北京大学 PEKING UNIVERSITY

A Hardware-friendly CTU-level IME Algorithm for VVC

Xizhong Zhu[1]*, Guoqing Xiang*[1]*, Xiaofeng Huangy*[3]*, Yunyao Yan*[2]*, Huizhu Jia*[1] *and Xiaodong Xie*[1]

School of Computer Science, Peking University [1]
School of Electronic and Computer Engineering, Peking University [2]
School of Communication Engineering, Hangzhou Dianzi University [3]

Presenter : ***Xizhong Zhu***

CONTENTS

01 *Introduction*

Integer Motion Estimation	**Existing Works on IME**	**Challenges**

- ***Goal***:
  - Finding the MV of a block by block-matching in integer precision
- ***Process:***
  1. Select Initial Search Point (ISP) from a **predicted MV (PMV)**
  2. Search the search points around the ISP within a search range
  3. Record the SP with the most similar block
- ***Criteria:***
  - Simplified Rate-Distortion (RD) Cost as $J = SAD + \lambda R(MVD)$
- ***Features***:
  - Important and with high complexity
- ***Dilemma***:
  - Coding performance vs Complexity

01 Introduction

Integer Motion Estimation	*Existing Works on IME*	Challenges

- ***Algorithms with search patterns:***
  - Diamond Search -> TZSearch -> SP further reduced
  - Problem of irregular data access :
    - ***Hardware schemes such as pipeline and reuse FAIL***

- ***Hardware-friendly algorithms:***
  - Full search (FS) with regular data access
  - FS within adaptive search range (ASR)
  - FS within down-sampled pictures

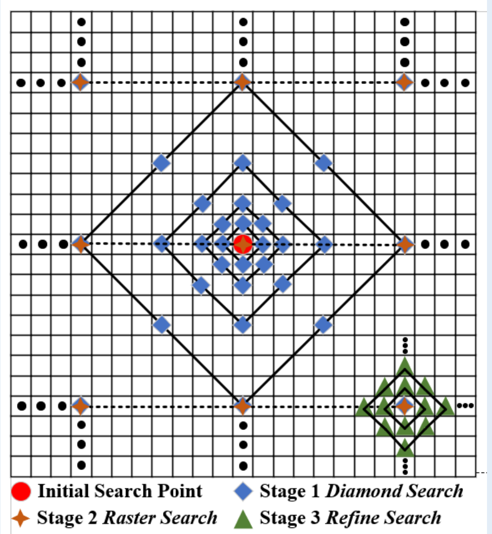

Fig. 1 TZSearch Patterns

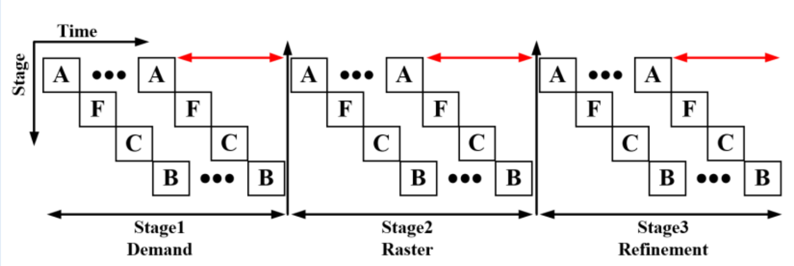

Fig. 2 Idle cycles in pipeline of TZSearch

01 Introduction

Integer Motion Estimation	Existing Works on IME	*Challenges*

- ***Architecture of a hardware encoder:***
  - Widely adopted : **CTU-level Pipeline => CTU-level IME**
  - Efficiency but with **stricter constrain** on data dependency
  - Challenge :
    - **Data dependency on deriving PMV :** AMVP
    - Early work : only apply PMV of CTU with **0.98% coding loss**

