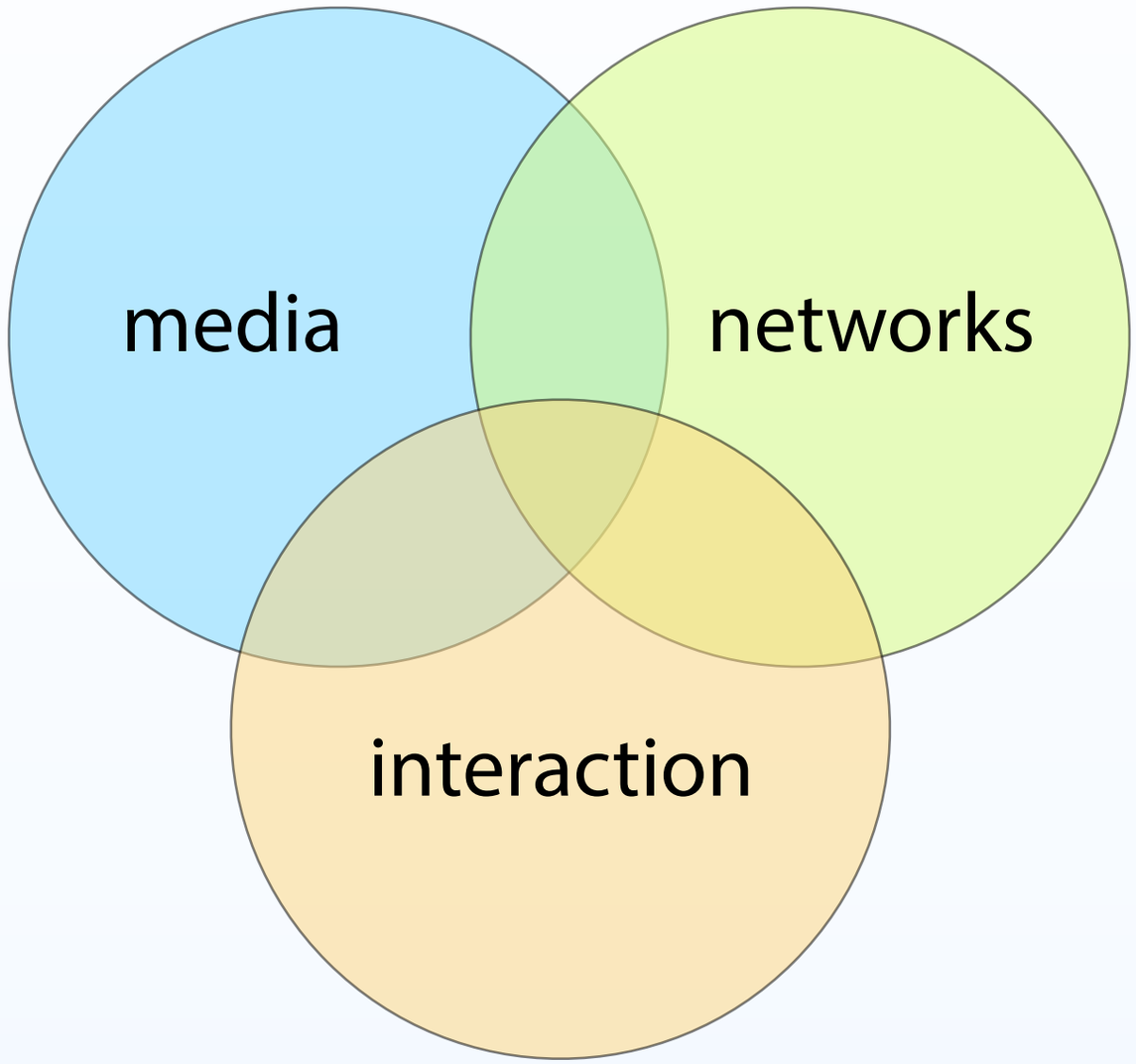
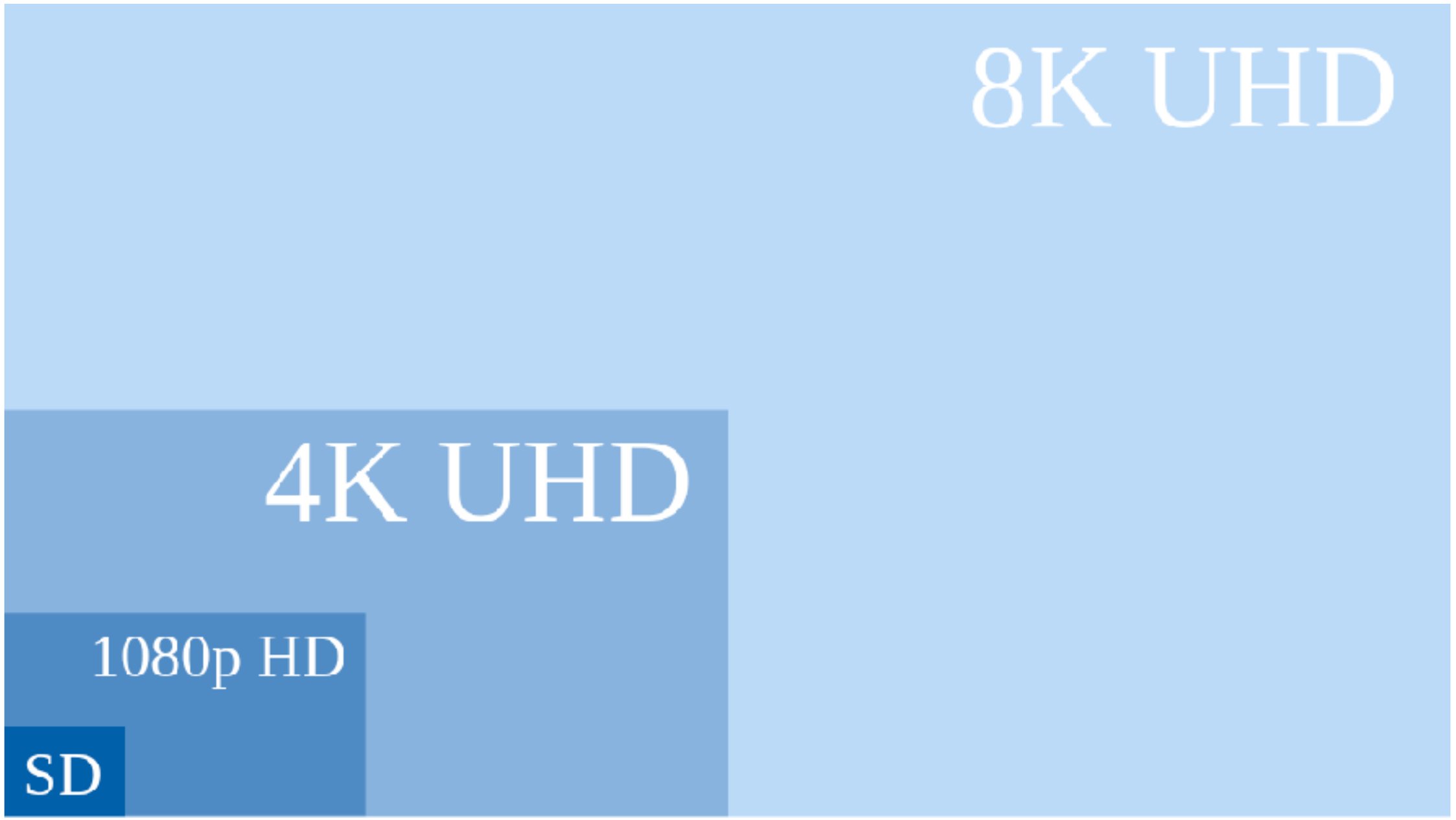
Zoomable Video Streaming

Ooi Wei Tsang
ooiwt@comp.nus.edu.sg

My Research Interests





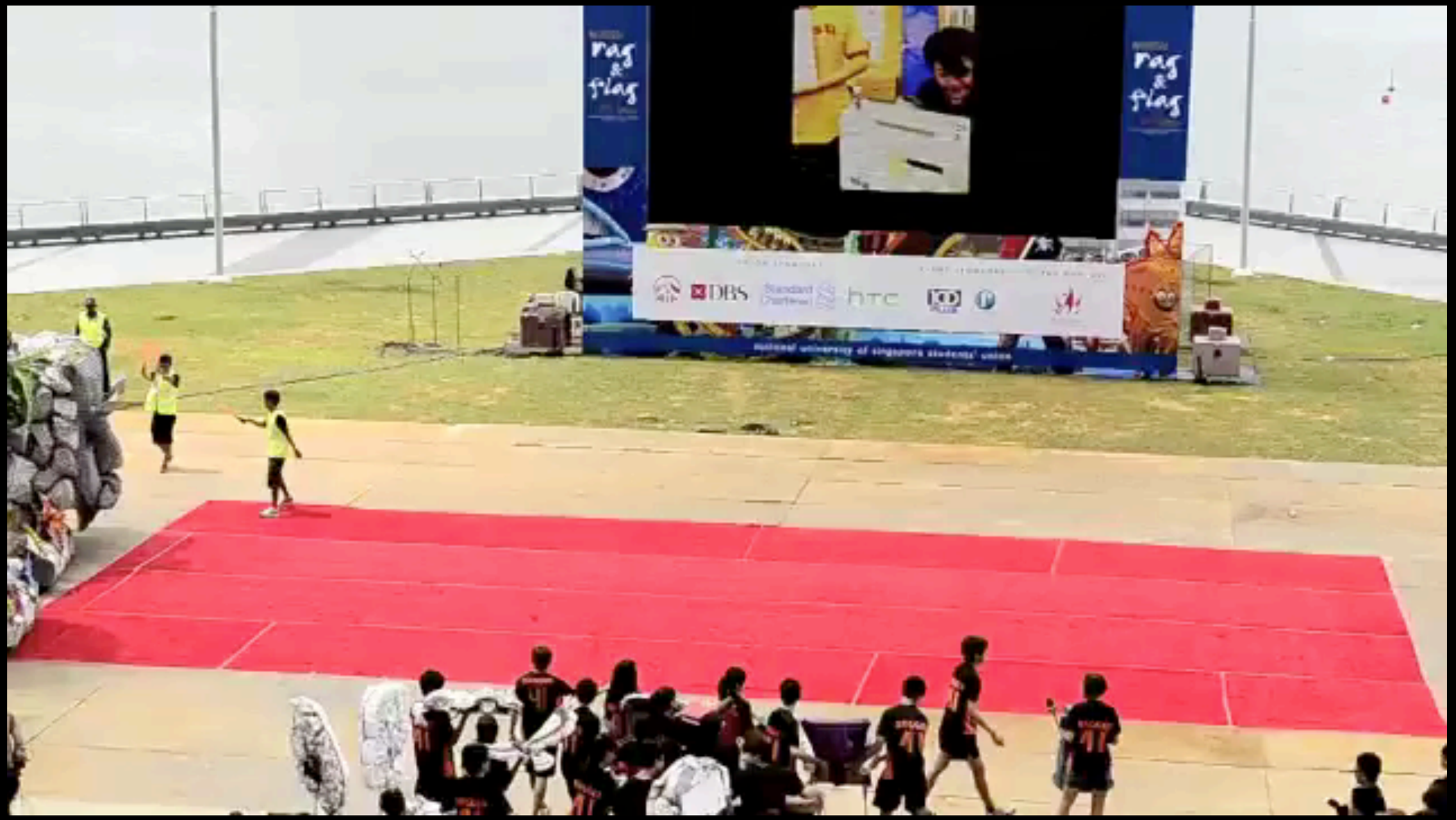
UHDTV Resolution Chart, from Wikipedia

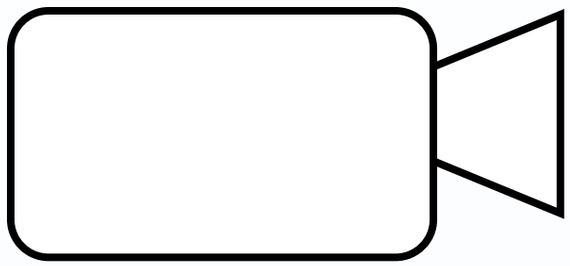
bits captured

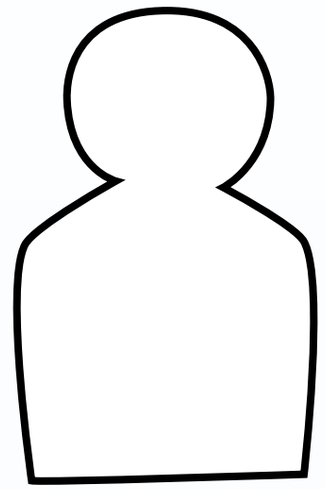
>

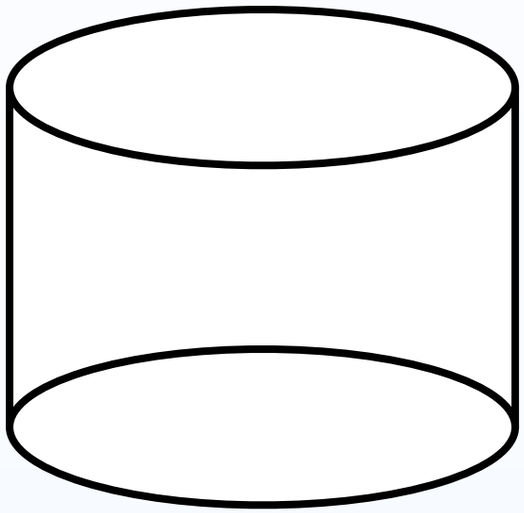
bits displayed

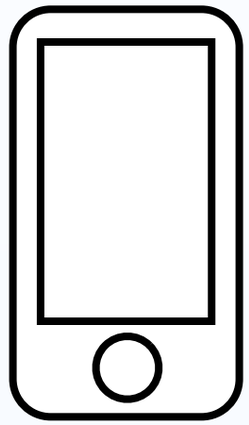

We want to zoom
and pan in videos,
just like in photos
and Web pages

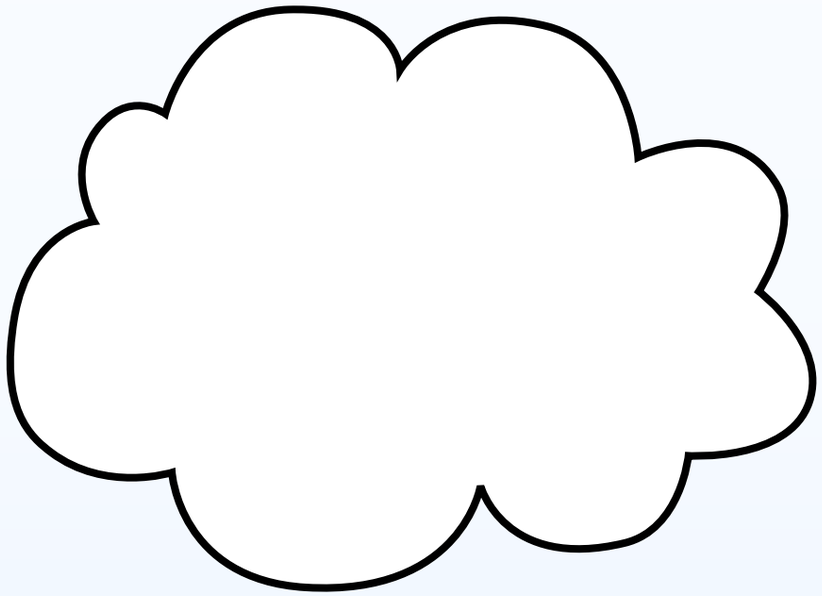

rag & flag
DBS
htc
rag & flag



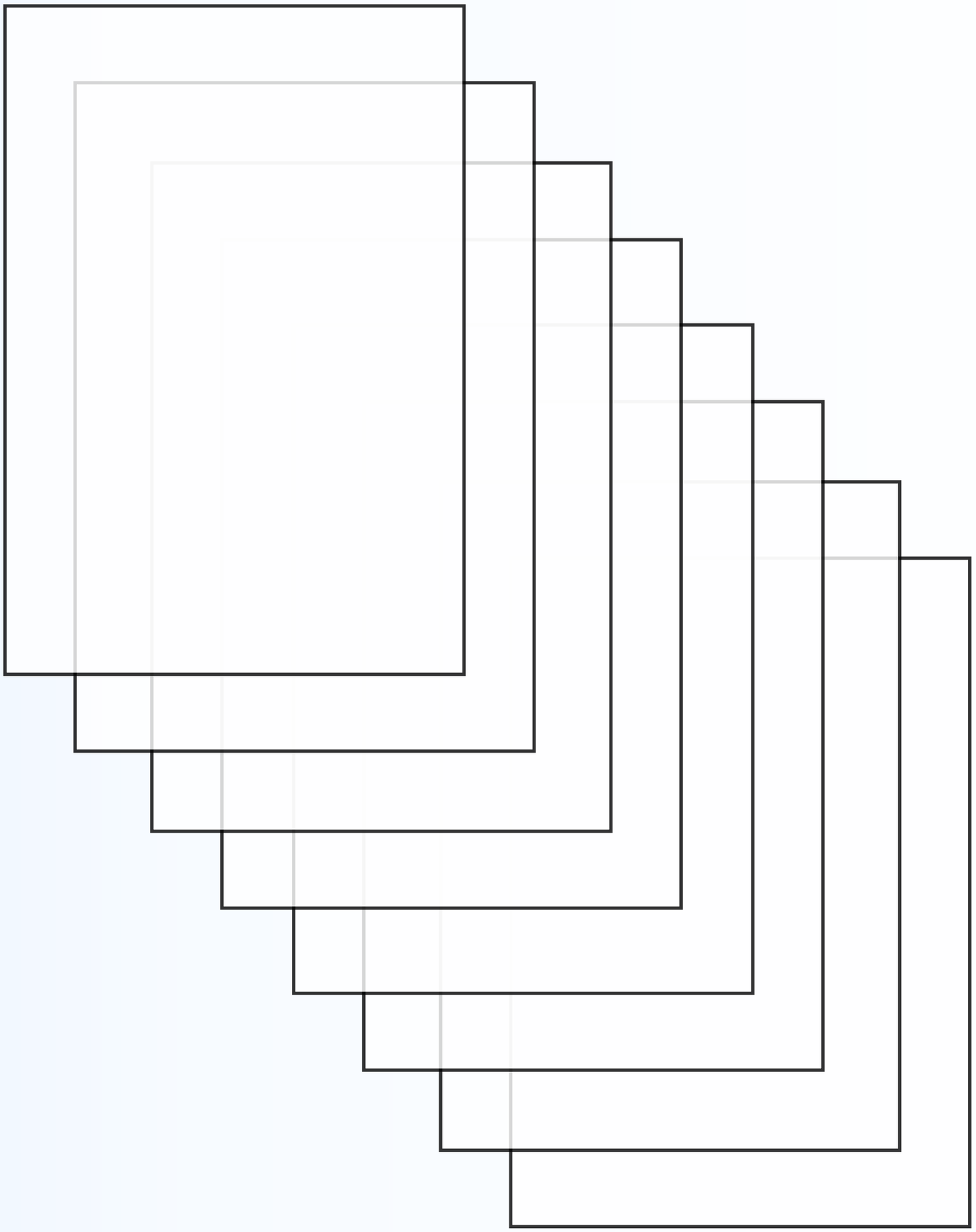


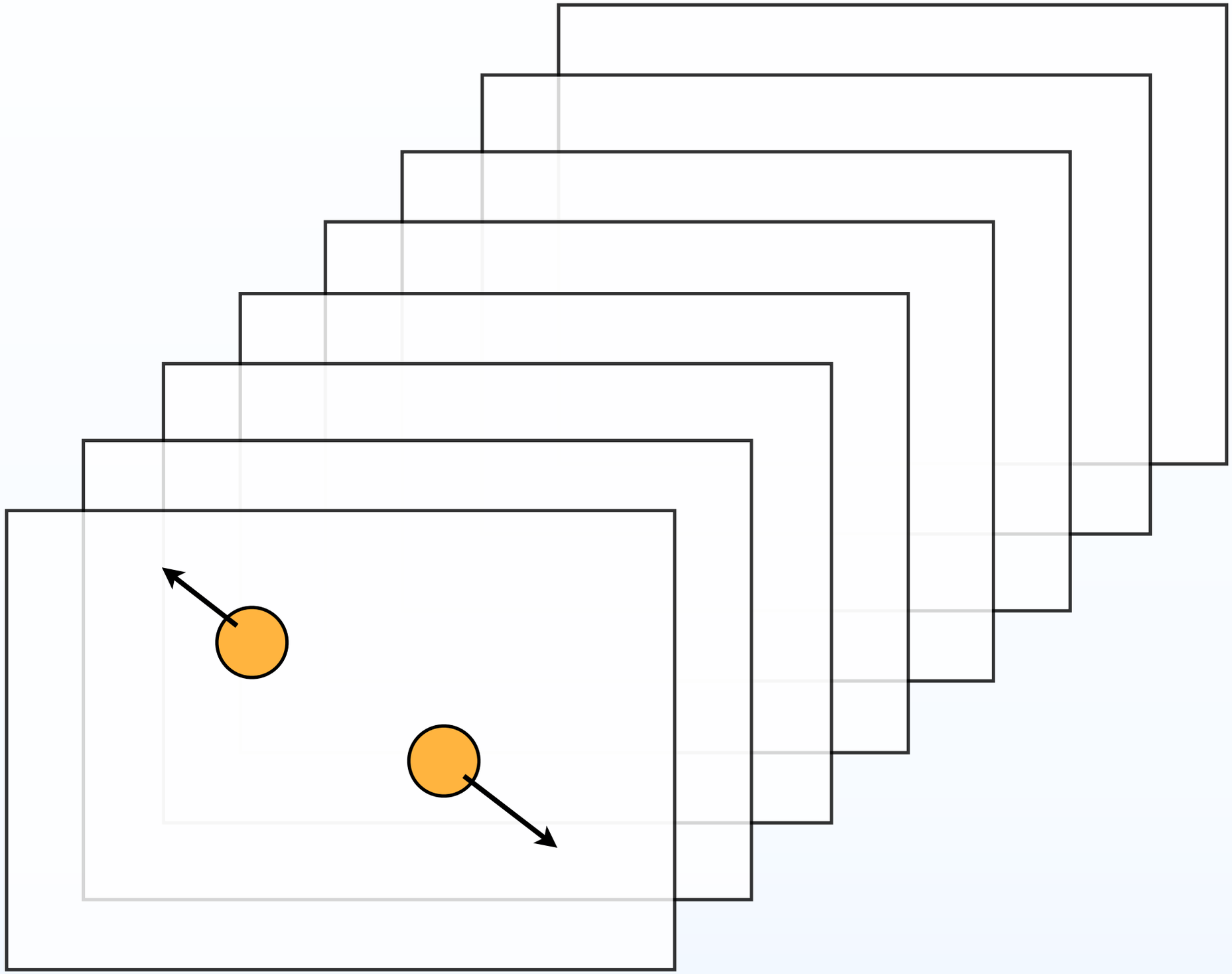


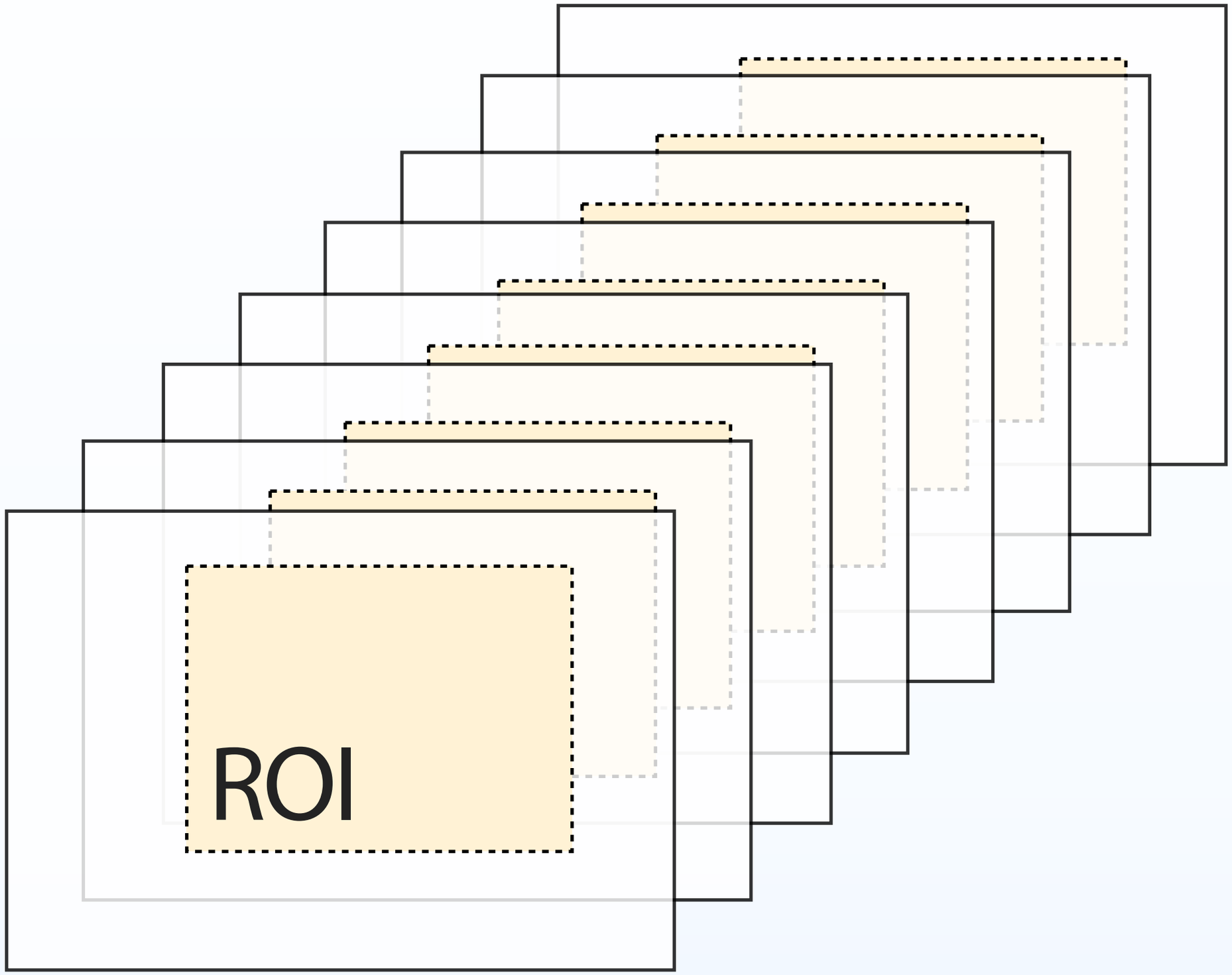




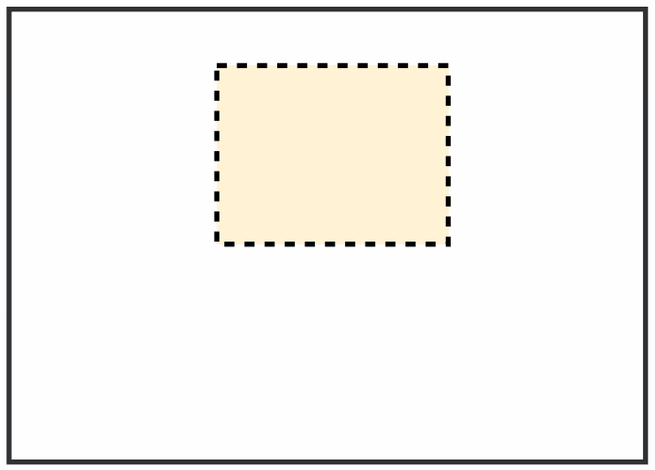
How to zoom on a video?