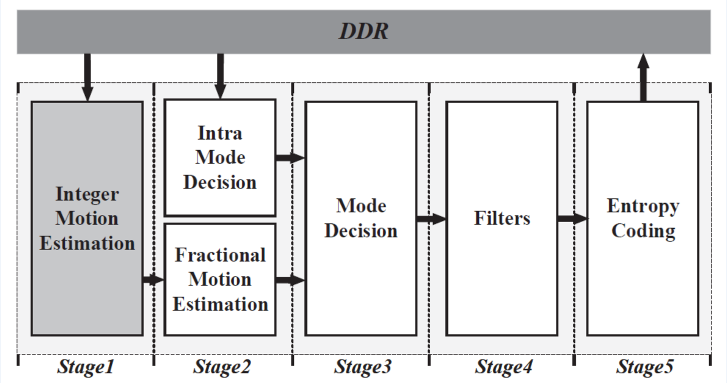

Fig. 1 CTU-level pipeline architecture

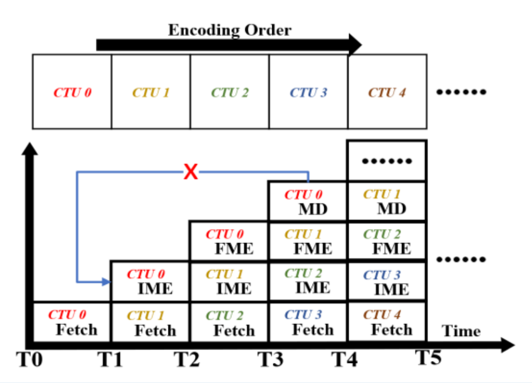

Fig. 2 CTU-level pipeline

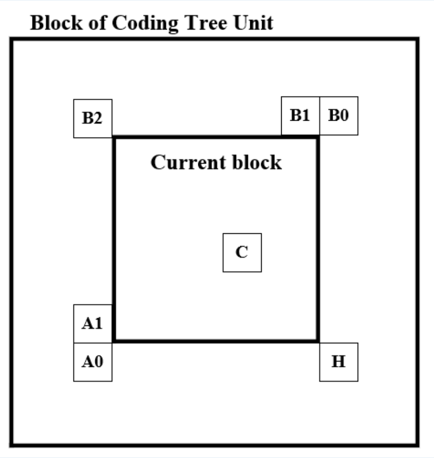

Fig. 3 Required coded MVs

01 Introduction

Integer Motion Estimation	Existing Works on IME	*Challenges*

- ***Increased number of possible divided block***
  - HEVC
    - Coding Unit (CU): Quart Tree (QT) -> Prediction Unit (PU)
    - Maximum **593** blocks from a CTU of 64x64
  - **VVC**
    - CU: QT + Binary Tree (BT) + Ternary Tree (TT)
    - Five spit types
    - CU Size = PU Size
    - Maximum **1661** blocks from a CTU of 64x64

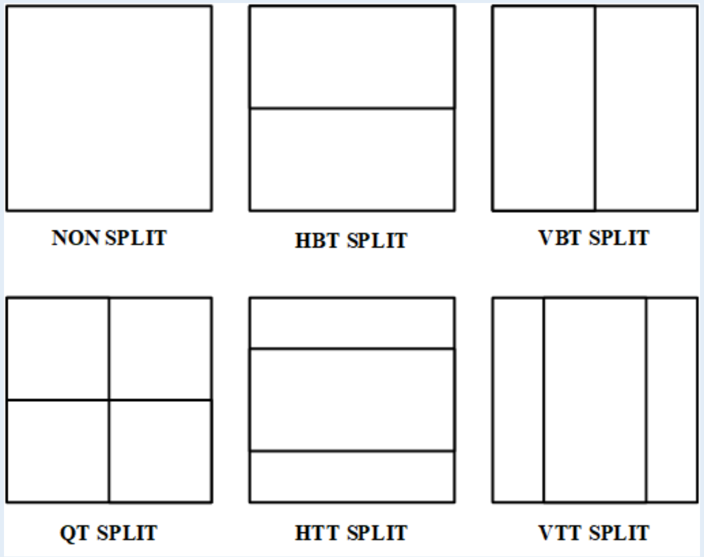

Fig. 1 Five split types

02 *Proposed Method*

北京大學 PEKING UNIVERSITY

Predicted Motion Vector Prediction Based on Affine Motion Model — 01 — Data dependency in PMVs

Multi-resolution Search Algorithm — 02 — Search with regular data access

Motion Vector Inference Based on Error Surface Model — 03 — Increased divided blocks

02 Proposed Method

Overall

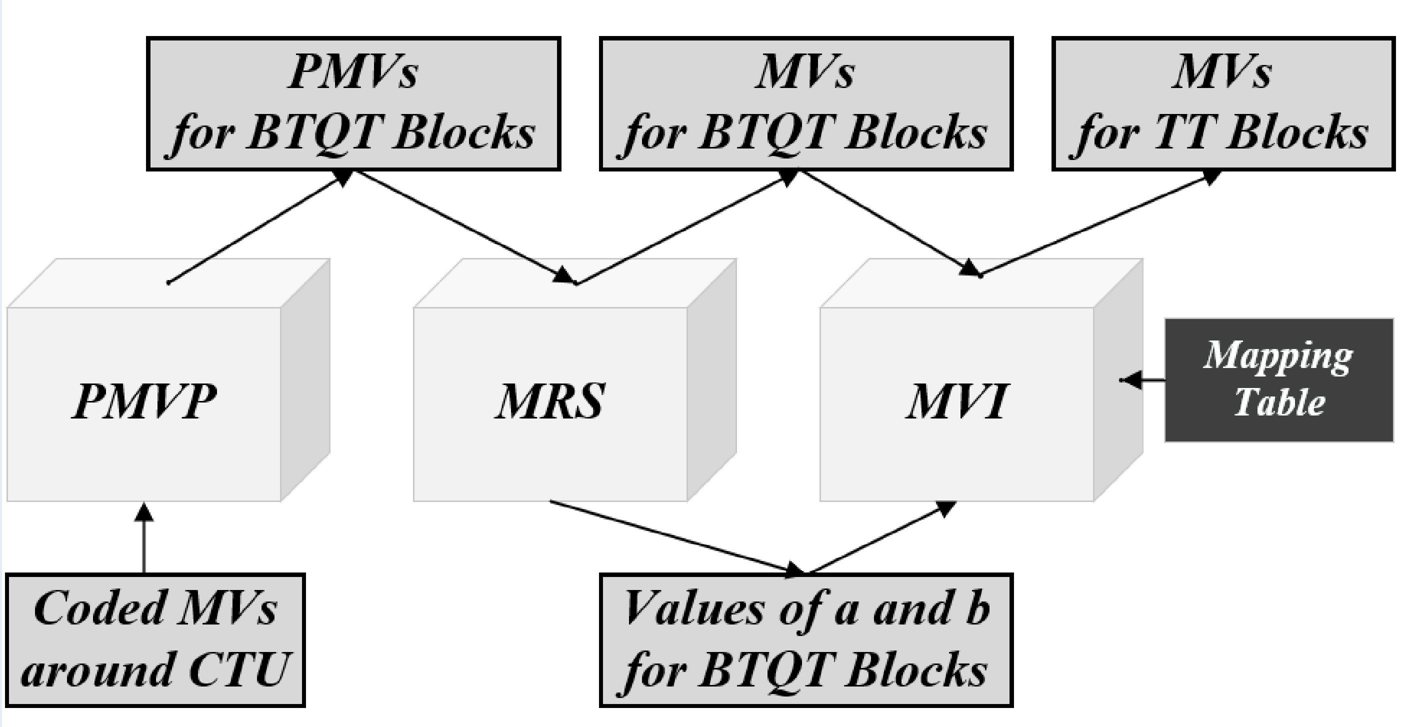

Fig. 1 Overall procedure

02 Proposed Method

Predicted Motion Vector Prediction Based on Affine Motion Model

- **Basic Idea**
  1. **Model** the coded MV field (MVF) within the CTU
  2. **Estimate** the model with the coded MVs around the CTU
  3. **Predict** the PMV with the model
- **Model : *Affine motion model (AMM)***
  - Affine motion model

$$\begin{cases} MVx = ax + by + c \\ MVy = dx + ey + f \end{cases} \quad (1)$$

  - Fitness of modeling the coded MVF

Resolution	Name	R^2	Name	R^2
3840x2160	FoodMarket4	0.89	Campfire	0.83
1920x1080	RitualDance	0.83	BasketballDrive	0.91
832x480	BQMall	0.87	PartyScene	0.82
416x240	RaceHorses	0.81	BQSquare	0.98

02 *Proposed Method*

Predicted Motion Vector Prediction Based on Affine Motion Model

- **Estimate :** ***MV extraction and parameter estimation***

Three lists of 4x4 blocks, each records first two ***MV*** in matrix as **M** and its ***position*** in matrix as **P**