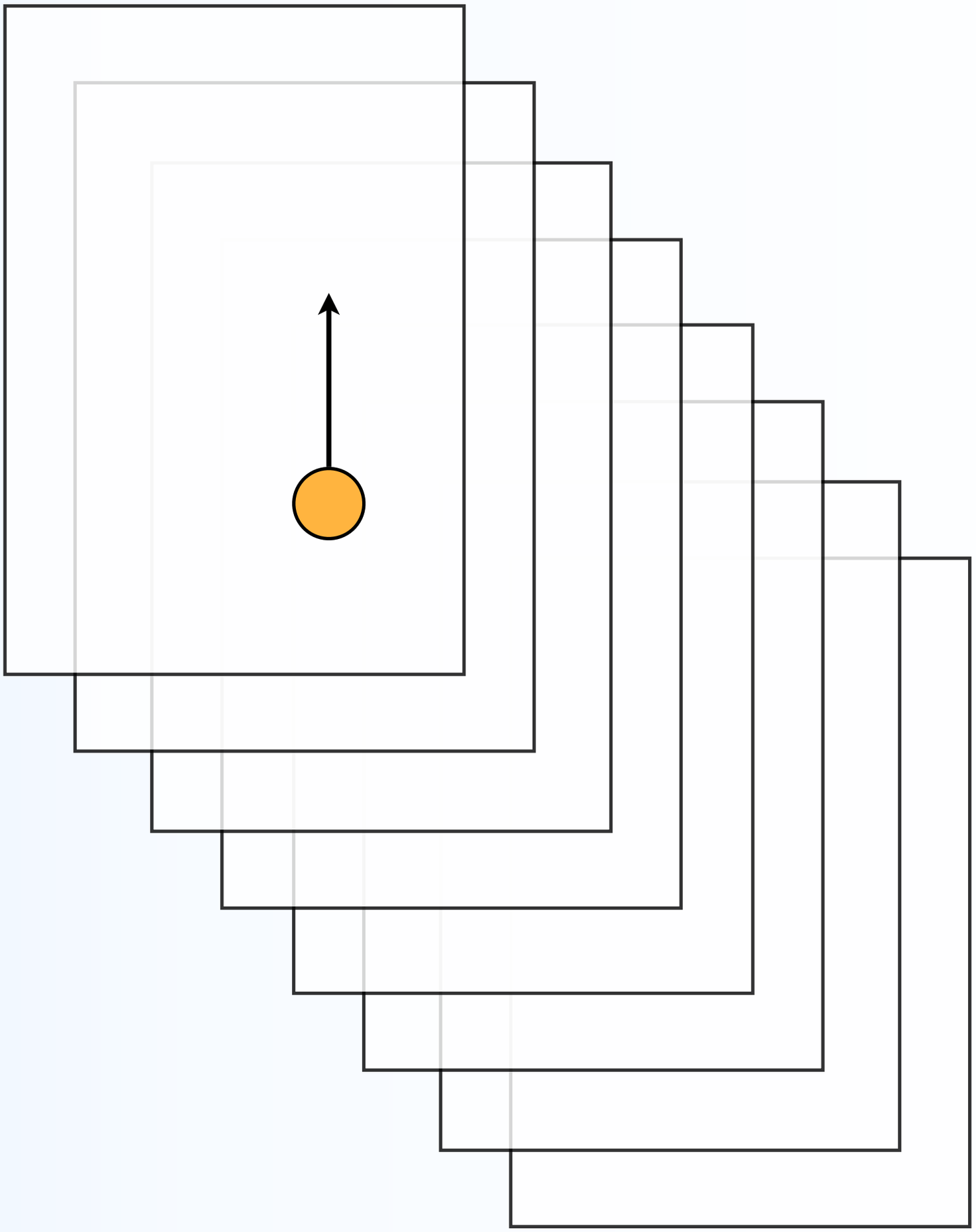

ROI

decompress

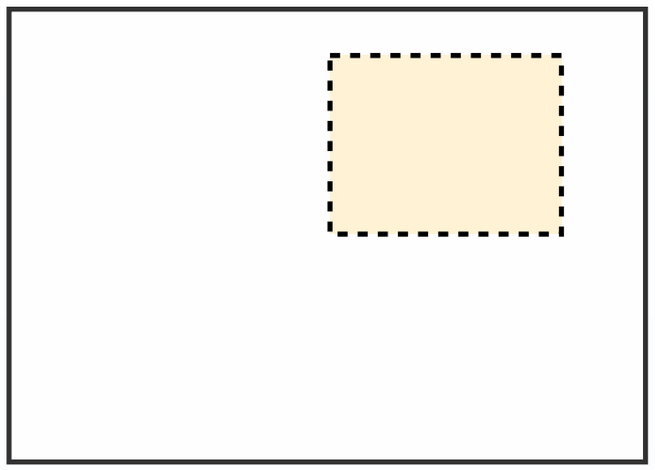


crop

scale



decompress



crop

scale

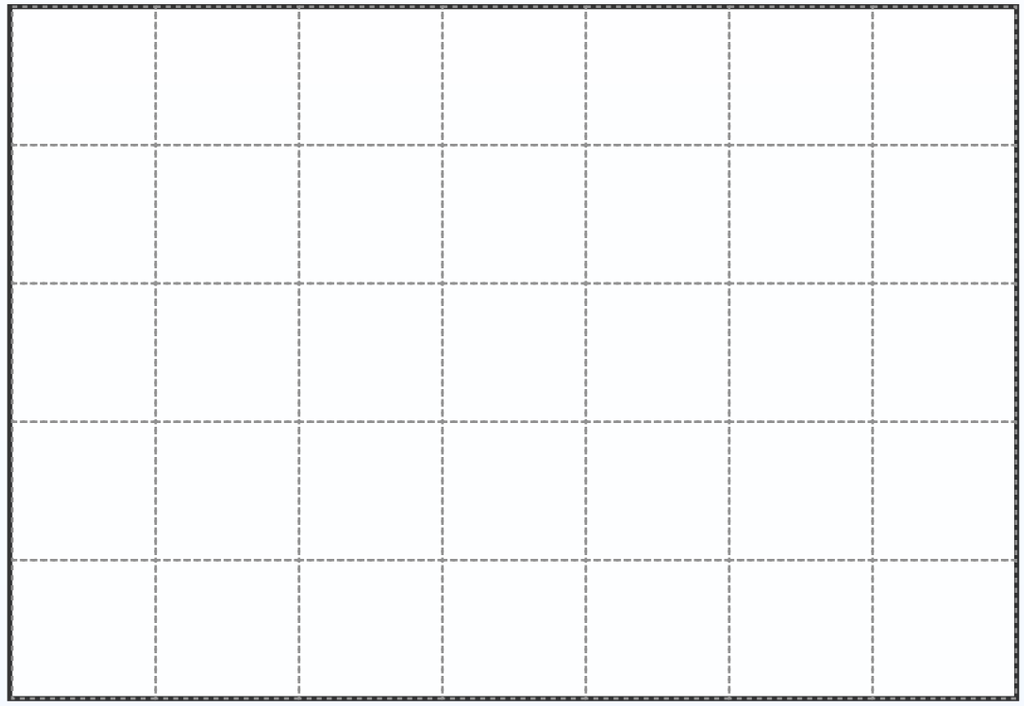
We need not decompress the whole frame!

detour:

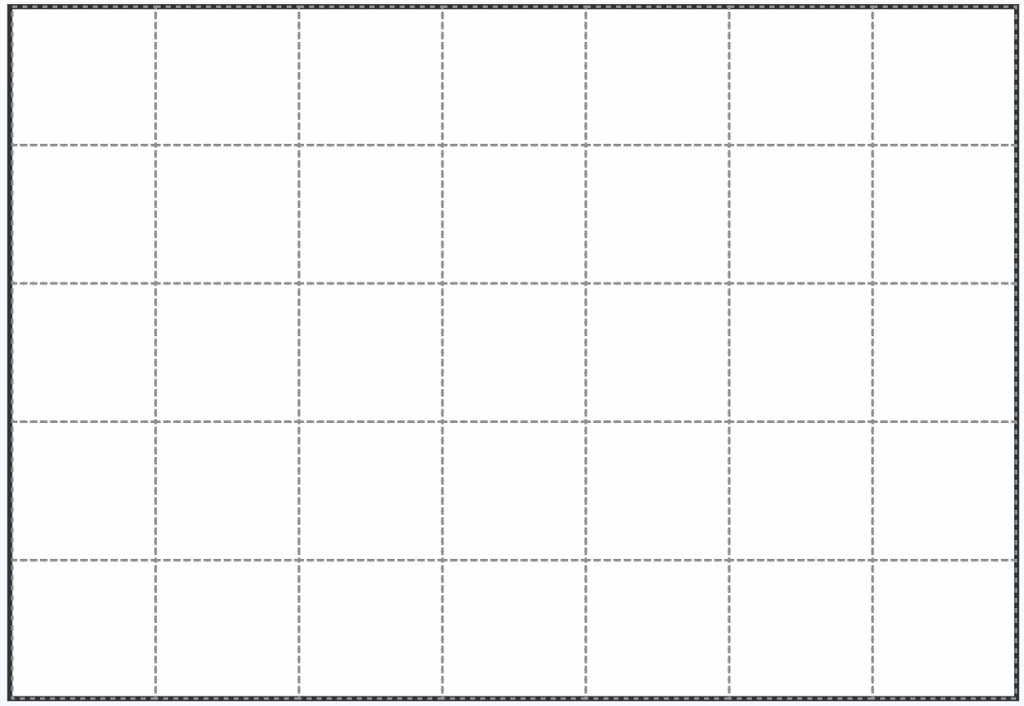
video compression in 2 minutes

a video frame is divided in

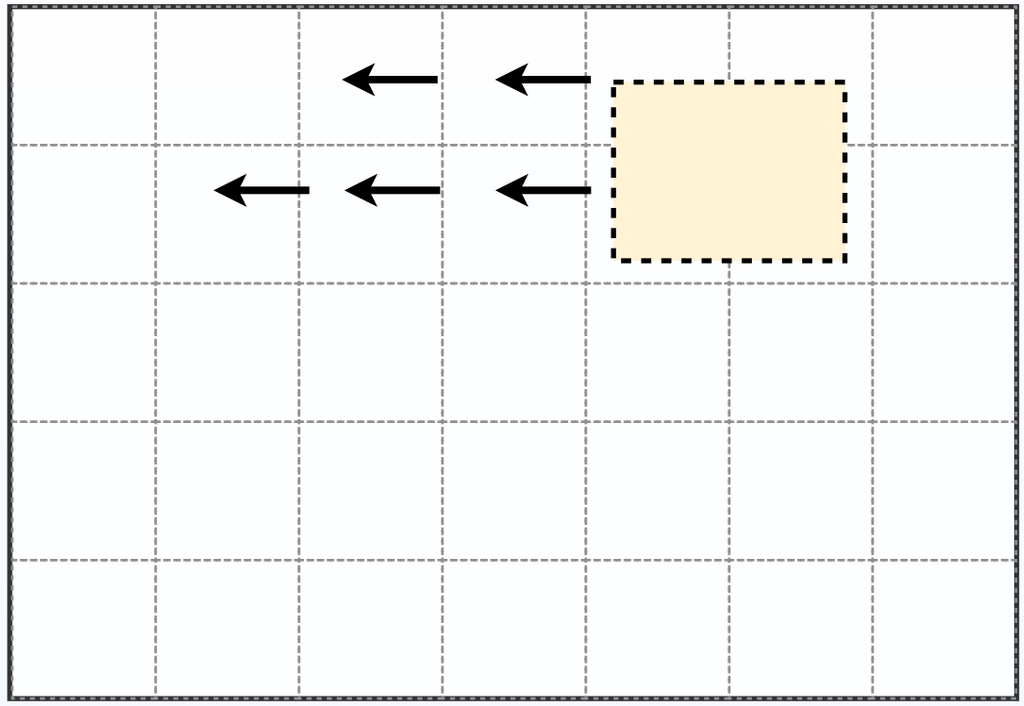
macroblocks



difference in macroblock information is compressed

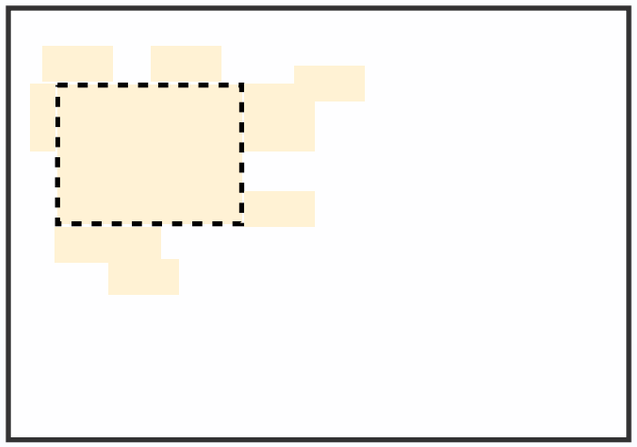


a compressed macroblock may
depend on another macroblock



need to decode the macroblocks of the region plus other macroblocks that it depends on

decode



crop

scale

scan through video and ask:

do we need to decode the current macroblock?

for each macroblock m

if m is in ROI **or**

m is needed by m' in ROI

(curr or future frames)

decode m

Questions:

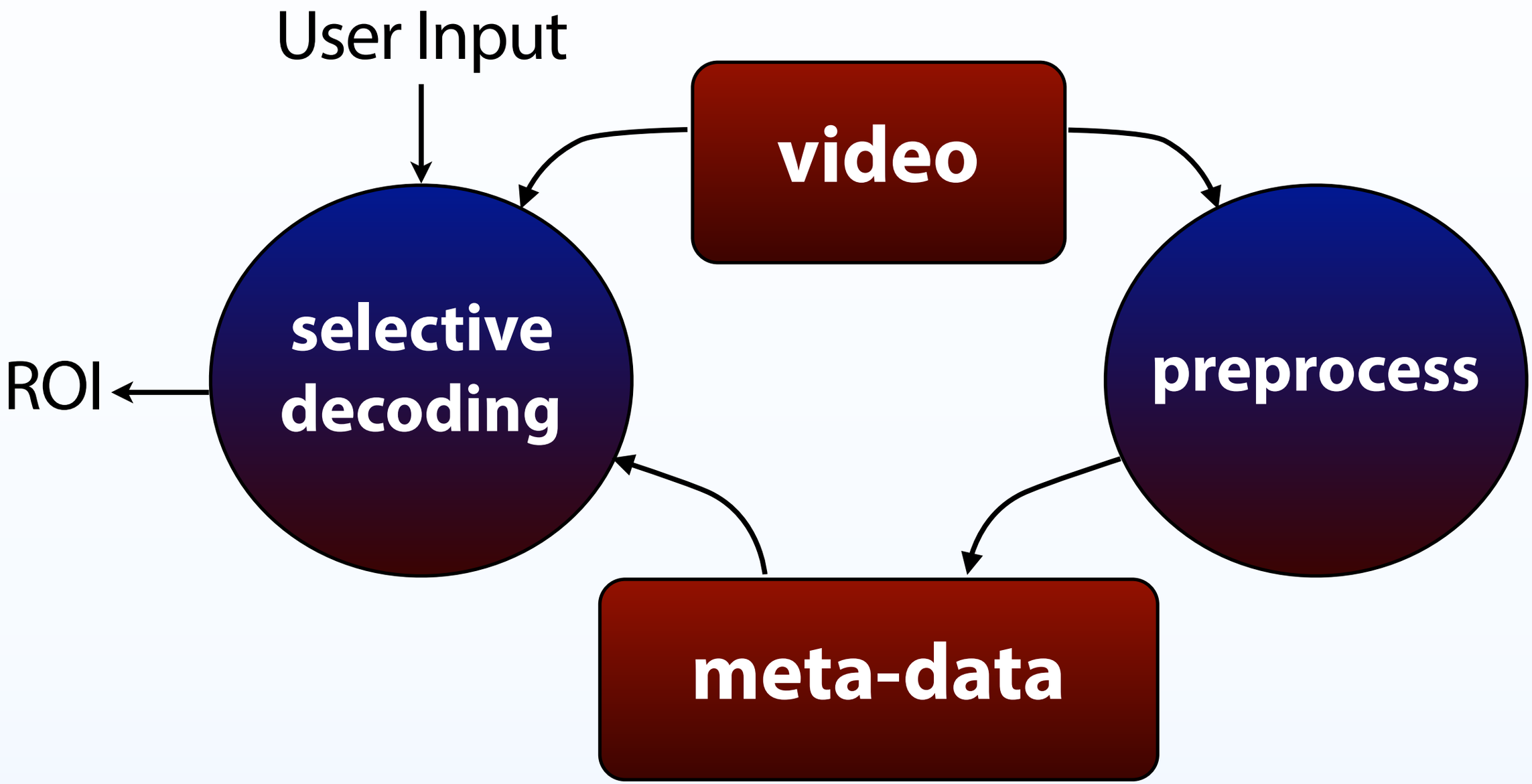
1. how to check if m is needed by m' in ROI?

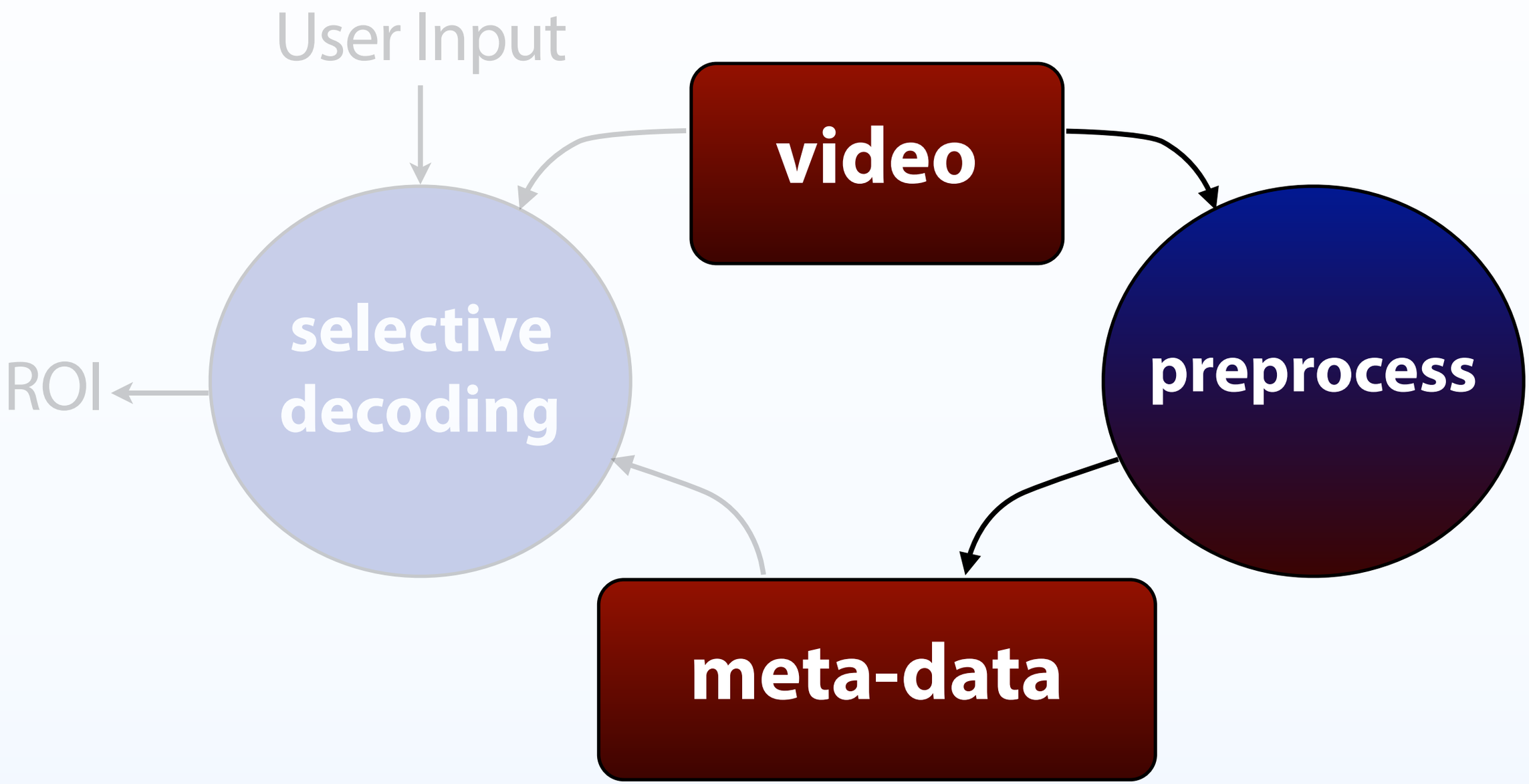
2. how to reduce the number of such m?

Requirements:

1. work with standard codec
2. no re-encoding of video

Our approach:




User Input
video
selective decoding
ROI
preprocess
meta-data

how it works with

MPEG-4 SP

(can be generalized
to other codec)

meta-data

=

MB

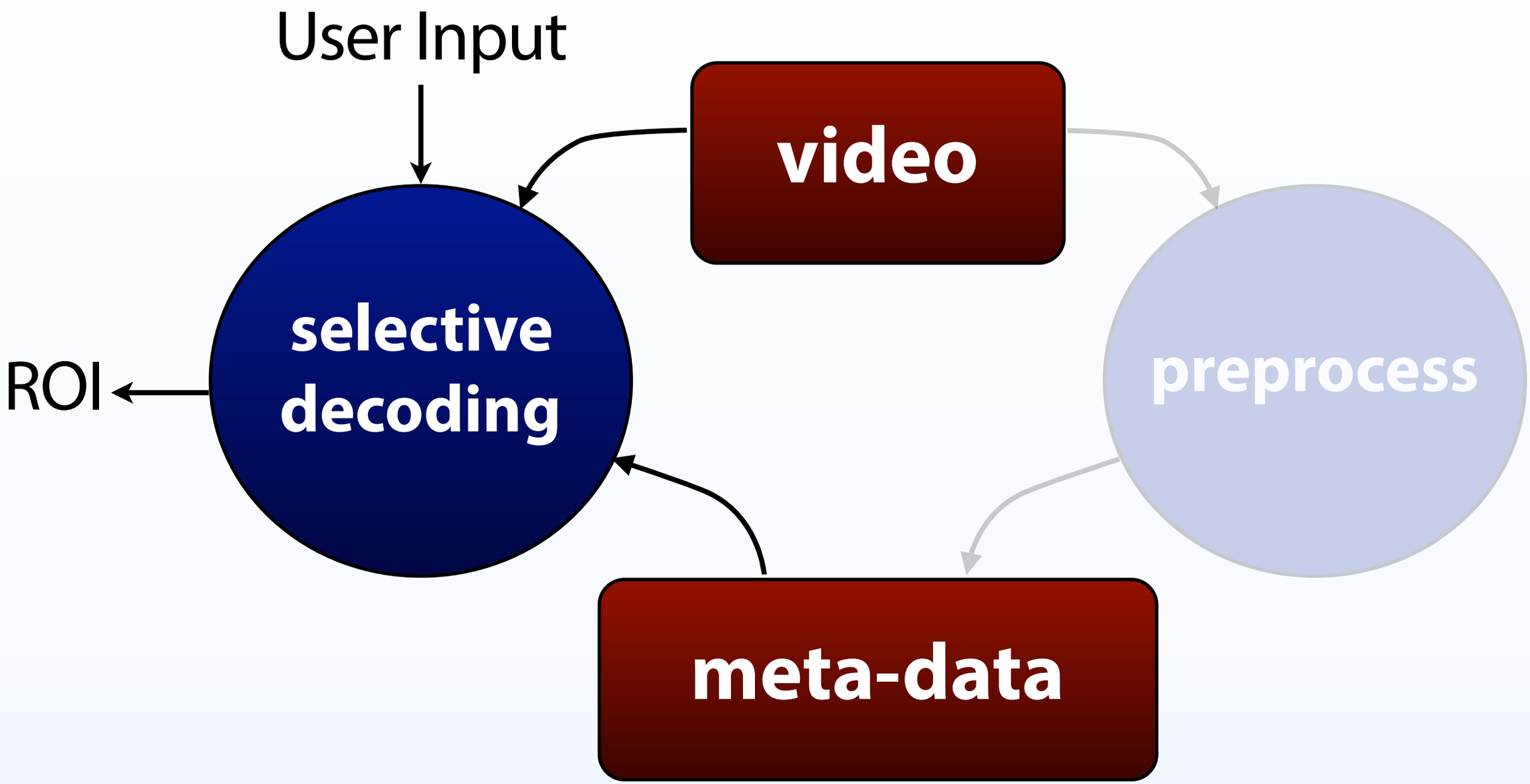
starting bit position

ending bit position

AC/DC prediction direction

MV values

Our approach:

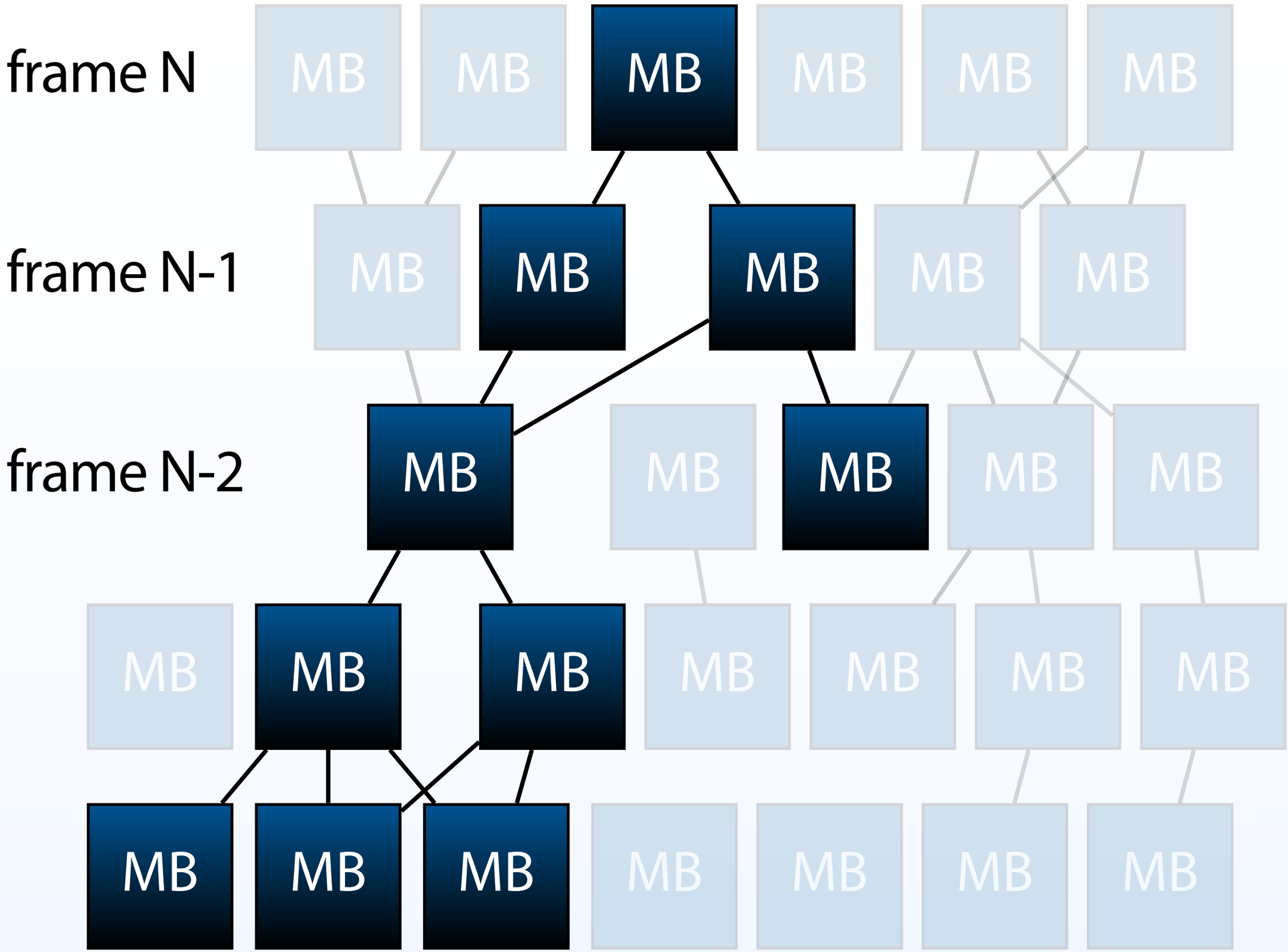


construct

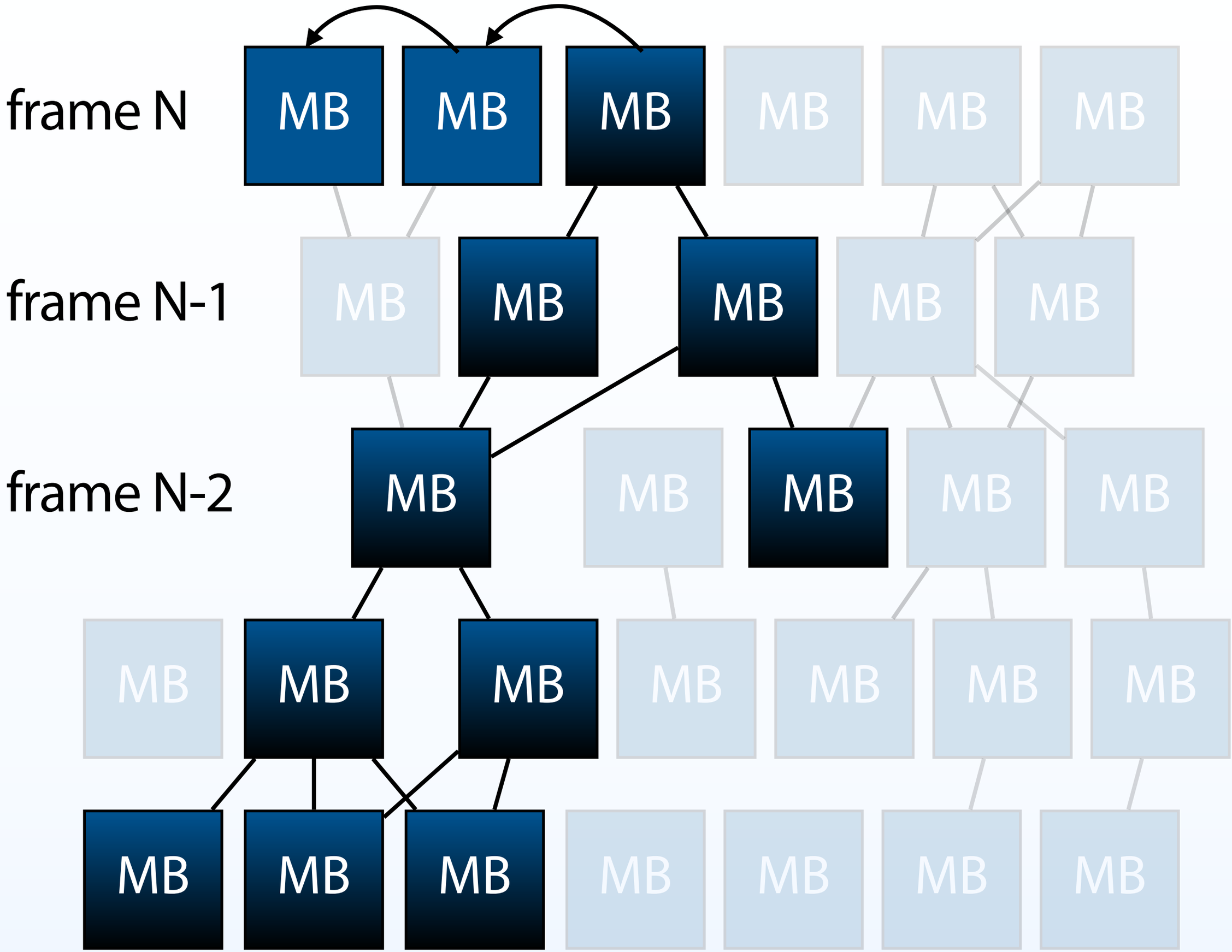
inter-frame

dependency graph

by tracing the motion


frame N
frame N-1
frame N-2
MB

construct
intra-frame
dependency graph
by tracing the AC/DC
prediction directions


frame N
frame N-1
frame N-2
MB

Questions:

1. how to check if m is needed by m' in ROI?

Answer:

lookup the data structure

Questions:

2. how to reduce the amount of dependencies?

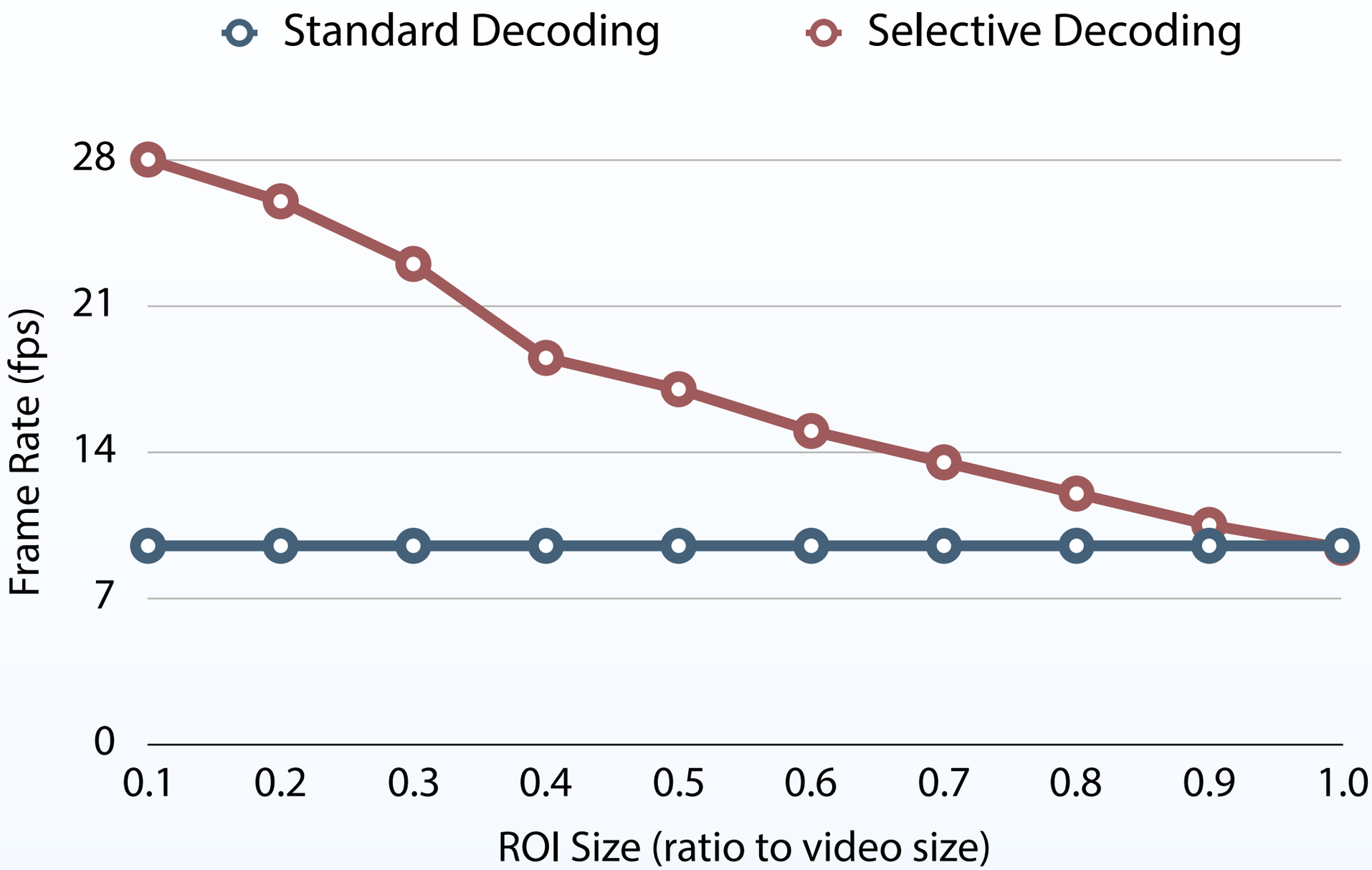
Answer:

storing AC/DC prediction directions and MV vectors

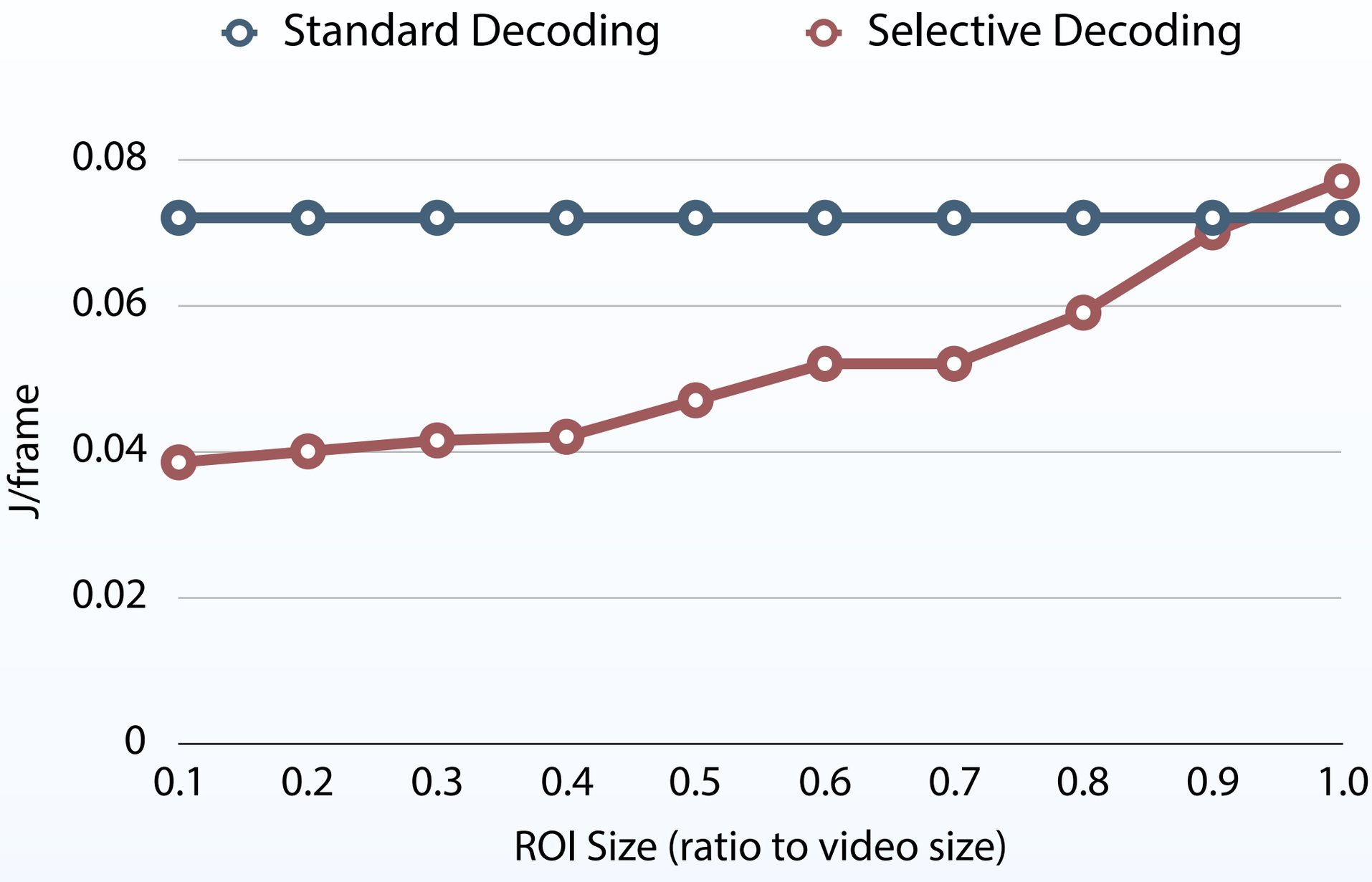
for each macroblock m
    if m is in ROI **or**
    m is needed by m' in ROI
    (curr or future frames)
        mark m for decoding

for each marked macroblock m
    seek to m
    decode and display m

Recall: aim to save computation and power as much as possible


Standard Decoding
Selective Decoding
28
21
14
7
0
Frame Rate (fps)
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
ROI Size (ratio to video size)

CPU power consumption (by PowerTutor)



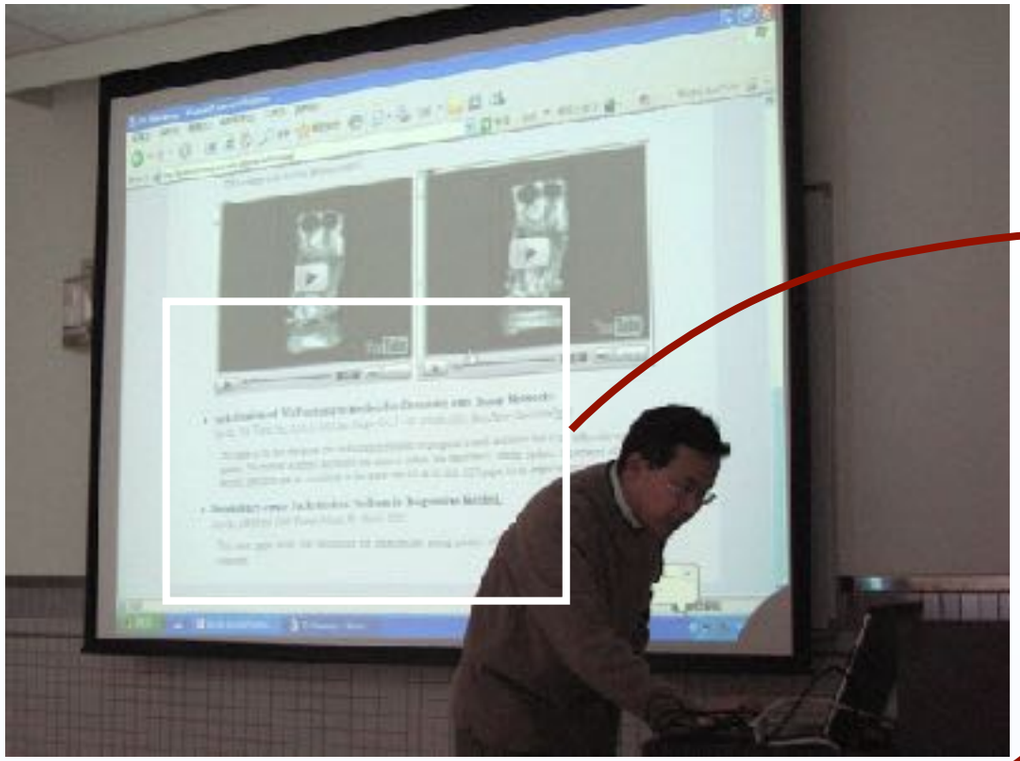
at the cost of
huge meta-data file
(up to 5 times the video size)

Zoomable Video Playback on Mobile Devices by Selective Decoding

Liu Feipeng

MComp Dissertation

Scaling reduces the resolution of frame

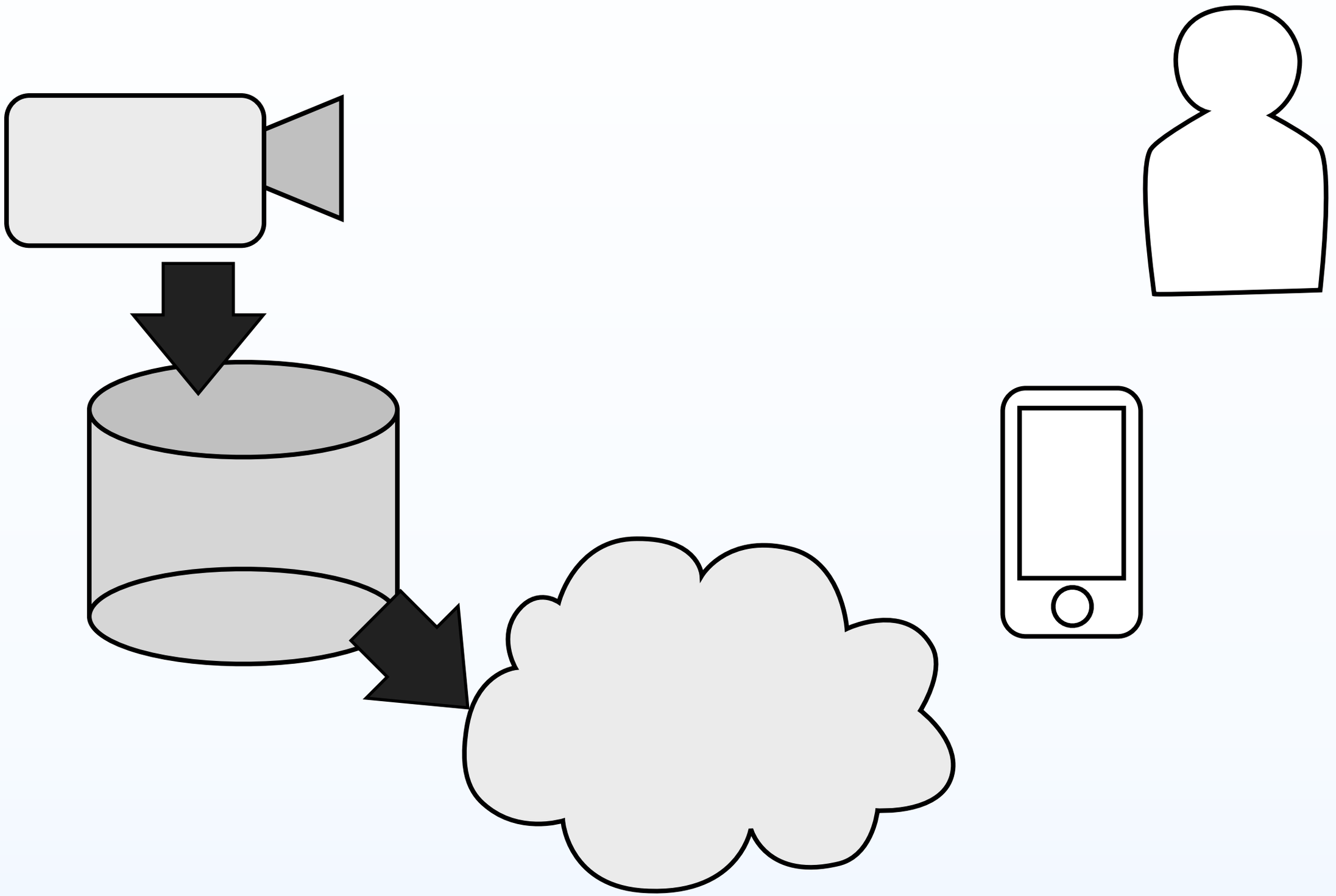


zoom

can we “zoom” on video streams?

Dynamic ROI Cropping

(which is not supported by standard video codec)

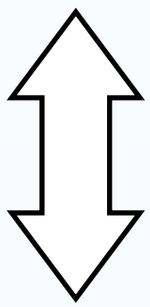


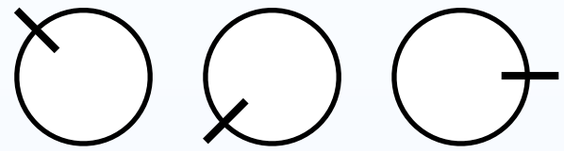
Local playback: need not decode the whole frame

Remote playback: need not send the whole frame

Video Compression

bitrate



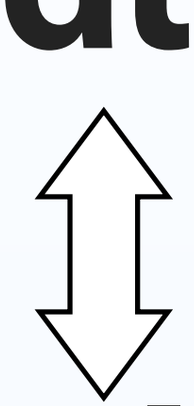


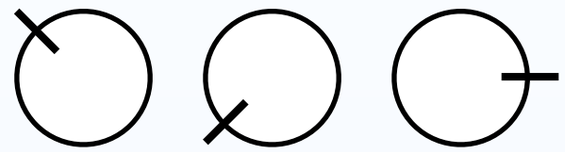
quality

Video Compression

(for zoomable video)

bandwidth of ROI

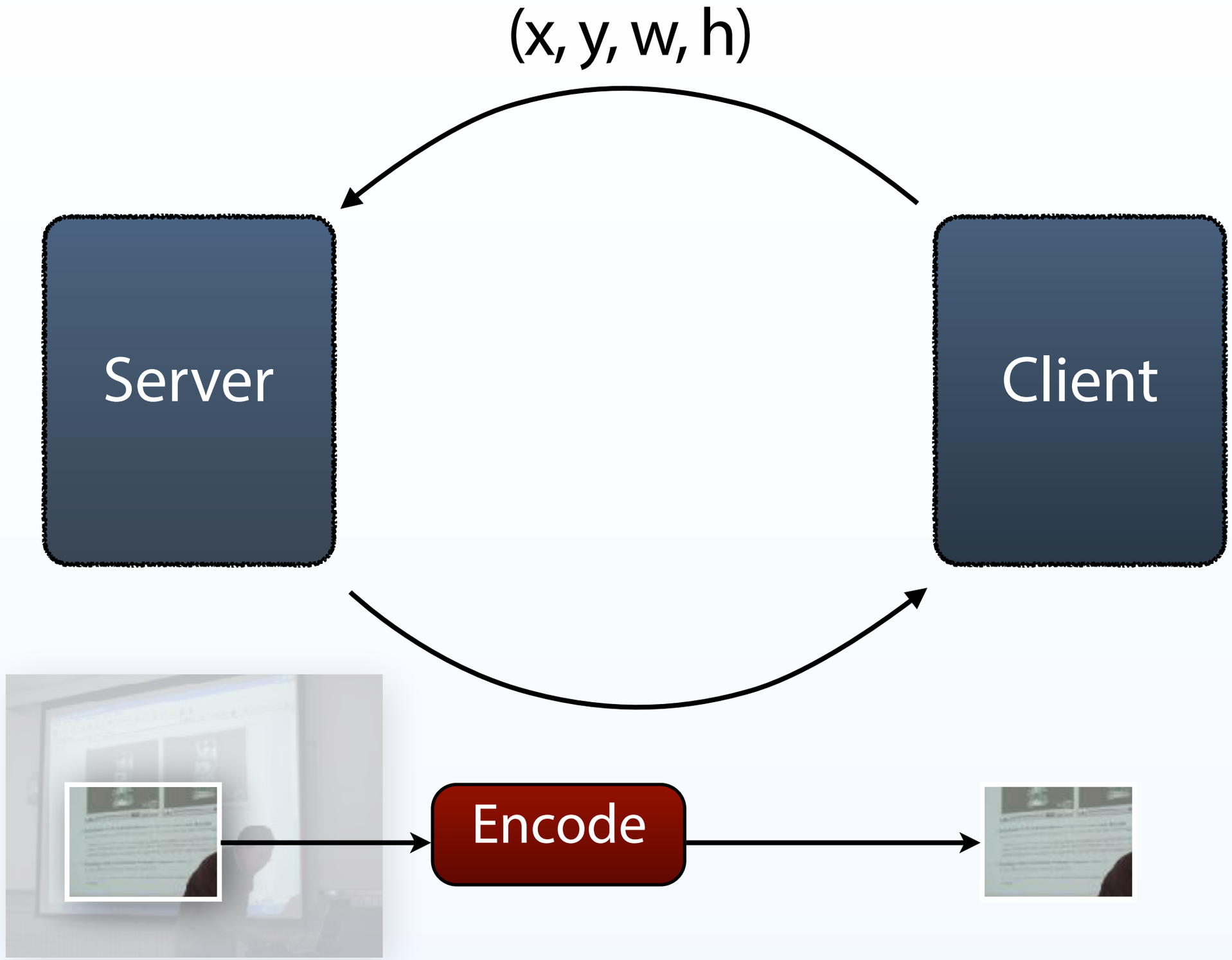


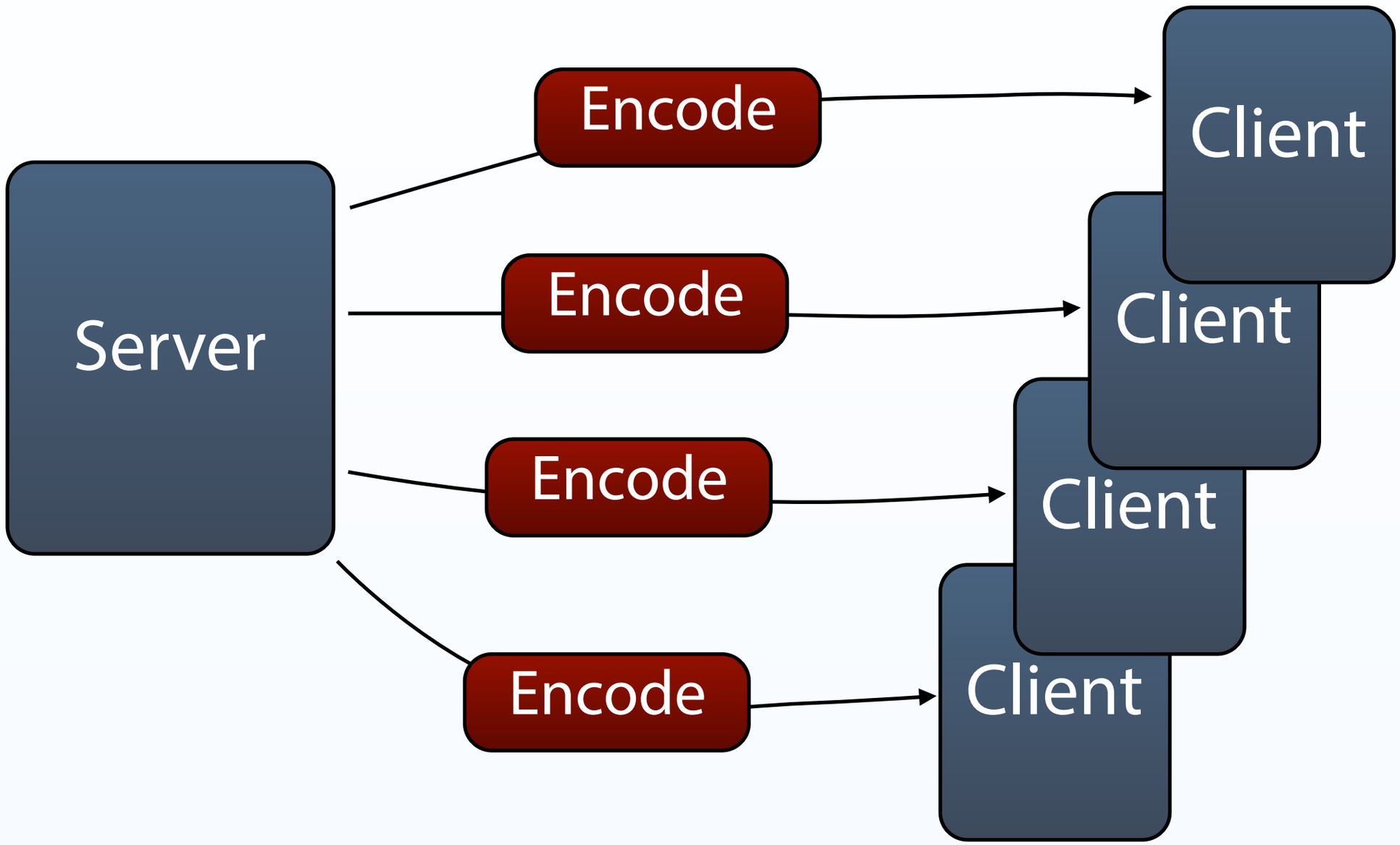


quality

Method 1:

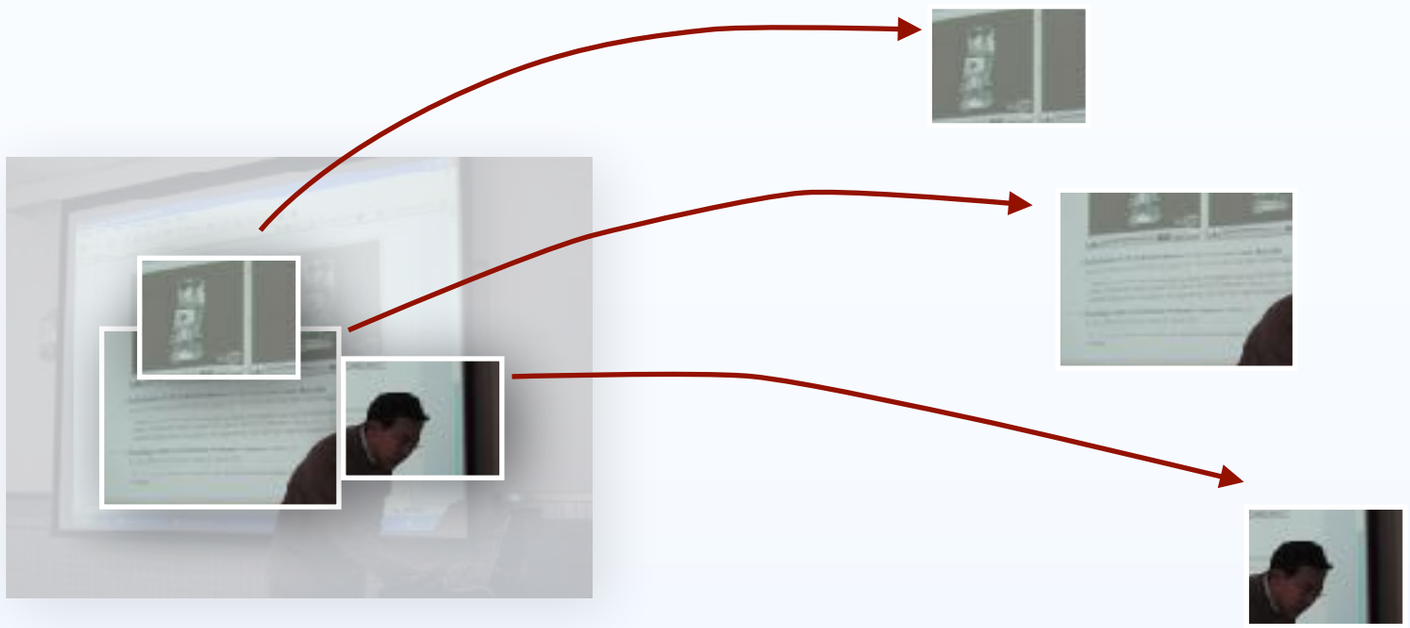
Dynamic Encoding


(x, y, w, h)
Server
Client
Encode



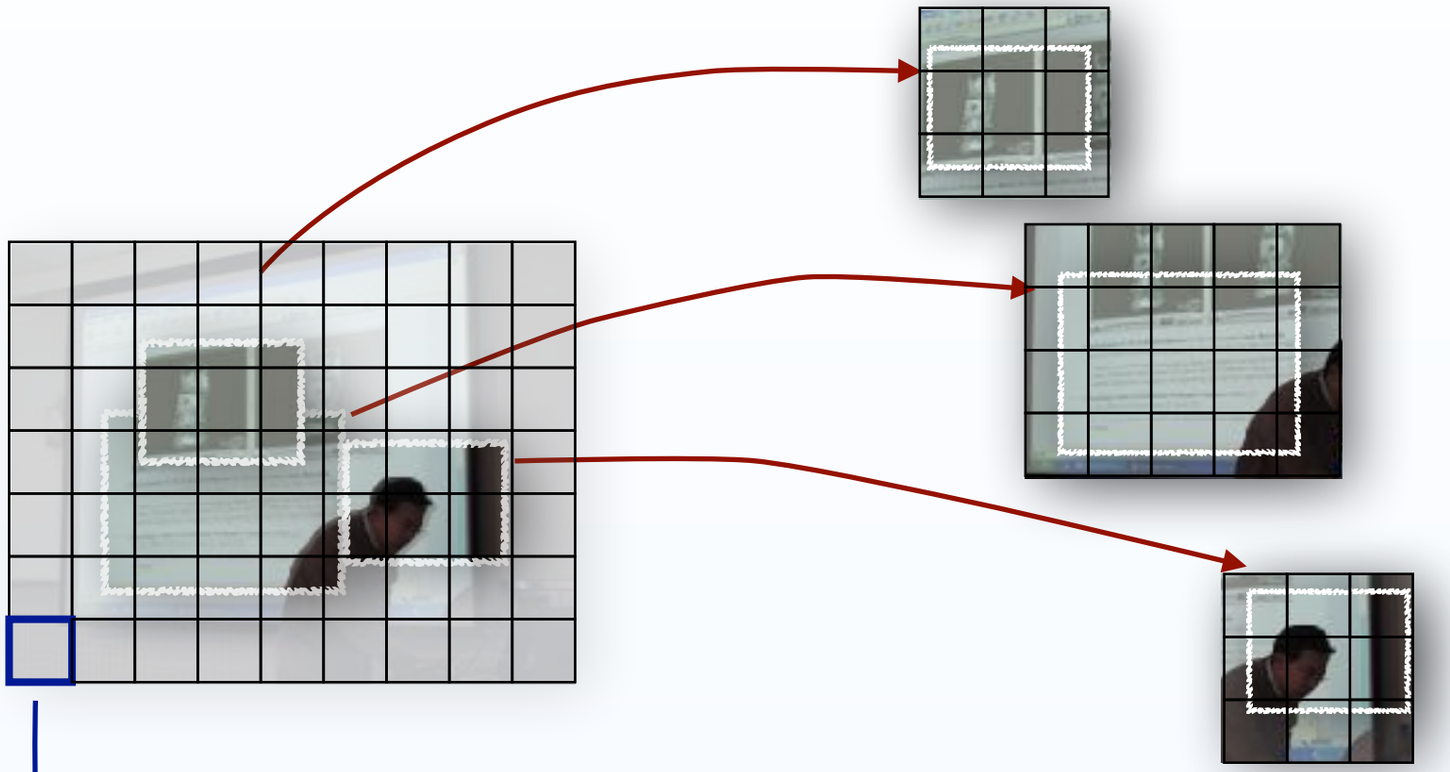
Not scalable

encode once
multiple ROIs



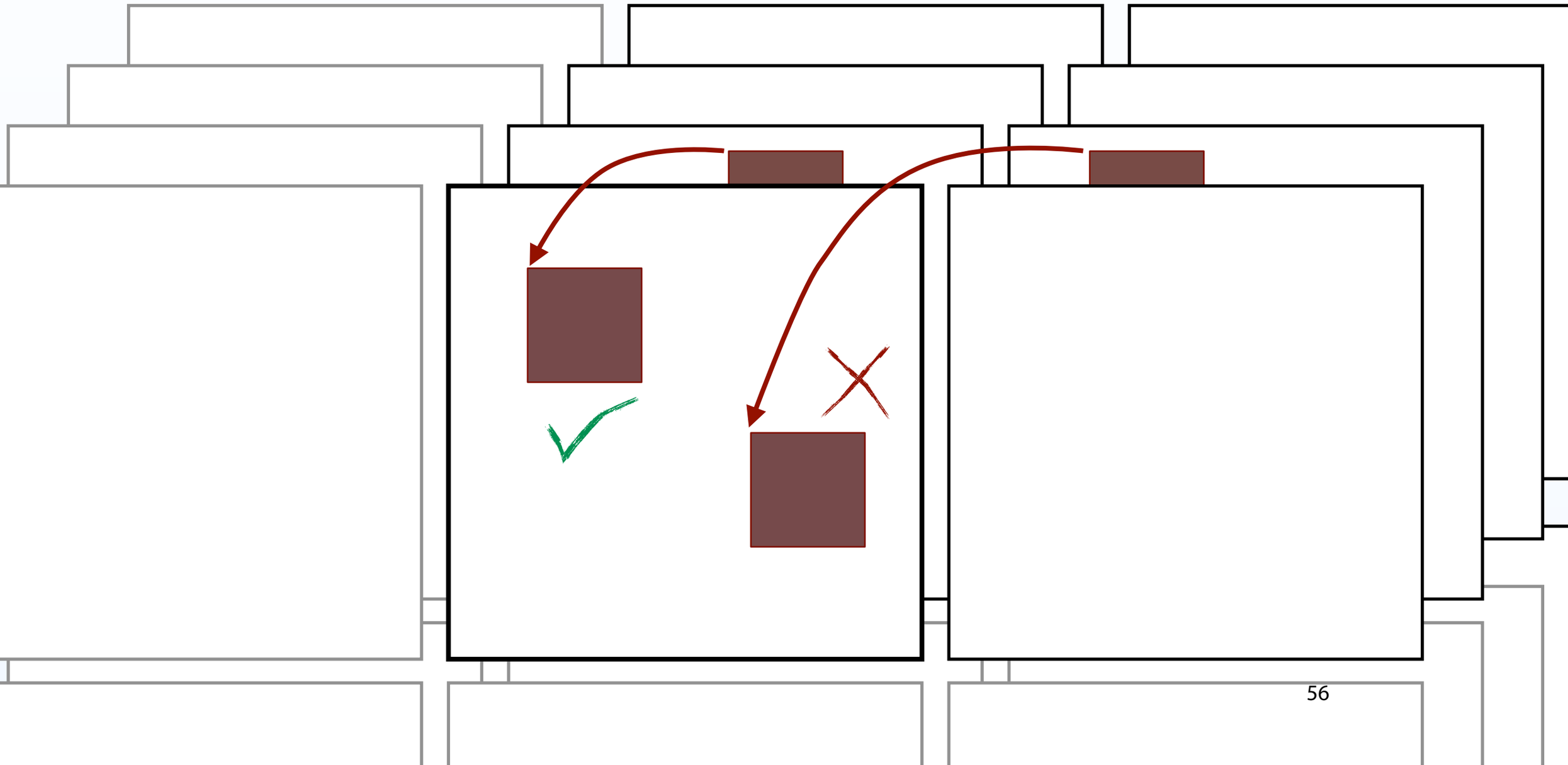
Method II:

Tiled Streams

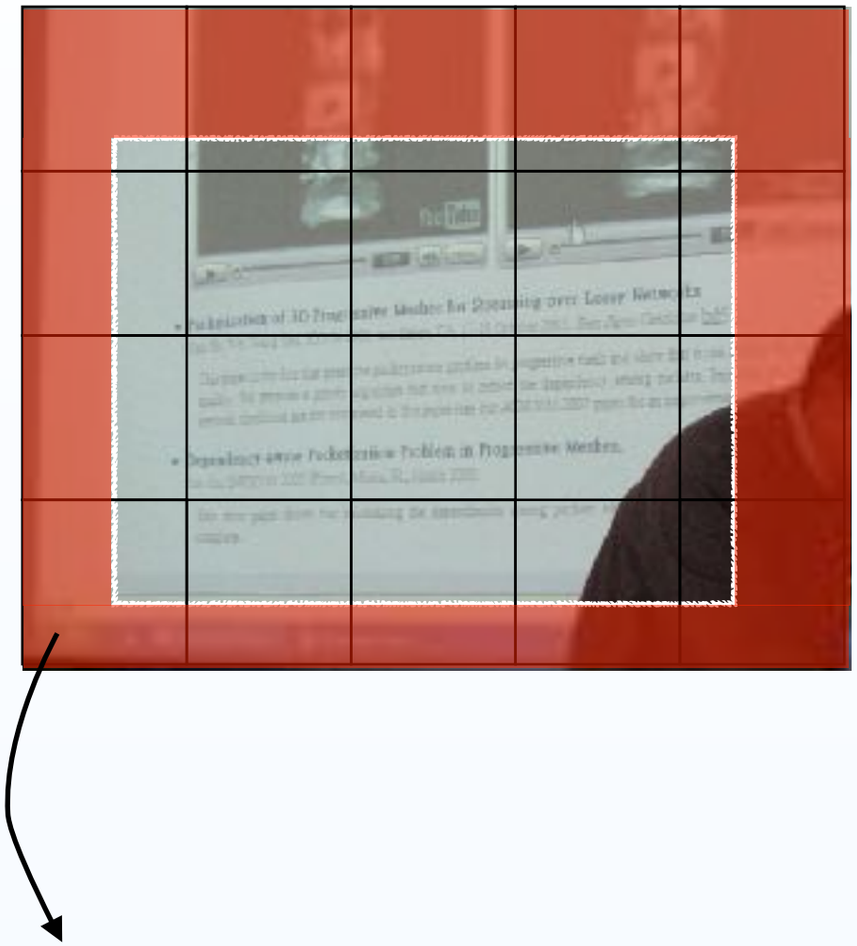


1 tile = *k* x *k* macroblocks

Motion vectors are constrained within a tile.
Each tile stream is independently decodable.



Big tile or small tile?



sent but not displayed (wasted bits)

bigger tile → more waste

→ more bits

smaller tile → less compression

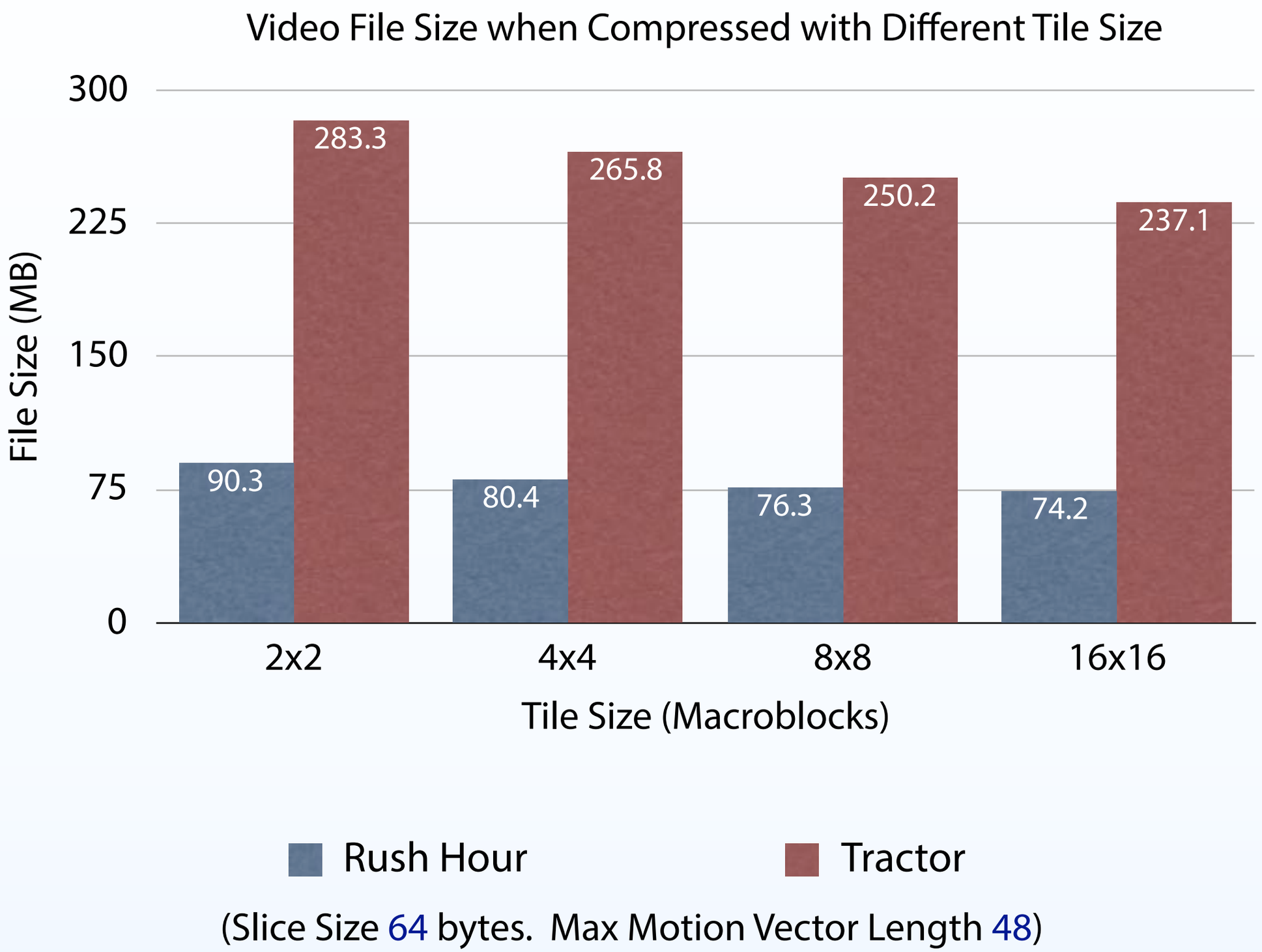
→ more bits

Evaluation of Tiled Streams

Standard test video
GOP size 7 (IPBBPBB)
Constant quality

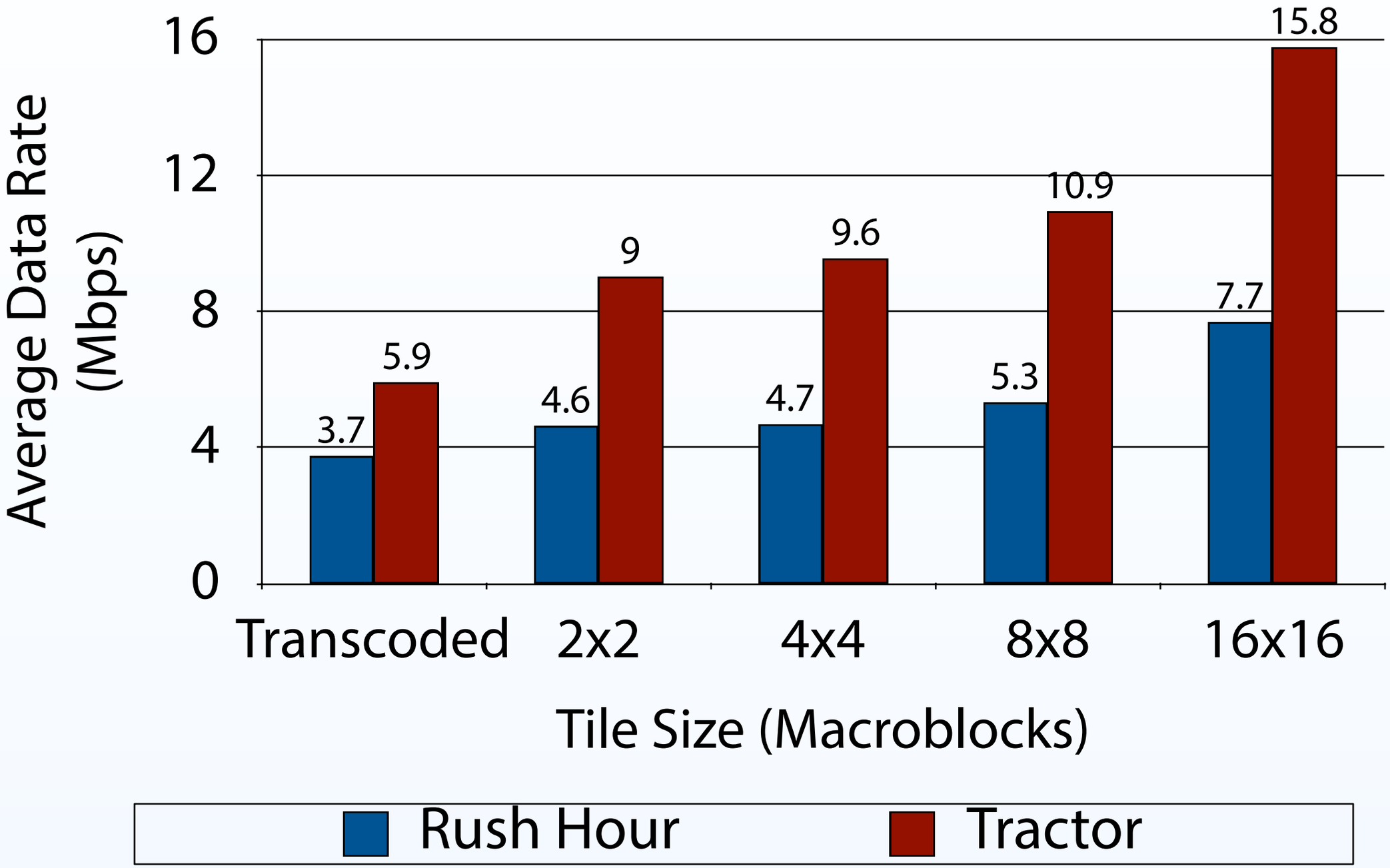
Rush Hour

Tractor



(Slice Size 64 bytes. Max Motion Vector Length 48)

Average Data Rate When Transmitted ROI of 30x30 Macroblocks



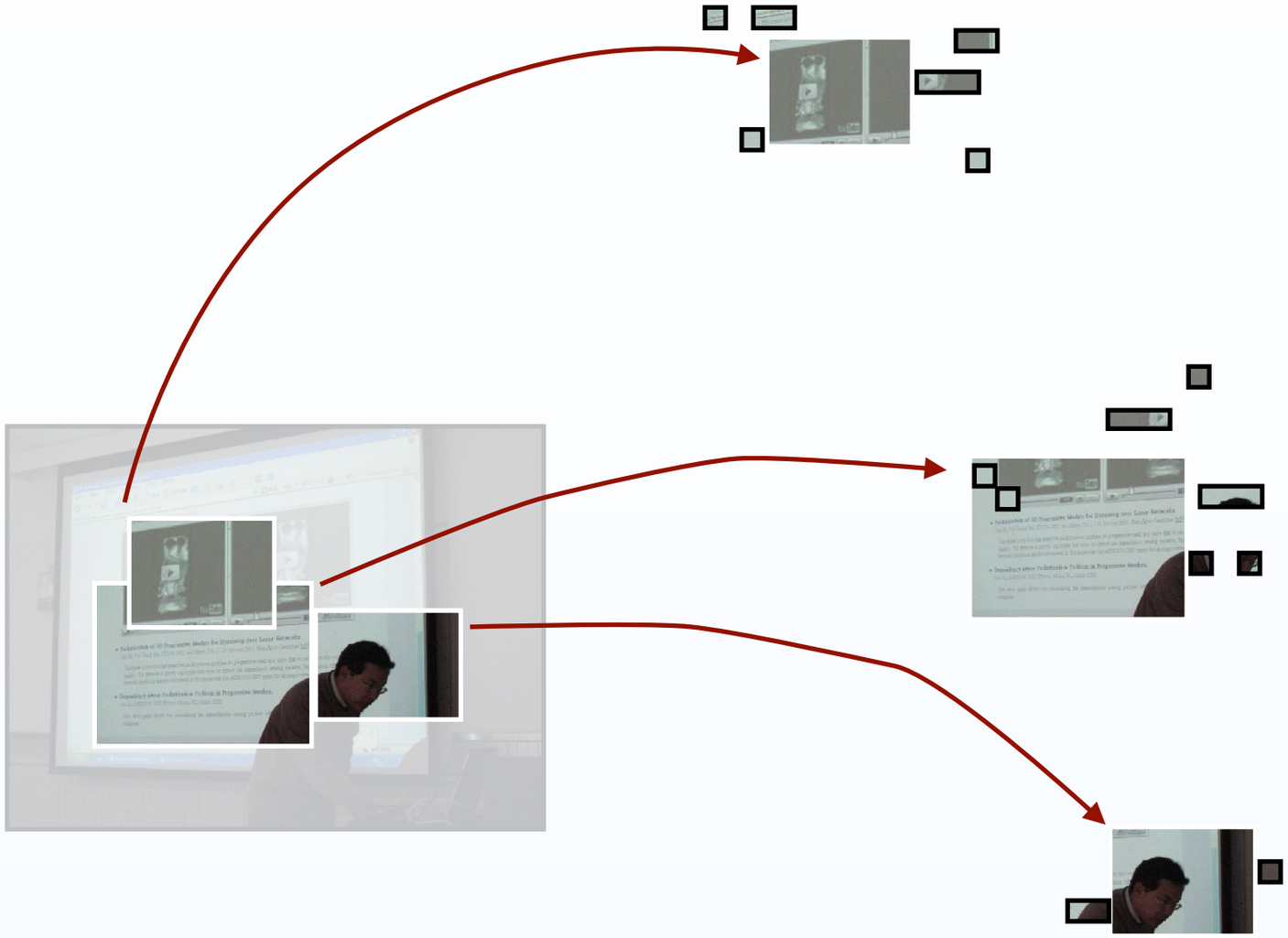
(Slice Size 64 bytes. Max Motion Vector Length 48)

Gain in compression is less significant than lost in wasted bits

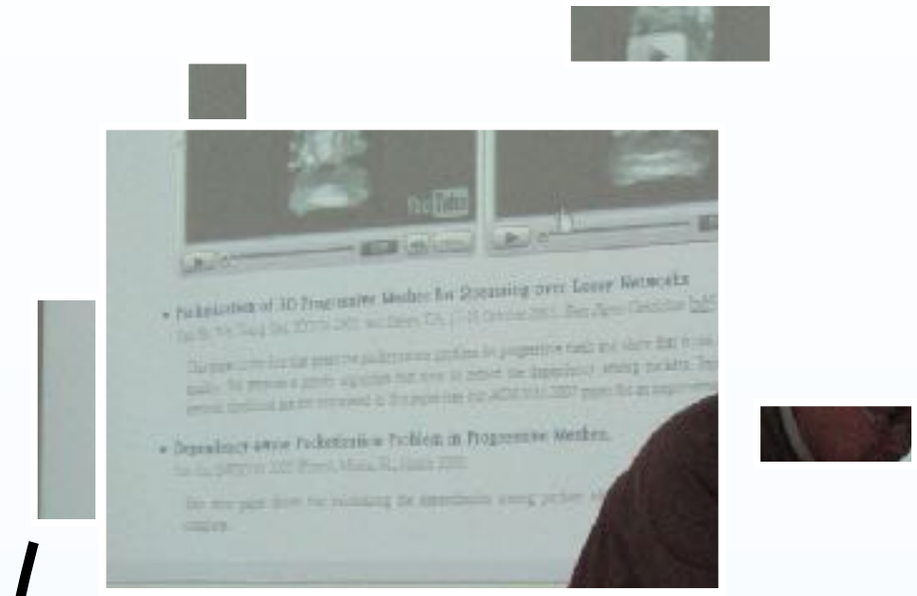
Can we reduce wasted bits?