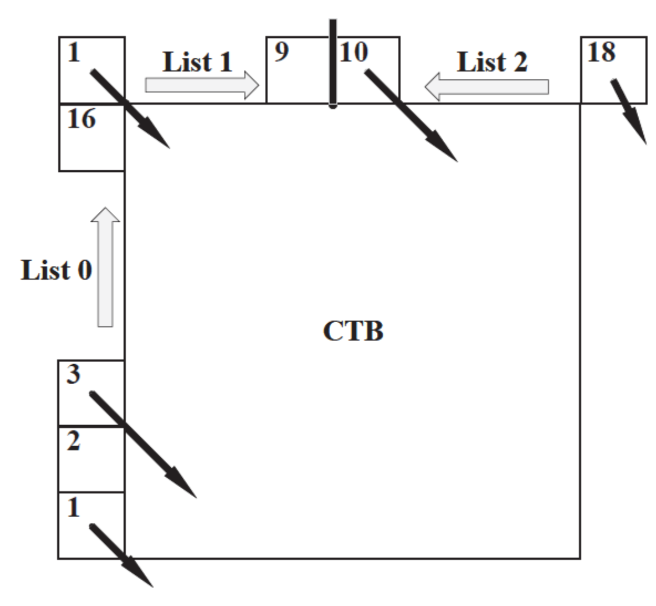

Fig. 1 Three candidate list

$$M = \begin{bmatrix} mvx_0 & mvy_0 \\ \dots & \dots \\ mvx_5 & mvy_5 \end{bmatrix}$$

$$P = \begin{bmatrix} x_0 & y_0 & 1 \\ \dots & \dots & \dots \\ x_5 & y_5 & 1 \end{bmatrix}$$

02 Proposed Method

Predicted Motion Vector Prediction Based on Affine Motion Model

- **Estimate :** ***MV extraction and parameter estimation***
  - $\begin{cases} MVx = ax + by + c \\ MVy = dx + ey + f \end{cases}$
  - $\boldsymbol{PA} = \mathbf{M}$

    $\Rightarrow$ **Norm equation**

    $\Rightarrow \mathbf{A} = \left(\mathbf{P}^{\mathbf{T}}\mathbf{P}\right)^{-\mathbf{1}}\mathbf{P}^{\mathbf{T}}\mathbf{M}$ **(2)**

$$P = \begin{bmatrix} x_0 & y_0 & 1 \\ \dots & \dots & \dots \\ x_5 & y_5 & 1 \end{bmatrix} \quad A = \begin{bmatrix} a & b & c \\ e & f & g \end{bmatrix}^T \quad M = \begin{bmatrix} mvx_0 & mvy_0 \\ \dots & \dots \\ mvx_5 & mvy_5 \end{bmatrix}$$

  - The parameter can be estimated.
- **Predict :**
  - $\begin{cases} PMVx = ax + by + c \\ PMVy = dx + ey + f \end{cases}$ **(3)**
  - The predicted PMVs will be utilized in the search process to derive the RD costs

02 *Proposed Method*

BTQT and TT blocks

- TT blocks
  - **Def : Only** available if **TT split** is allowed
  - Number : 700

- BTQT blocks
  - The rest of the blocks
  - Number : 961
  - MVs are derived from ***MRS***