Method III:

Monolithic Stream

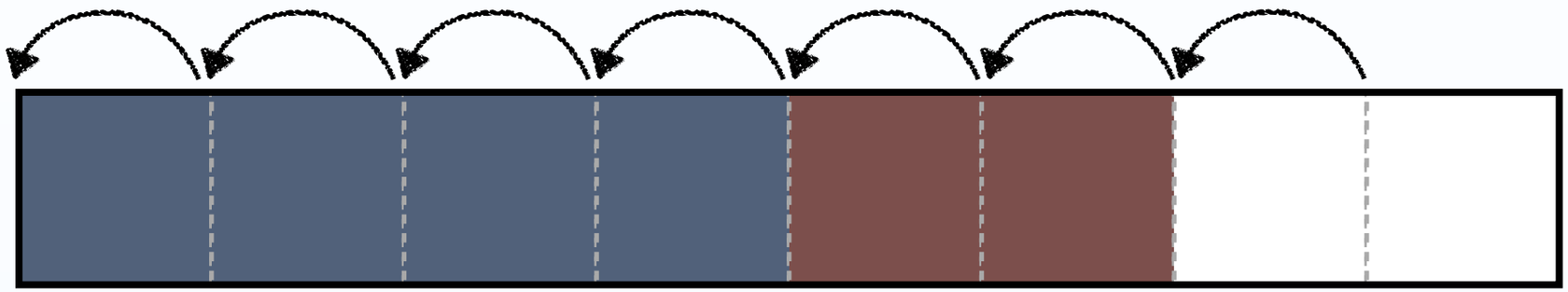


send the ROI, plus any extra bits needed to decode the pixels within ROI



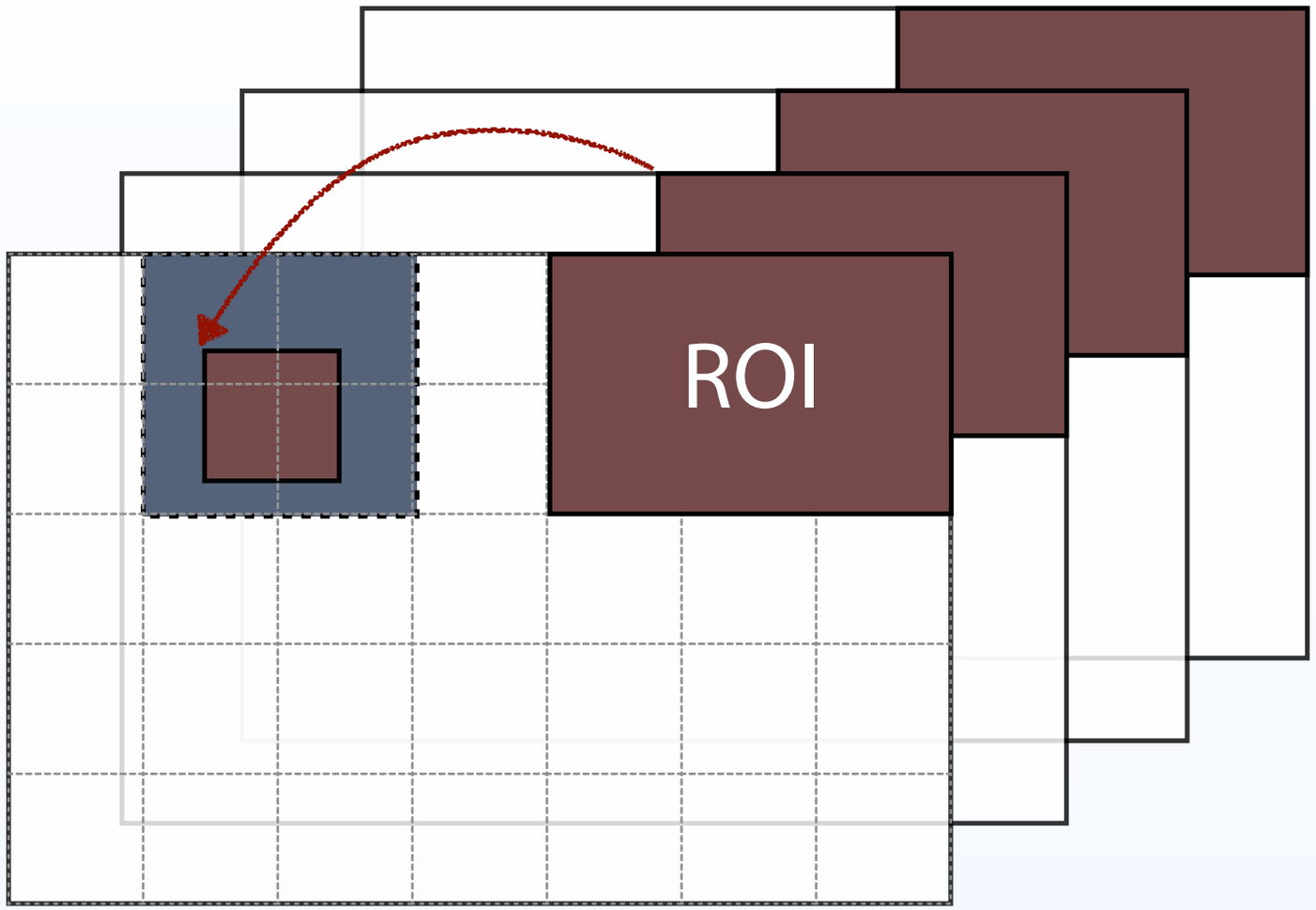
some macroblocks
within ROI depends on these

VLC dependency in a slice

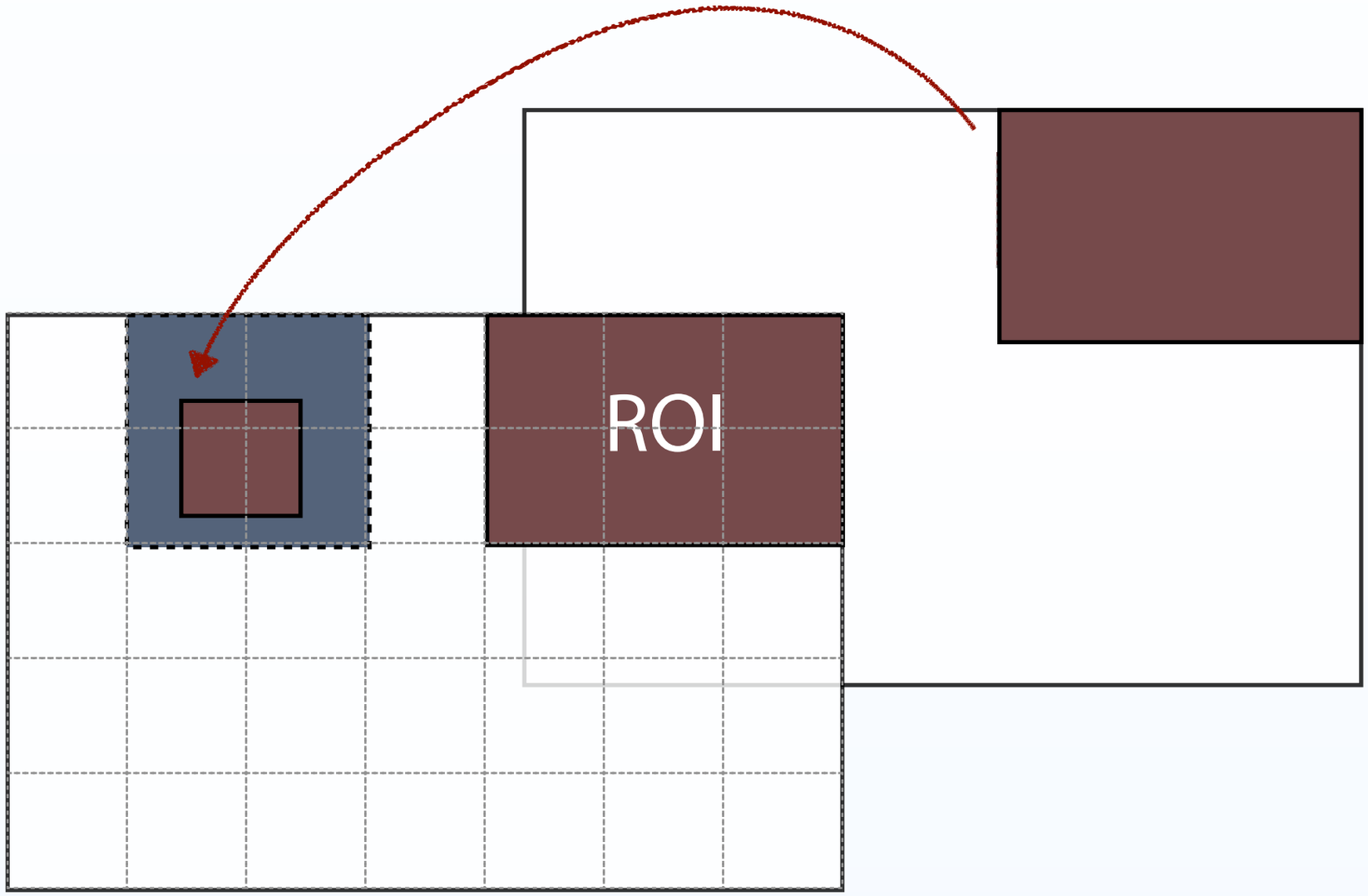


Within a slice, preceeding macroblocks ■ need to be parsed to access macroblocks in the middle ■ (no random access to macroblocks)

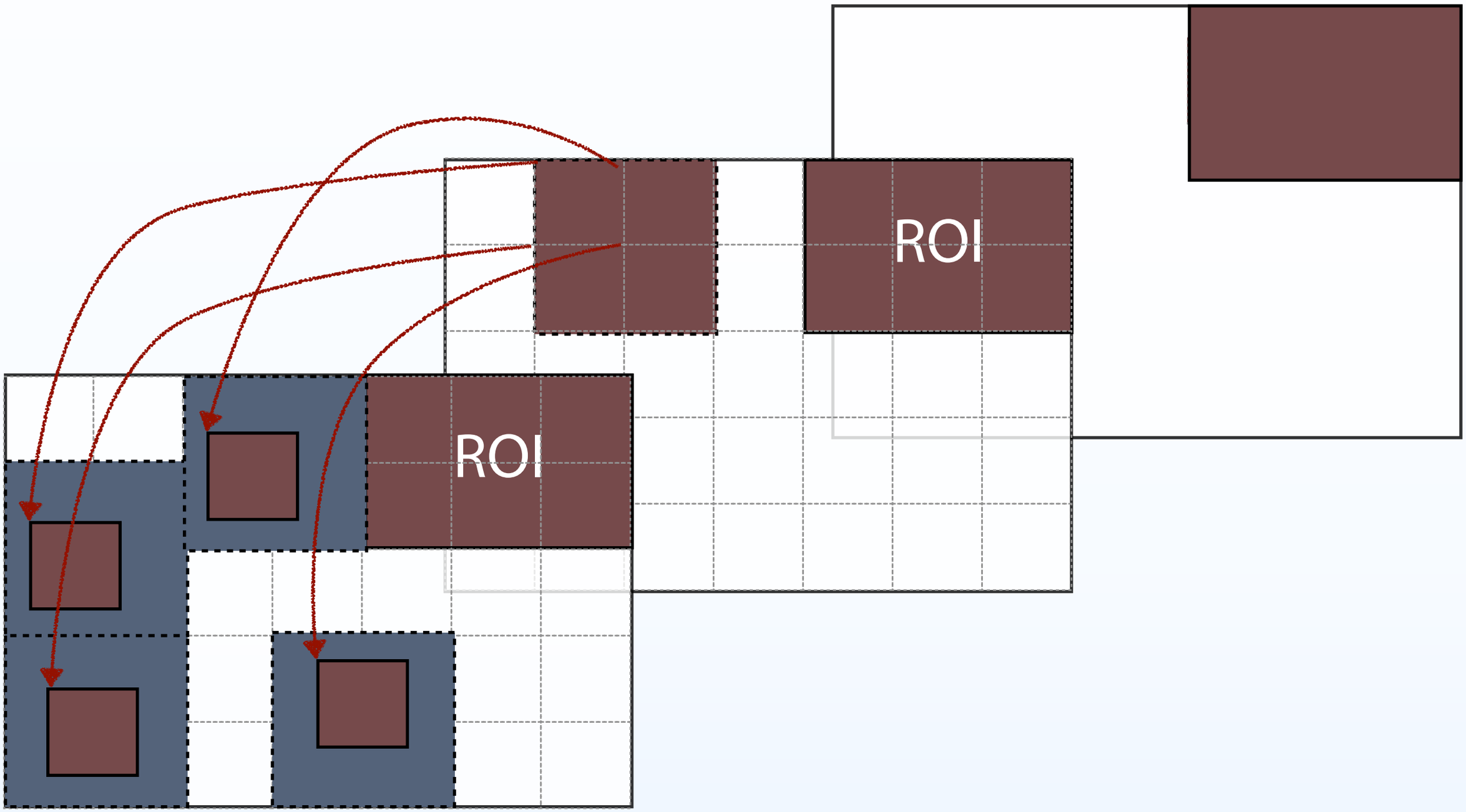
motion dependency across frames

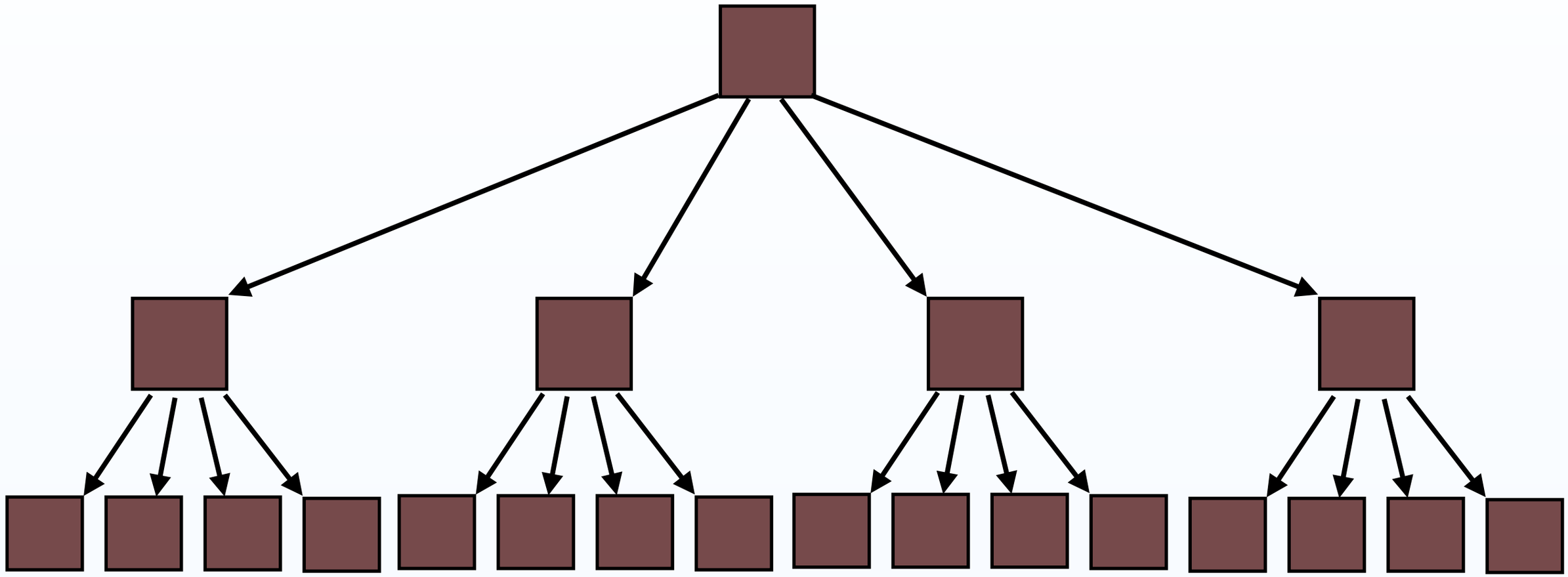


motion dependency propagates

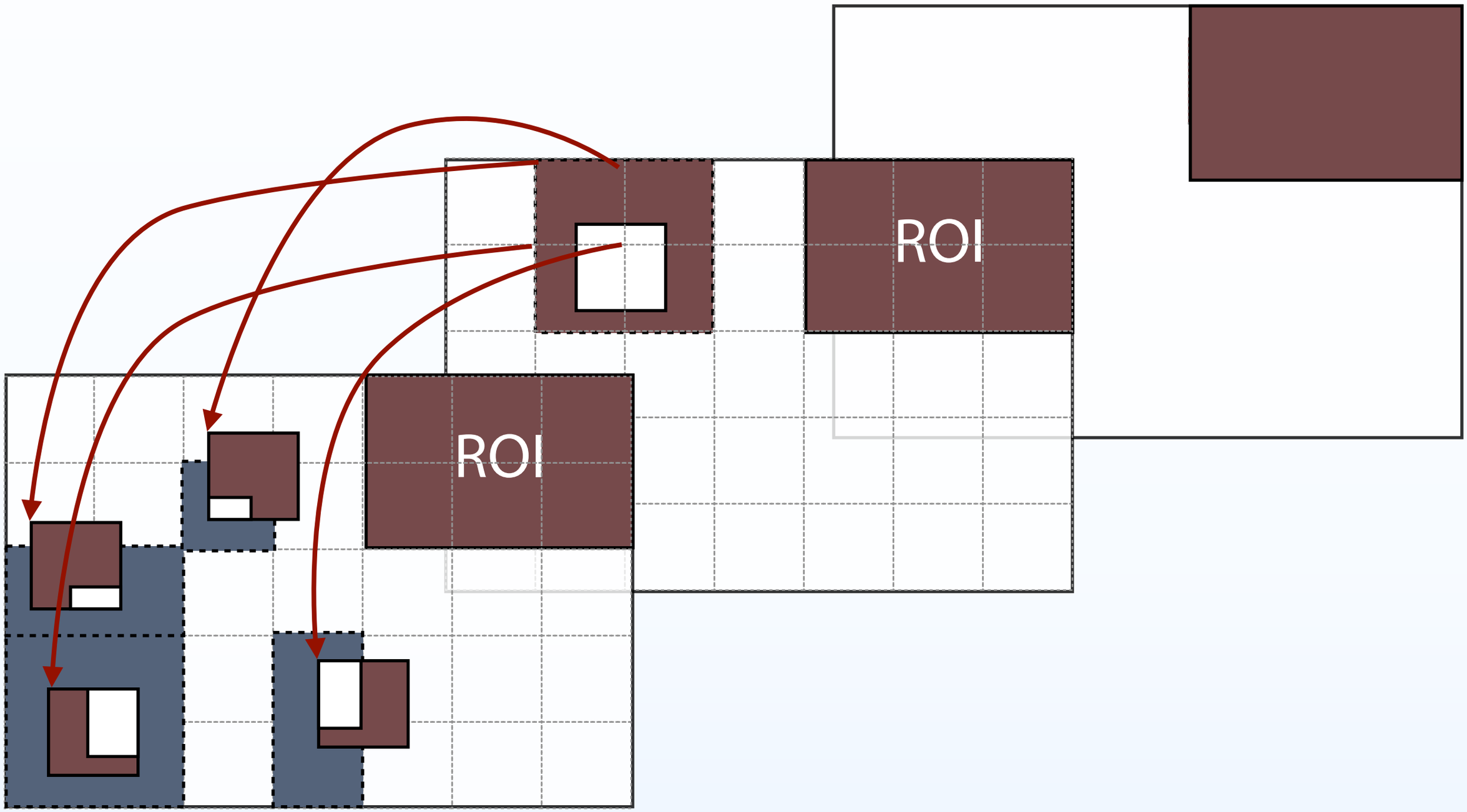


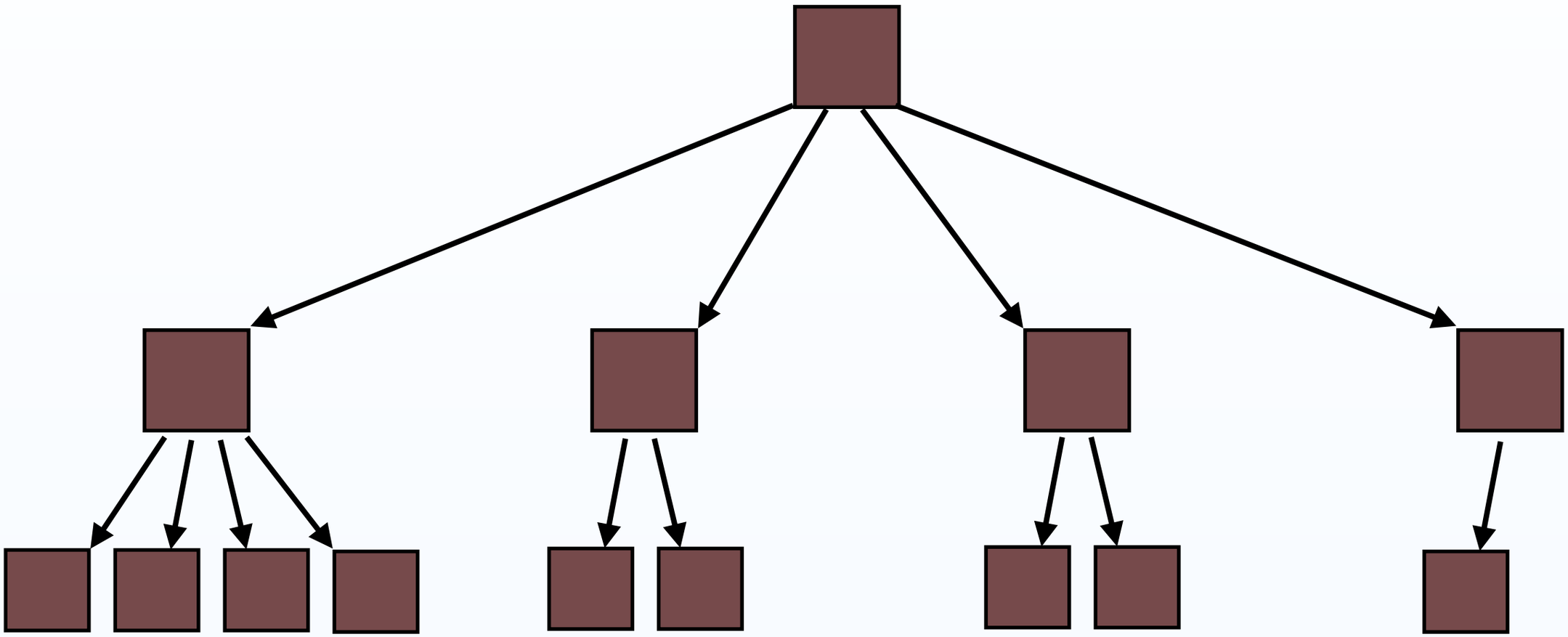
motion dependency propagates





careful optimization can reduce the dependency





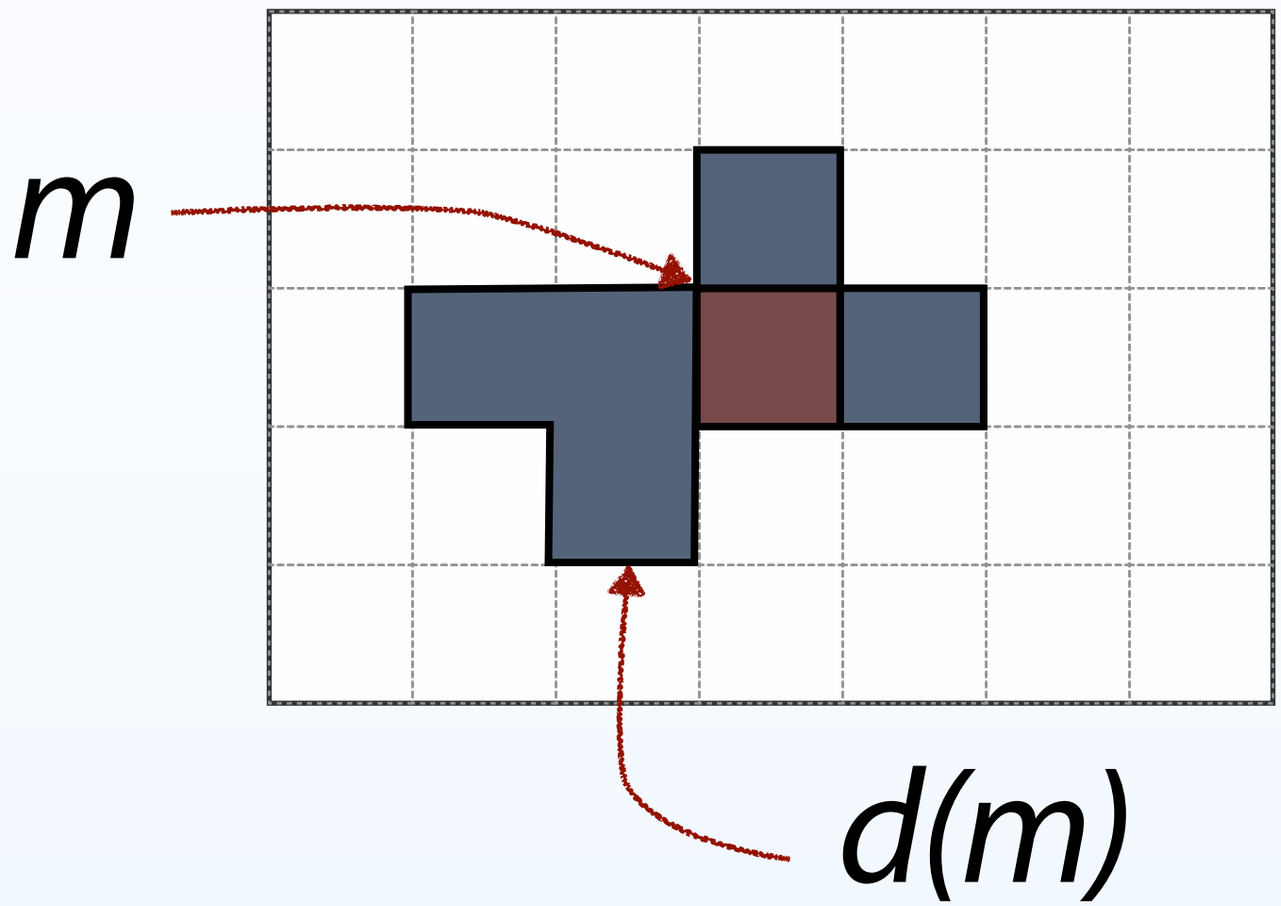
data structure: given a macroblock *m*, is there another macroblock *m'* inside the ROI that depends to *m*?

$$d(m) = \{ (x,y) \mid m' \text{ at position } (x,y) \text{ depends on } m \}$$

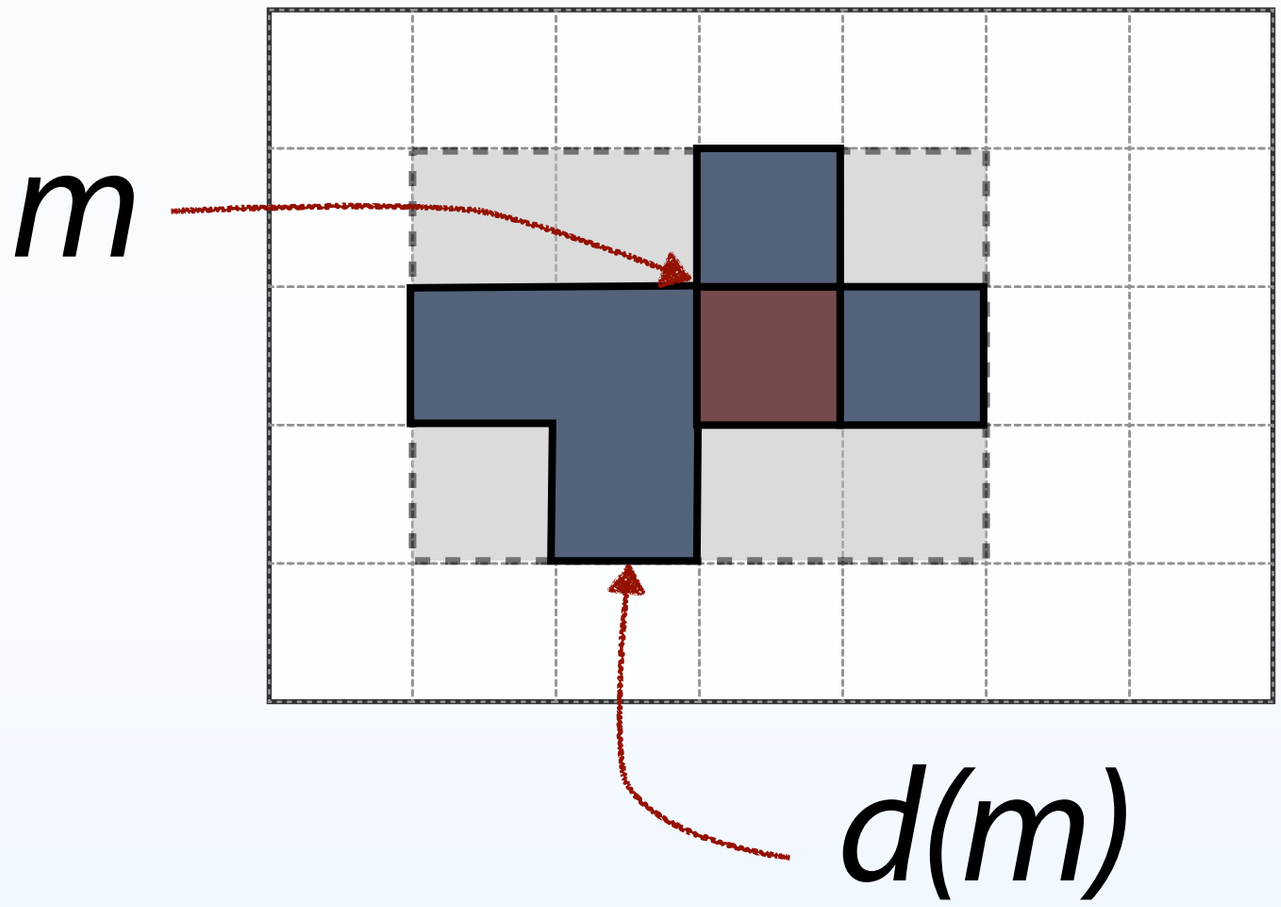
offline: compute *d(m)* for each macroblock *m* in I-Frames and P-Frames

online: send *m* if *m* is in ROI, or a position in *d(m)* is in ROI.

$$d(m) = \{ (x,y) \mid m' \text{ at position } (x,y) \text{ depends on } m \}$$



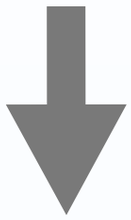
compute bounding box and only inspect d(m) if the bounding box intersects with ROI



average 0.44 ms per frame

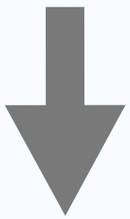
Effects of Encoding Parameters

larger slices



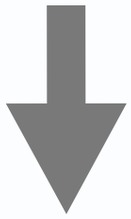
better compression,

less overhead

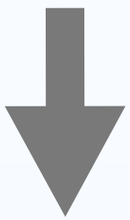


less bits

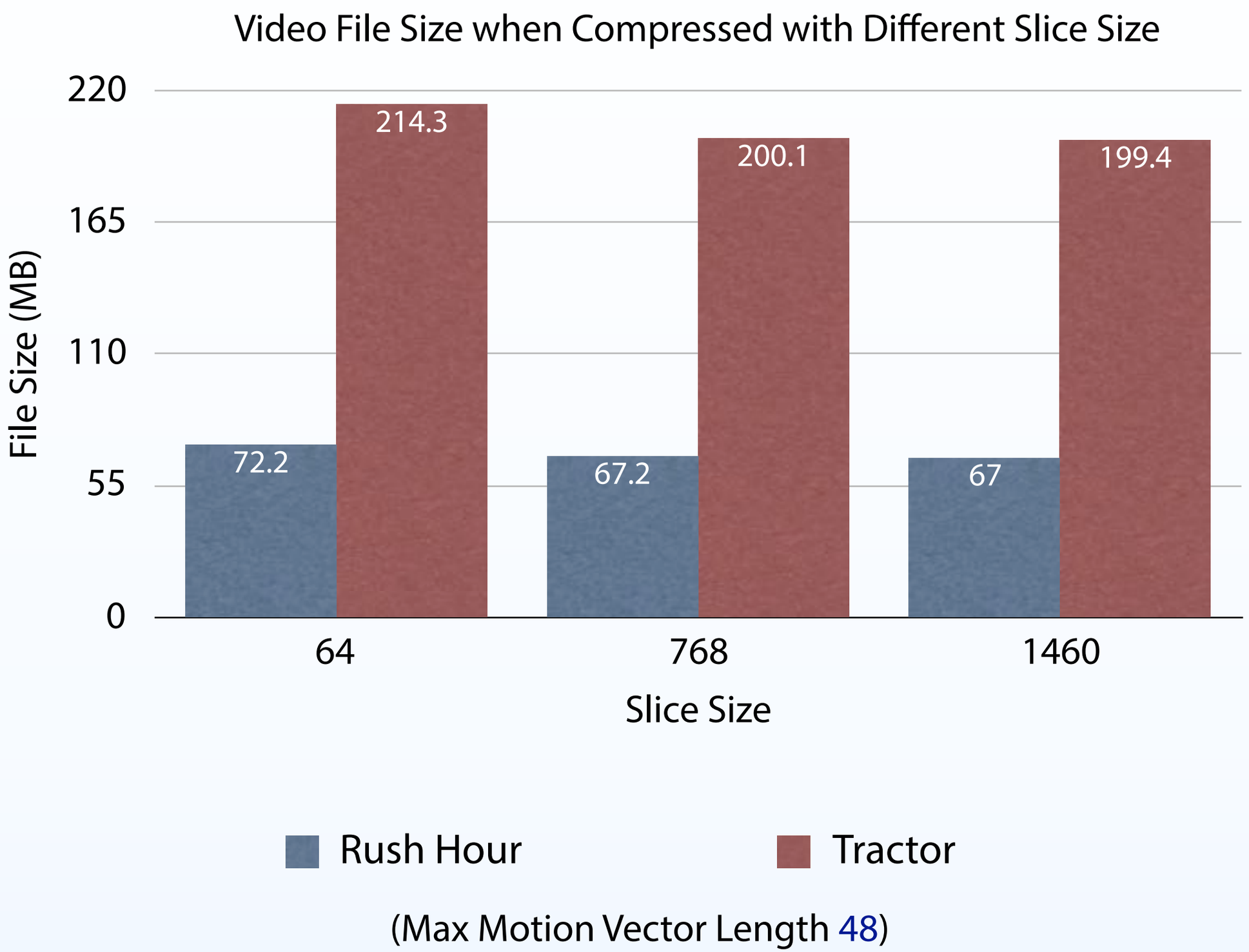
larger slices

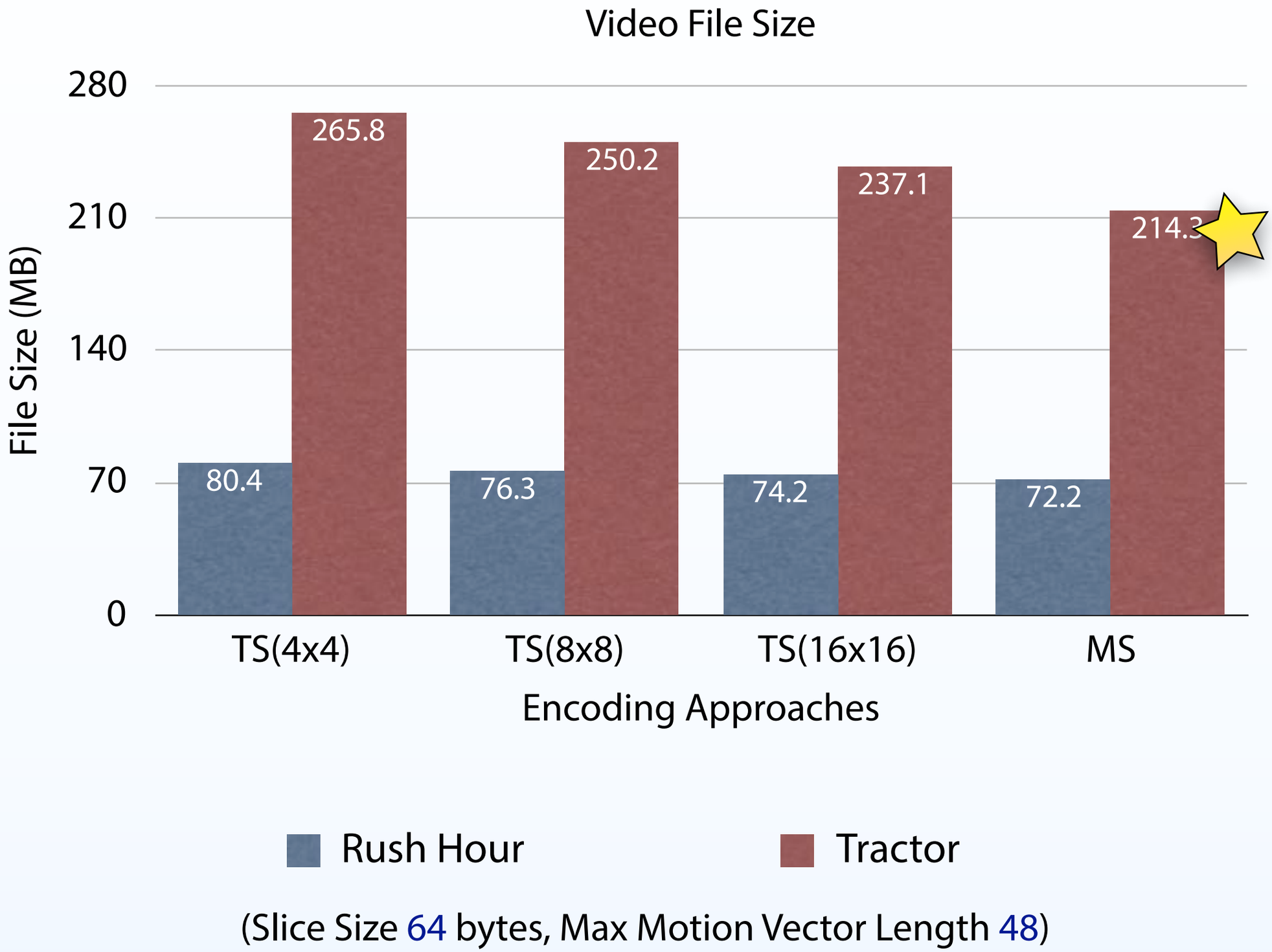


more dependency



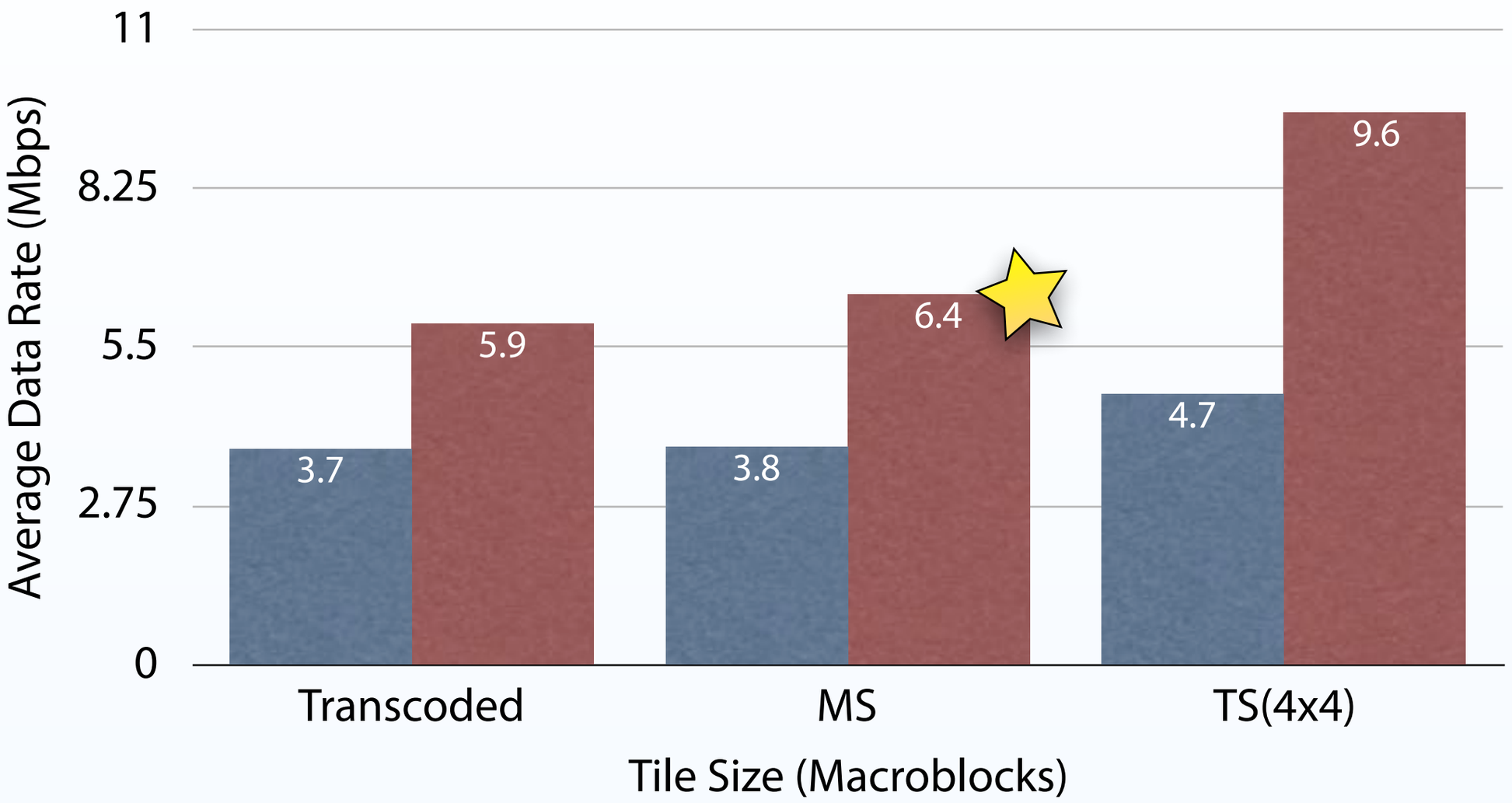
more bits


Video File Size when Compressed with Different Slice Size
220
165
110
55
0
File Size (MB)
214.3
200.1
199.4
72.2
67.2
67
64
768
1460
Slice Size
Rush Hour
Tractor
(Max Motion Vector Length 48)



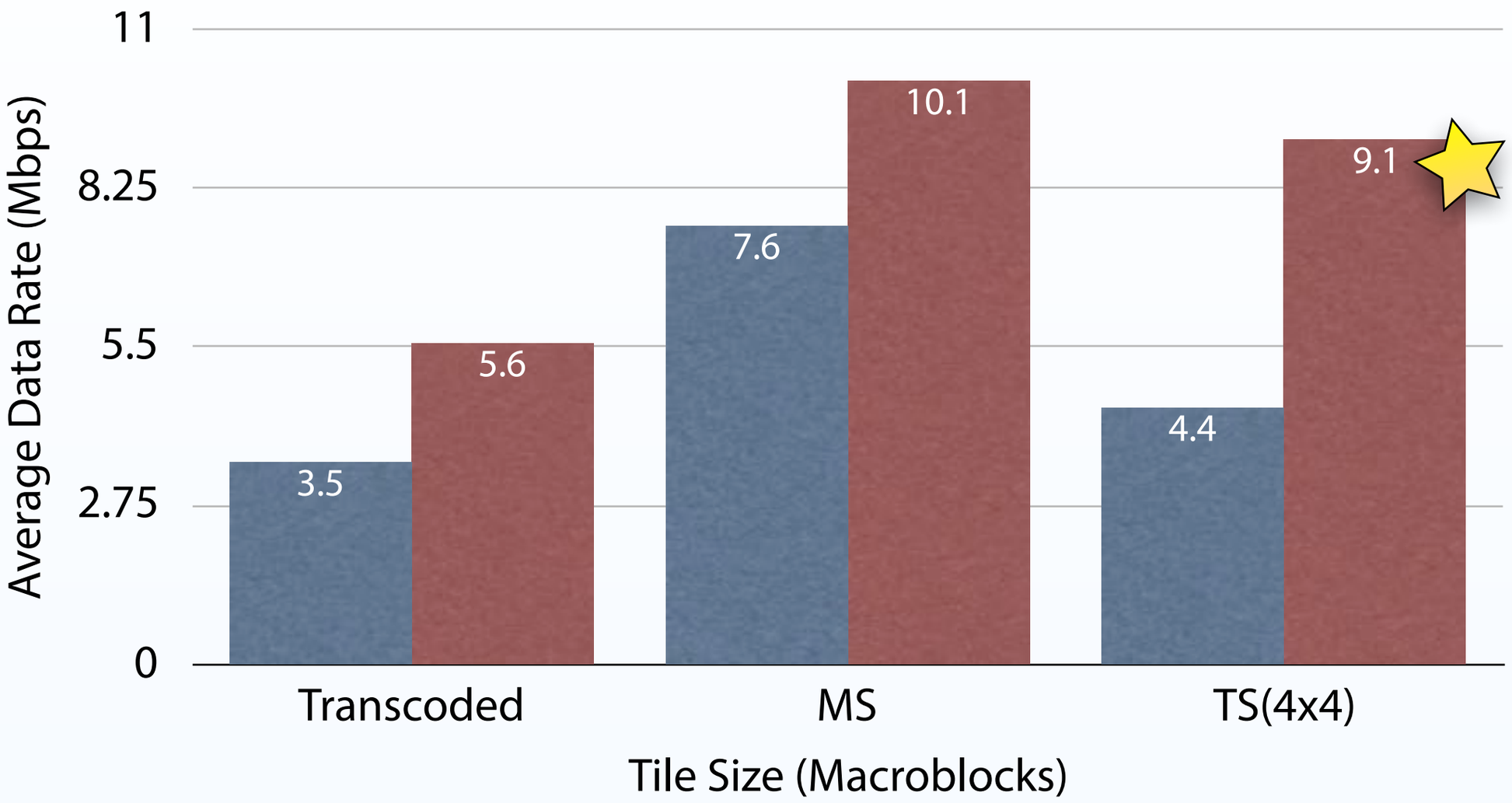
(Slice Size 64 bytes, Max Motion Vector Length 48)

Average Data Rate When Transmitted ROI of 30x30 Macroblocks



(Slice Size **64** bytes. Max Motion Vector Length 48)

Average Data Rate When Transmitted ROI of 30x30 Macroblocks



(Slice Size **1460** bytes. Max Motion Vector Length 48)