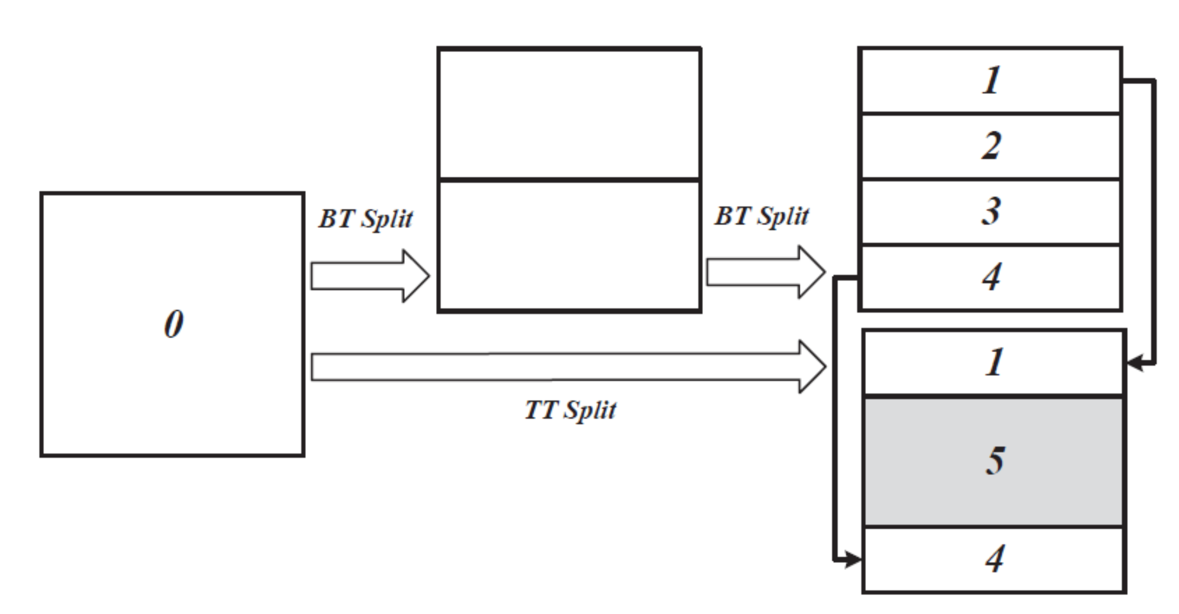

Fig. 1 An example of BTQT and TT blocks

02 Proposed Method

Multi-resolution Search (MRS)

- **Process**
  - ***Level 2***
    - 16:1 down-sampled picture
    - Range [-128, 127] center on (0, 0)
    - Two best MVs of CTB : MV_0^2 and MV_1^2
  - ***Level 1***
    - 4:1 down-sampled picture
    - Range [-32, 31] center on MV_0^2, MV_1^2 and PMV^{CTB}
    - One best MVs of CTB : MV^1
  - ***Level 0***
    - 1:1 fine picture
    - Range [-8, 7] center on MV^1
    - MVs for all blocks to be searched

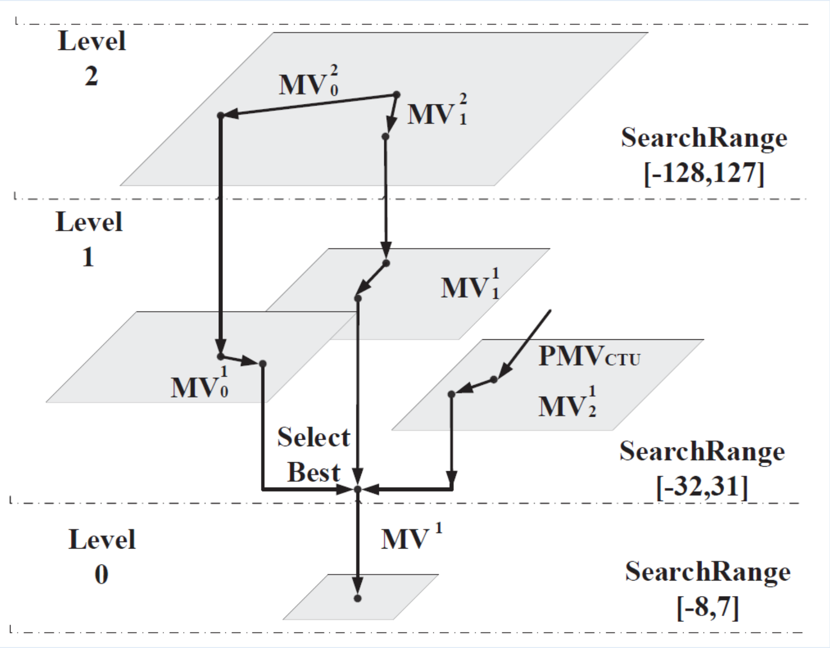

Fig. 1 Process of MRS

02 Proposed Method

Motion Vector Inference Based on Error Surface Model

- **Any TT block can be split into two BTQT blocks**
- For a certain SP at (x, y)

$$SAD_{TT}(x,y) = SAD_{BTQT}^{0}(x,y) + SAD_{BTQT}^{1}(x,y) \quad \textbf{(3)}$$

- Quadratic Error Surface Model

$$SAD(x,y) = a(x - x_{min})^2 + b(y - y_{min})^2 + k \quad \textbf{(4)}$$

- TT SAD in Quadratic Error Surface model

$$SAD_{TT}(x,y) = \alpha\left(x - \frac{a_0 x_{min}^0 + a_1 x_{min}^1}{a_0 + a_1}\right)^2 + \beta\left(y - \frac{b_0 y_{min}^0 + b_1 y_{min}^1}{b_0 + b_1}\right)^2 + \lambda \quad \textbf{(5)}$$