small slice size is better

longer motion vector

↓

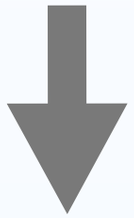
better compression

↓

less bits

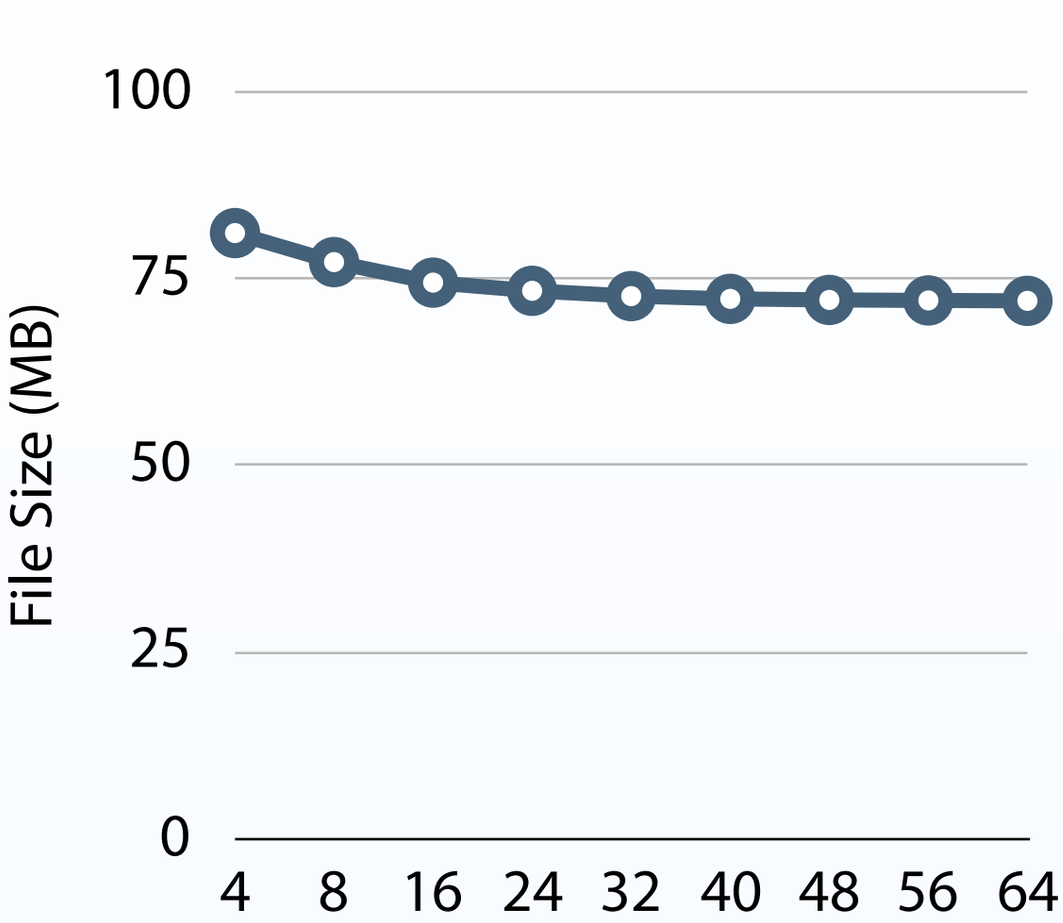
longer motion vector

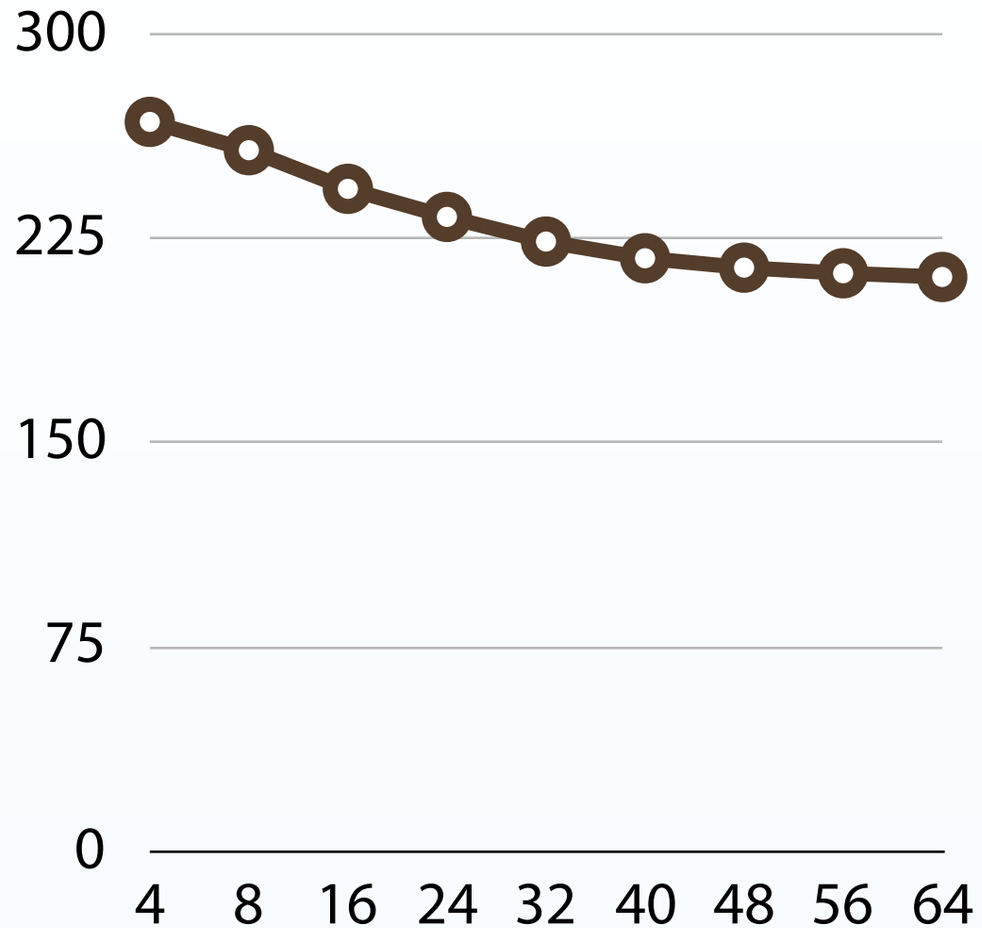
more dependency



more bits

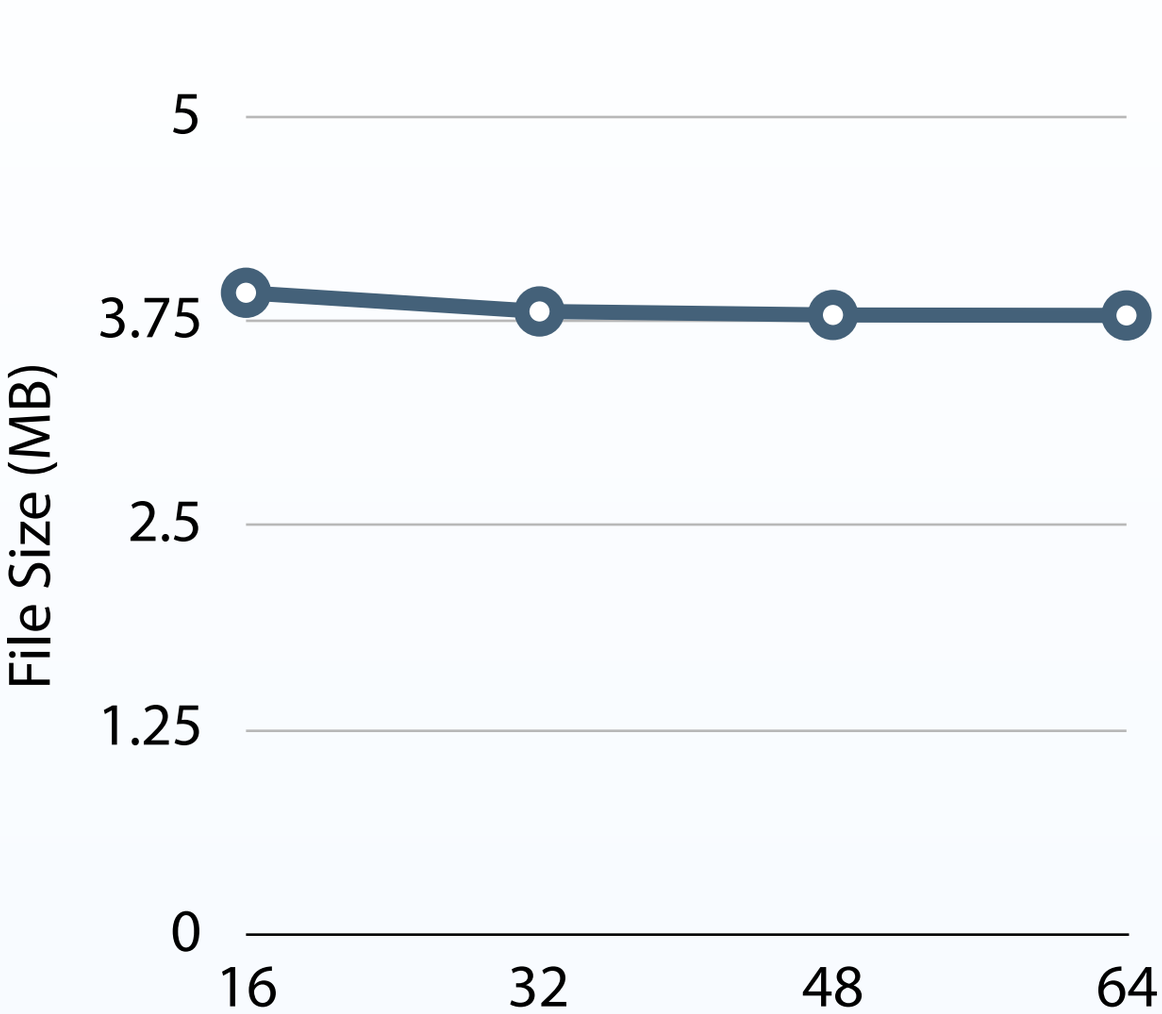
Video File Size when Compressed with Different Maximum Motion Vector Length

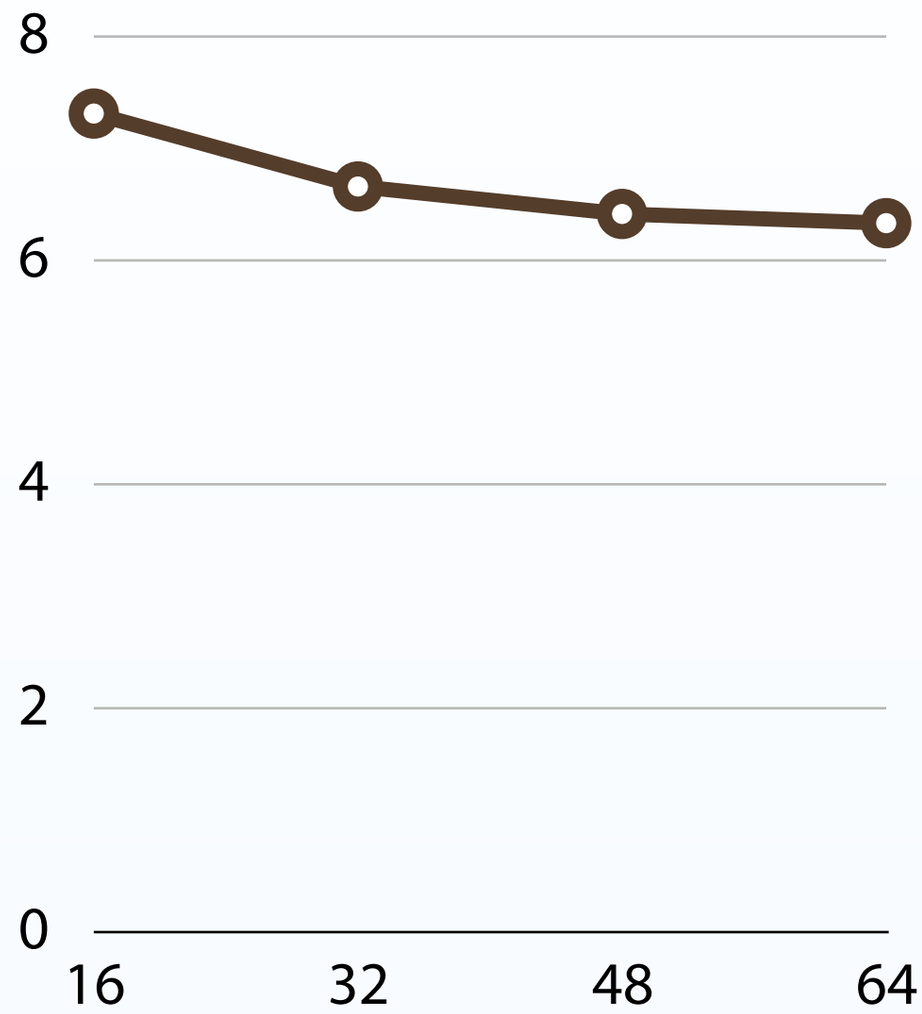




(Slice Size 64 bytes)

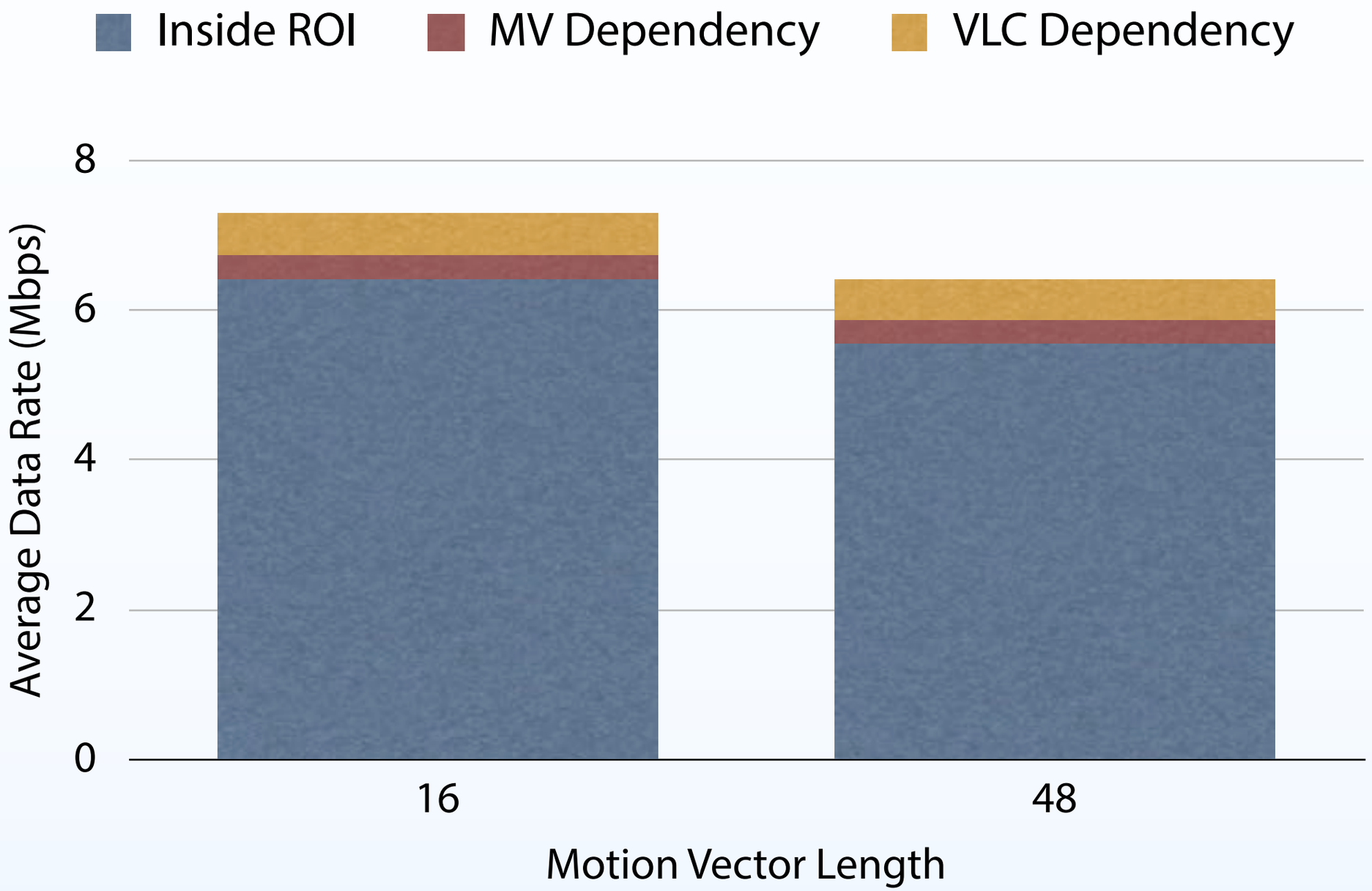
Average Data Rate when Streaming ROI of Size 30x30 Macroblocks





(Slice Size 64 bytes)

Breakdown of Bits Sent

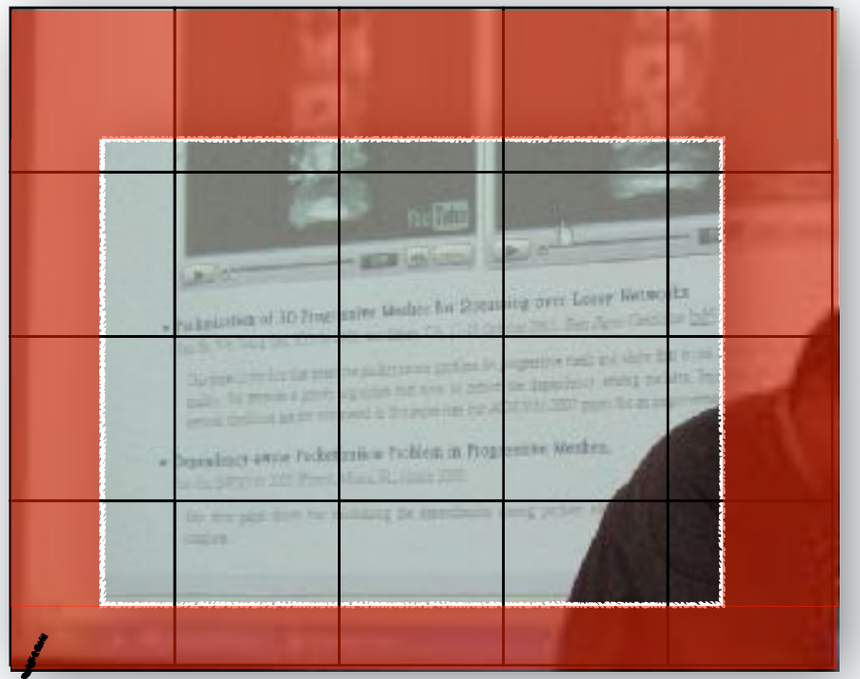


longer motion vector
is better

longer motion vector
is better

Hybrid

(Smarter Tiled Streams)



not used +
used for parsing +
used for decoding

for each tile:

send as if it is a monolithic stream

results: big improvements for large tiles, but still can’t beat monolithic stream

Summary

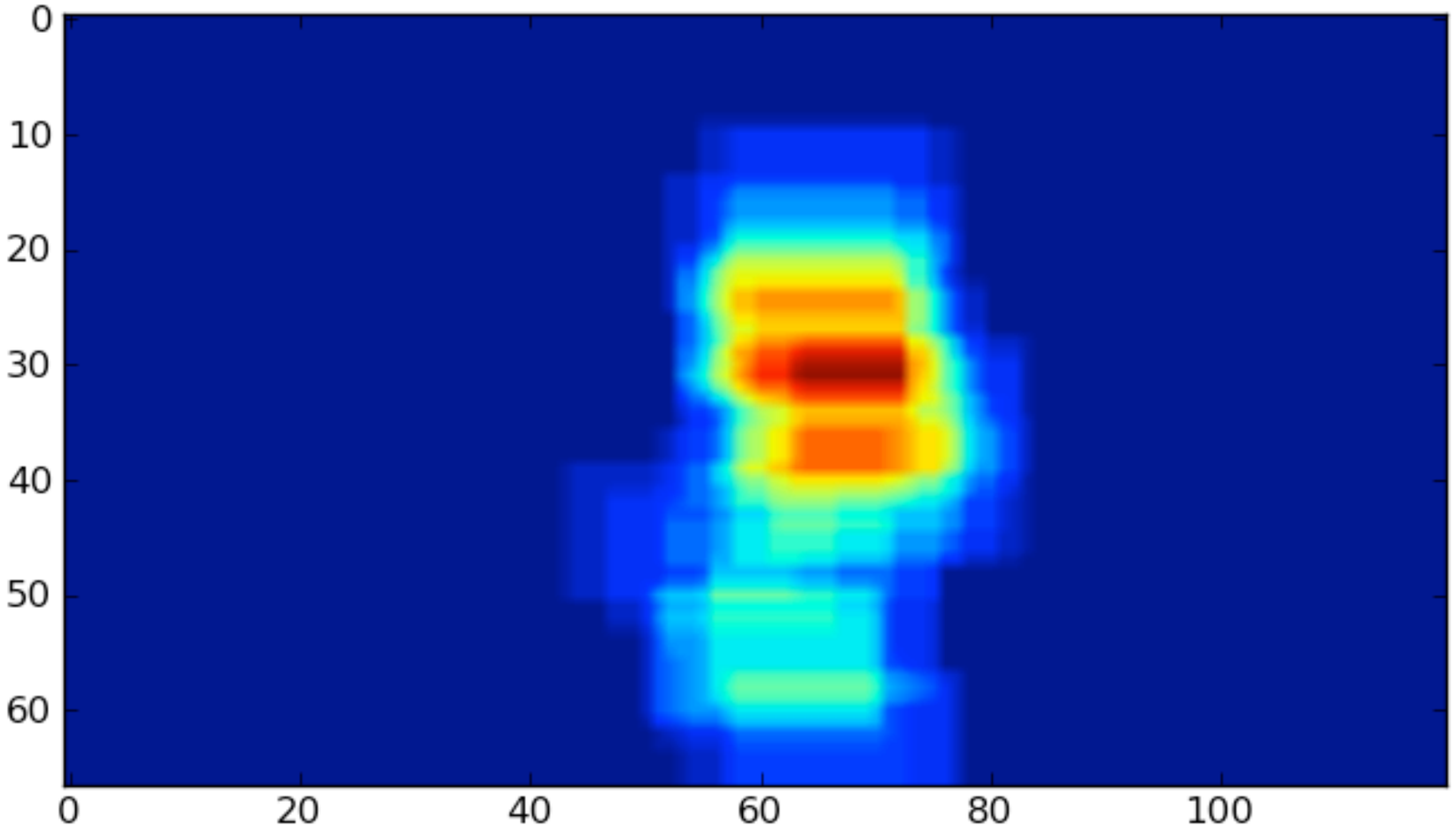
Monolithic	Tile
less bandwidth if encoding parameters carefully chosen	higher bandwidth
standard encoder can be used	need to modify encoder
much metadata needed	little metadata needed
no prefetching	prefetches surrounding areas
hard to multicast	multicast is easy: one group per tile

monolithic streaming is both **scalable** and **bandwidth-efficient** for dynamic ROI streaming,

encoding parameters need to be chosen carefully -- long motion vector is OK but long slice is not.

Parameters that are good for compression are not necessary good for zoomable video

Can we adapt the encoding parameters based on user access patterns?


0
10
20
30
40
50
60
0
20
40
60
80
100

Observation 1

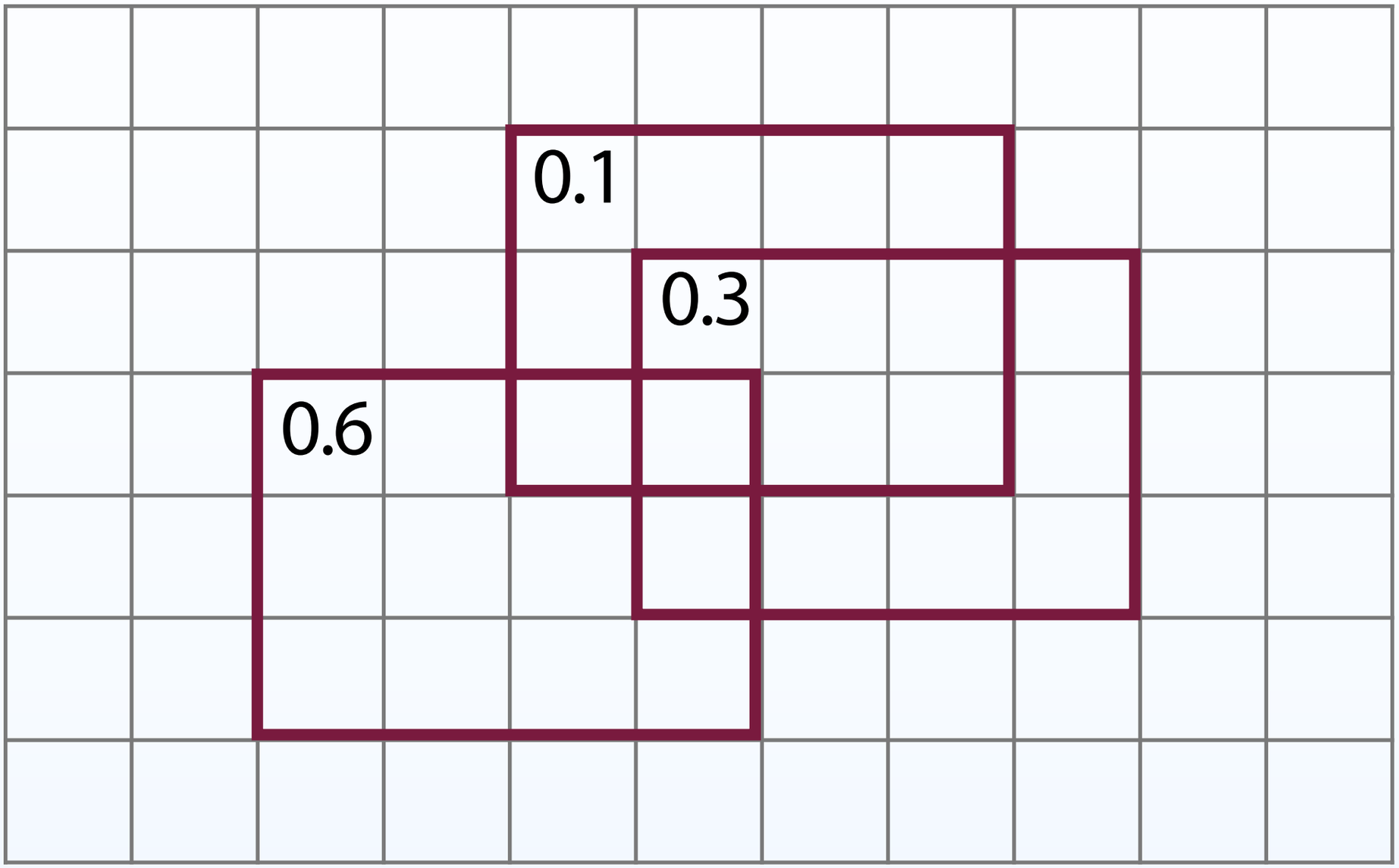
Areas with small access probability can have more dependencies and larger tiles

Observation 2

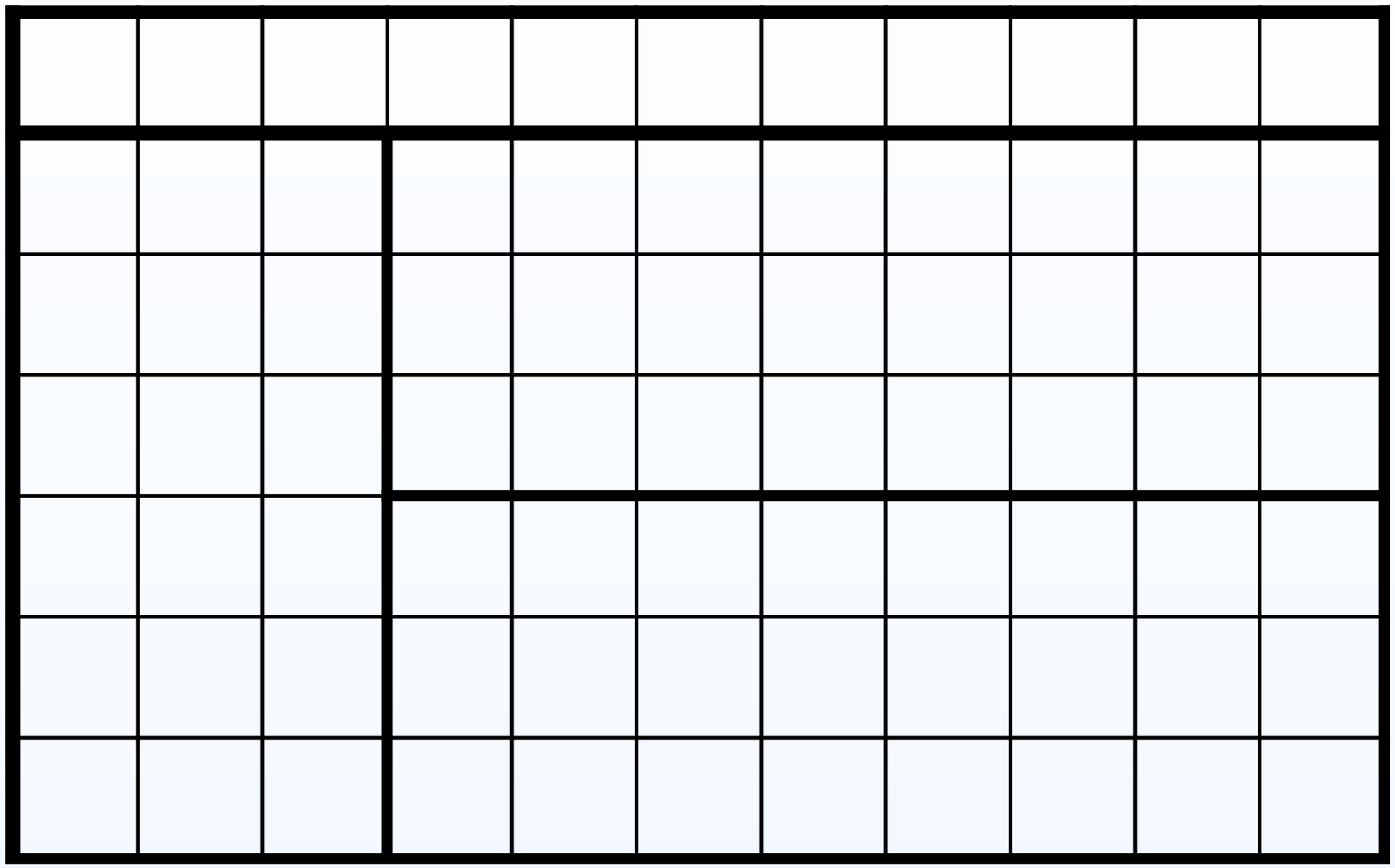
Areas that are accessed together can be put in the same tile

Given access pattern, find the best way to tile the video to reduce the expected bandwidth.

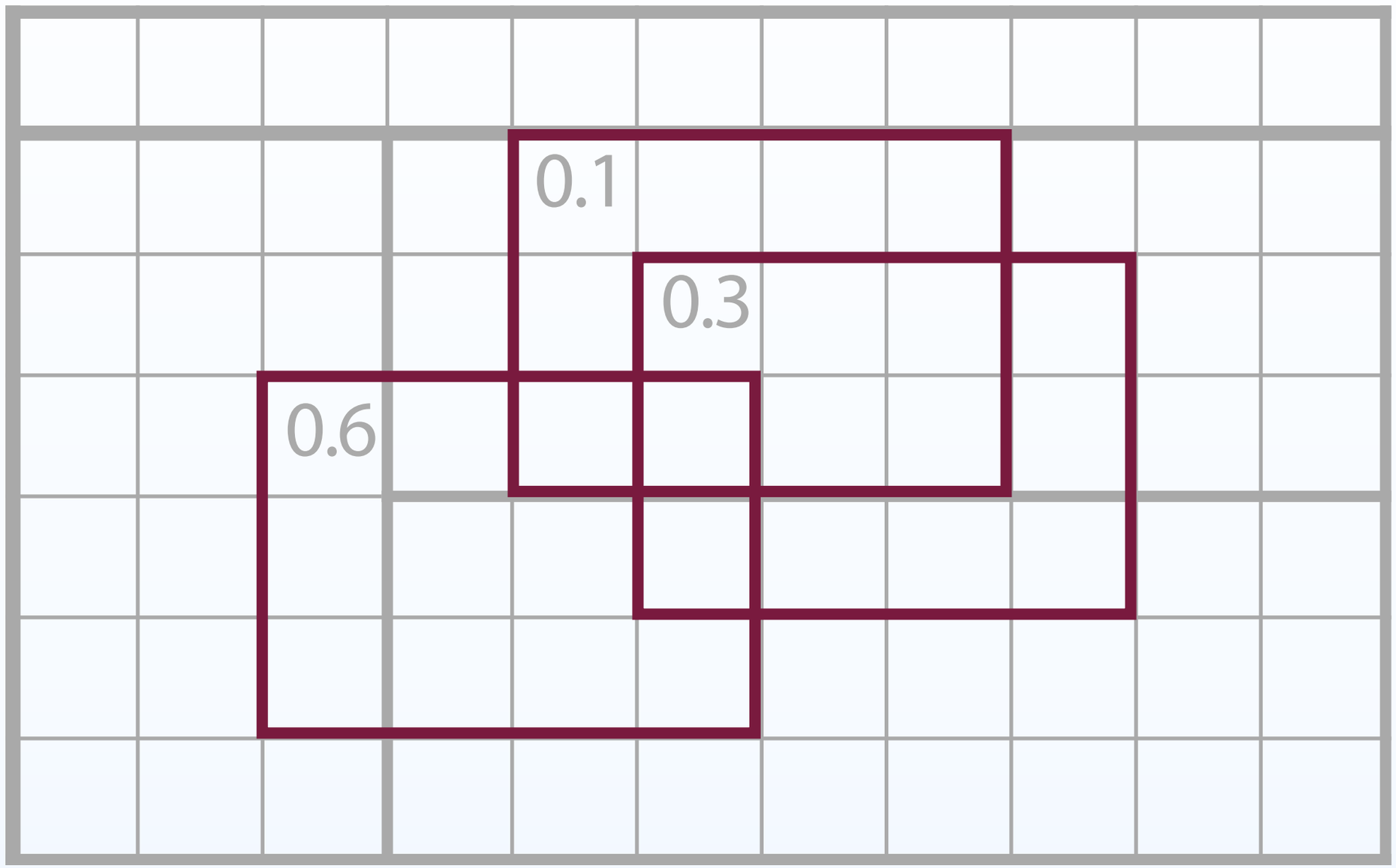
Access Probability



cost = size (in bytes) x total probability



cost = size of tile x total probability



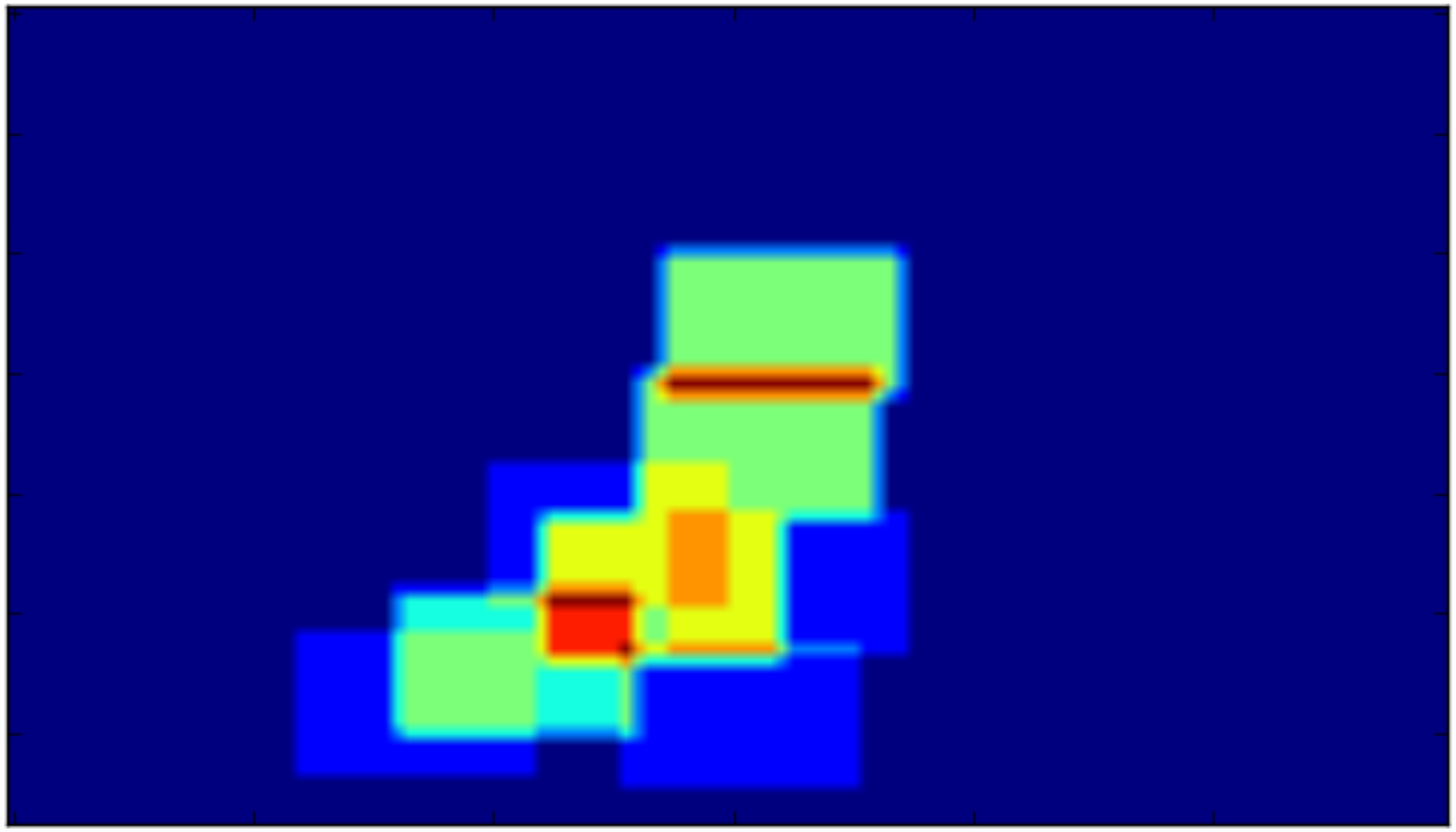
A greedy heuristic:

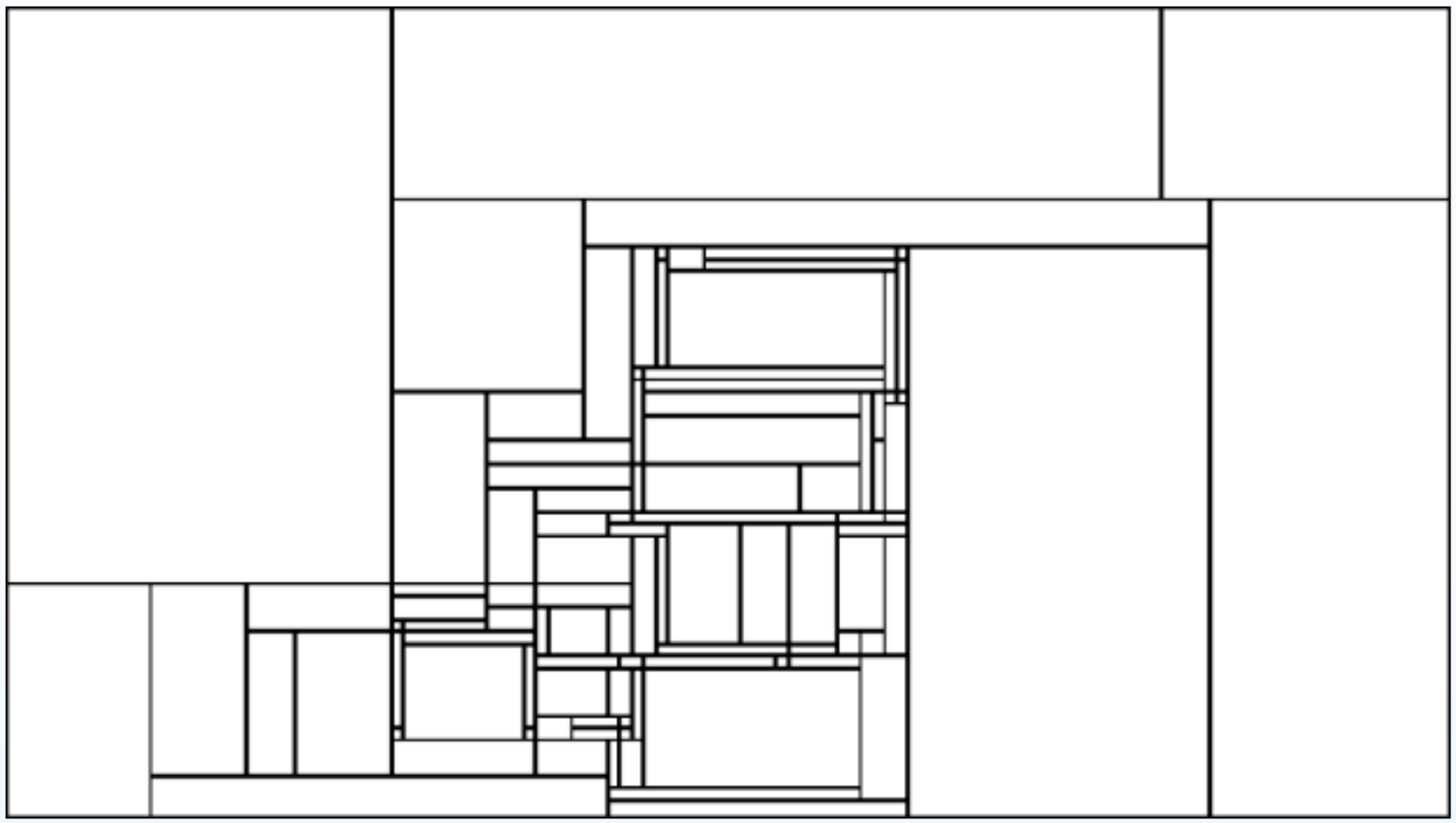
from left to right, top to bottom merge with neighbors if the expected size is reduced

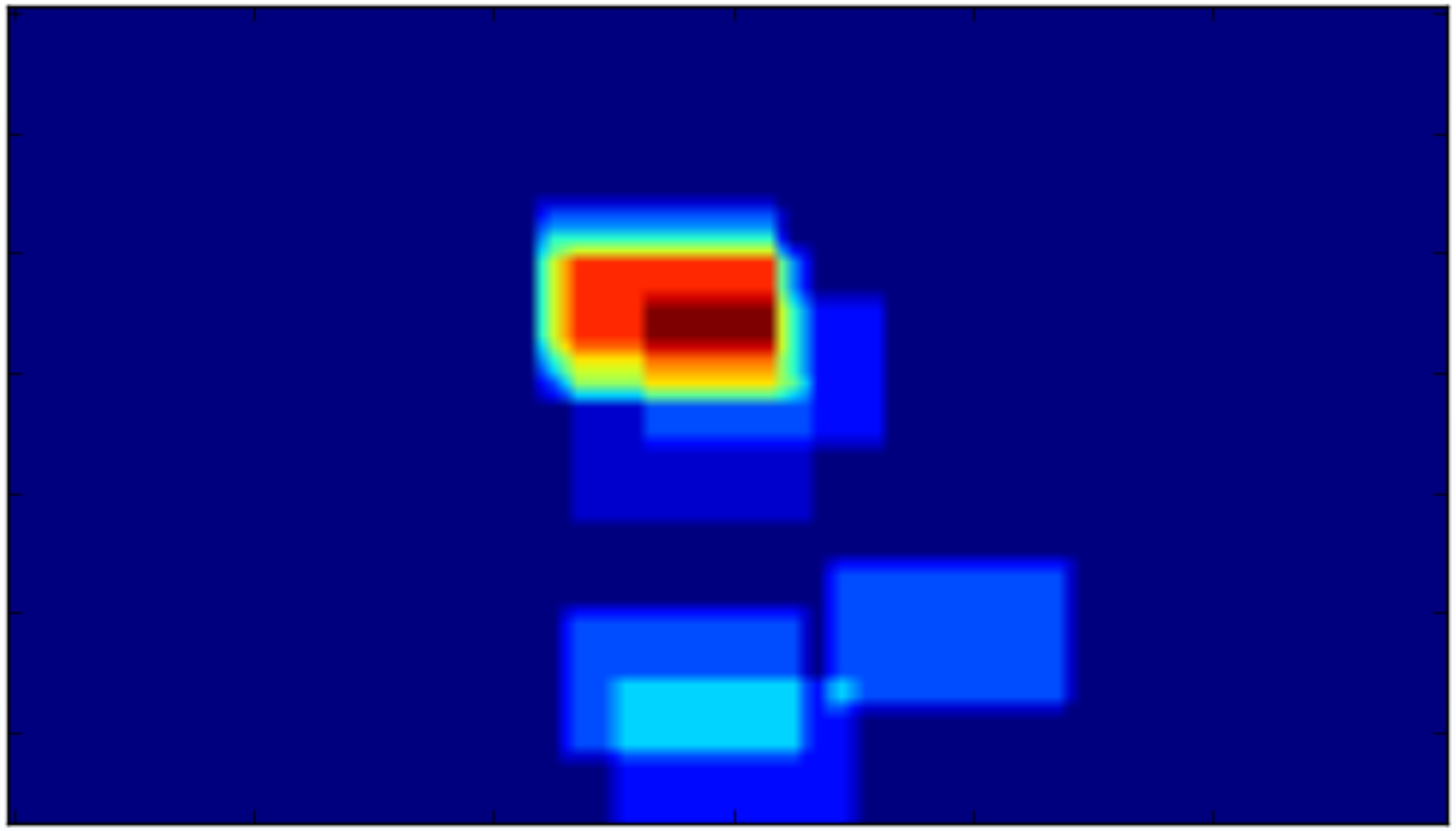
cost = 9
p = 0.7

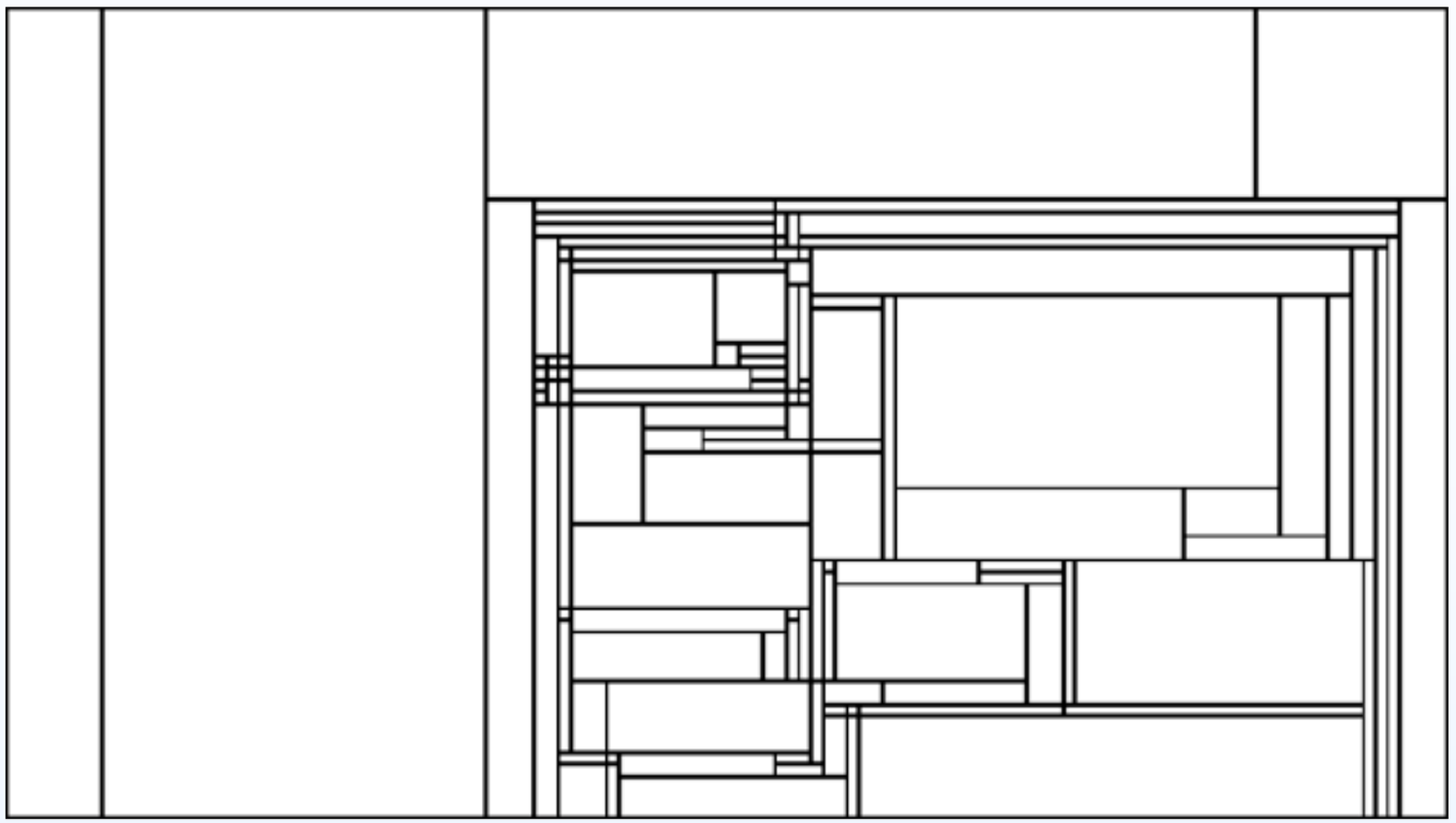
cost = 5
p = 0.8

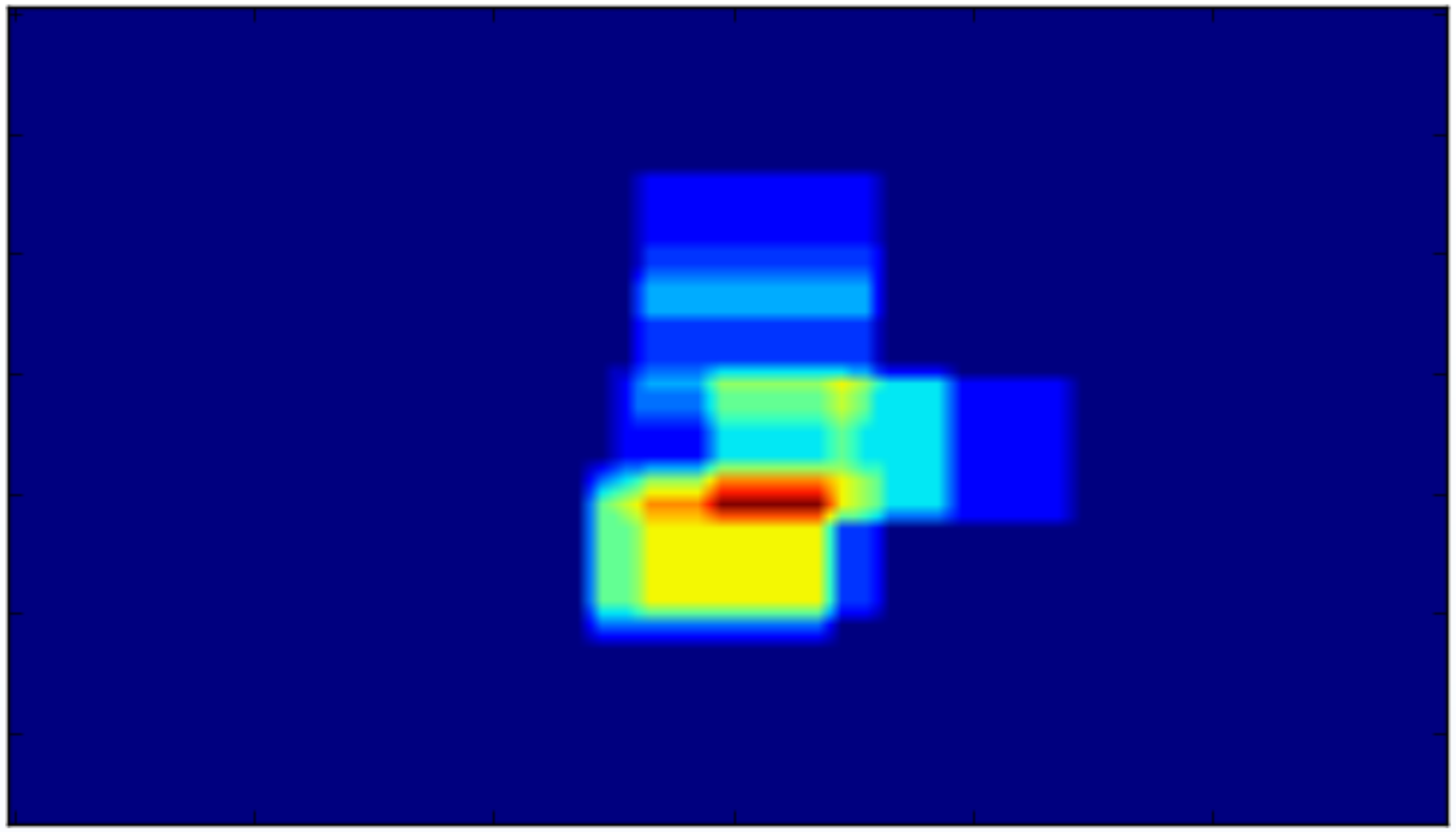
cost = 12
p = 1

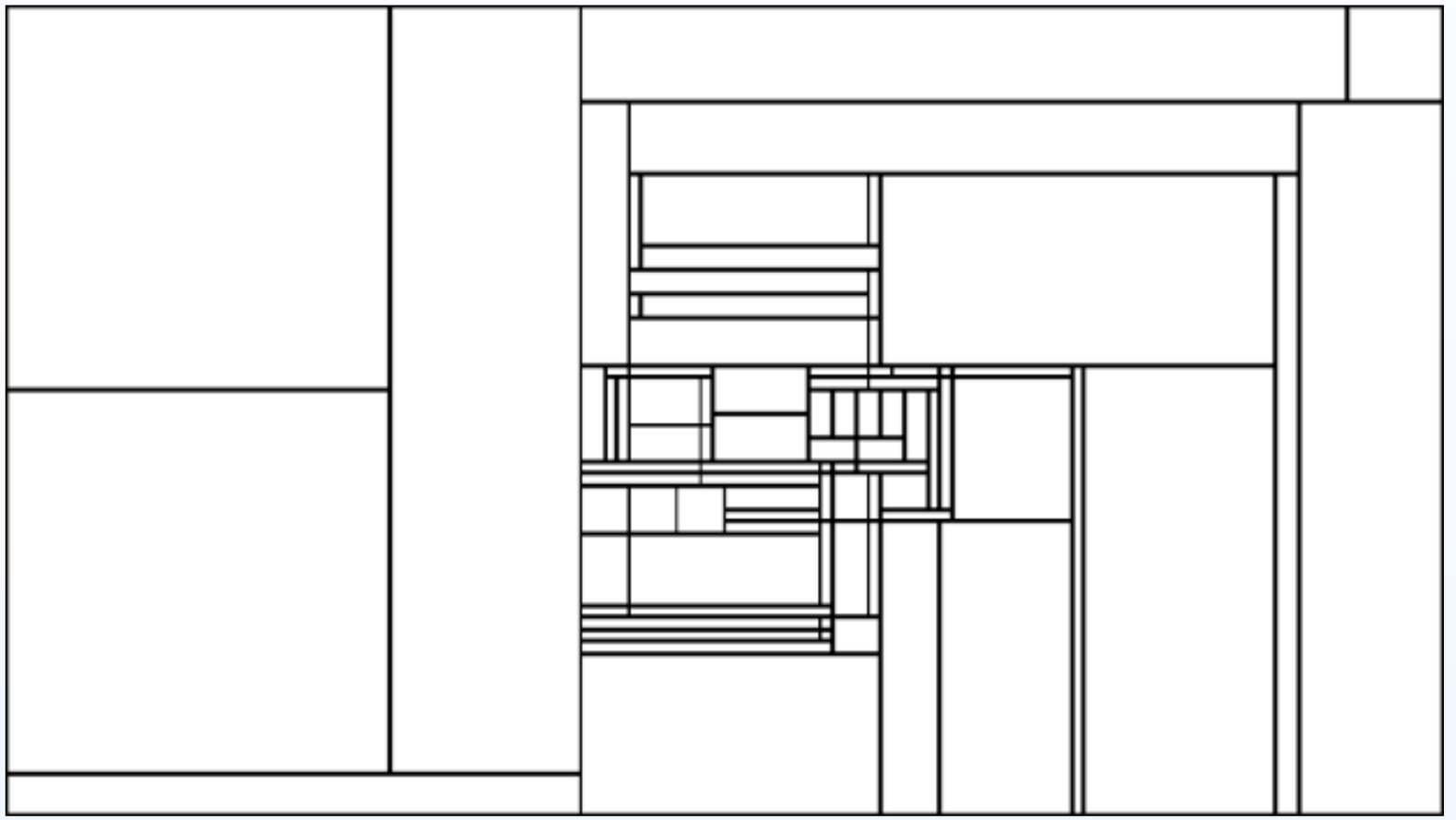












20 - 27%

Savings in Bandwidth

(even better than Monolithic Streaming)

Work done with

Ravindra Guntur

Postdoc

Work done with

Ngo Quang Minh Khiem

UROP/FYP

Work done with

Axel Carlier

UG Intern / RA

Main References

Supporting Zoomable Video Streams via Dynamic Region-Of-Interest Cropping

Ngo Quang Minh Khiem, Guntur Ravindra, Axel Carlier, and Wei Tsang Ooi, In Proceedings of the 1st ACM Multimedia Systems Conference (MMSYS'10), Scottsdale, AZ, 22-23 February 2010, 259-270.

Adaptive Encoding of Zoomable Video Streams Based on User Access Pattern

Ngo Quang Minh Khiem, Guntur Ravindra, and Wei Tsang Ooi, In Proceedings of the 2nd ACM Multimedia Systems Conference (MMSYS'11), Santa Clara, CA, 23-25 February 2011, 211-222.

Zoomable Video Playback on Mobile Devices by Selective Decoding

Feipeng Liu and Wei Tsang Ooi, In Proceedings of the 2012 Pacific-Rim Conference on Multimedia (PCM'12), Singapore, 4-6 December 2012, 251-262.

Additional Readings

Towards Characterizing Users' Interaction With Zoomable Video

Axel Carlier, Guntur Ravindra, and Wei Tsang Ooi, In Proceedings of the International Workshop on Social, Adaptive, and Personalized Multimedia Interaction and Access (SAPMIA'10), Florence, Italy, 29 October 2010.

Towards Understanding User Tolerance to Network Latency in Zoomable Video Streaming

Ngo Quang Minh Khiem, Guntur Ravindra, and Wei Tsang Ooi, In Proceedings of the 19th ACM International Conference on Multimedia (MM'11), Scottsdale, AZ, 28 November - 1 December 2011, 977-980, Short Paper

On Tile Assignment for Region-Of-Interest Video Streaming in a Wireless LAN

Guntur Ravindra and Wei Tsang Ooi, In Proceedings of the 22nd International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV'12), Toronto, Canada, 7-8 June 2012.

and more on http://www.comp.nus.edu.sg/~ooiwt/publications_by_topic.html#ZoomableVideo

Questions?