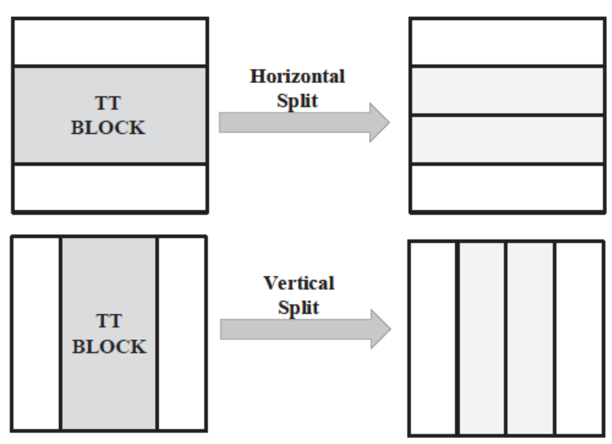

Fig. 1 Relation between a TT block and two BTQT blocks

02 Proposed Method

Motion Vector Inference Based on Error Surface Model

$$SAD_{TT}(x,y) = \alpha\left(x - \frac{a_0 x_{min}^0 + a_1 x_{min}^1}{a_0 + a_1}\right)^2 + \beta\left(y - \frac{b_0 y_{min}^0 + b_1 y_{min}^1}{b_0 + b_1}\right)^2 + \lambda \quad \textbf{(5)}$$

- Min value is obtained at $\left(\frac{a_0 x_{min}^0 + a_1 x_{min}^1}{a_0 + a_1}, \frac{b_0 y_{min}^0 + b_1 y_{min}^1}{b_0 + b_1}\right)$
- (x_{min}^0, y_{min}^0) and (x_{min}^1, y_{min}^1)
  - => ***MVs of two BTQT block* (From *MRS*)**
- a_0, b_0, a_1, b_1
  - => Parameters of BTQT blocks' error surfaces **(Unknown)**
  - => ***Can be estimated in MRS***

02 Proposed Method

Motion Vector Inference Based on Error Surface Model

- Estimation of a_0, b_0, a_1, b_1

$$\begin{cases} \mathrm{a} = \frac{1}{2}(\mathrm{SAD}(\mathrm{x_{min}} + 1, \mathrm{y_{min}}) + \mathrm{SAD}(\mathrm{x_{min}} - 1, \mathrm{y_{min}}) - 2\mathrm{SAD}(\mathrm{x_{min}}, \mathrm{y_{min}})) \\ \mathrm{b} = \frac{1}{2}(\mathrm{SAD}(\mathrm{x_{min}}, \mathrm{y_{min}} + 1) + \mathrm{SAD}(\mathrm{x_{min}}, \mathrm{y_{min}} - 1) - 2\mathrm{SAD}(\mathrm{x_{min}}, \mathrm{y_{min}})) \end{cases} \quad (6)$$

Simplified

$$\begin{cases} \mathrm{a} = \mathrm{SAD}(\mathrm{x_{min}} + 1, \mathrm{y_{min}}) - \mathrm{SAD}(\mathrm{x_{min}}, \mathrm{y_{min}}) \\ \mathrm{b} = \mathrm{SAD}(\mathrm{x_{min}}, \mathrm{y_{min}} + 1) - \mathrm{SAD}(\mathrm{x_{min}}, \mathrm{y_{min}}) \end{cases} \quad (7)$$

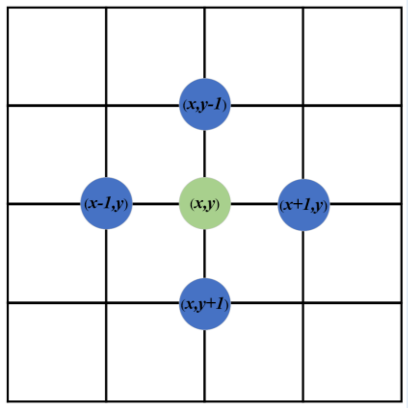

Fig. 1 Required SAD values

Recorded for all BTQT blocks in MRS

- Finally

$$\boldsymbol{MV_{TT}} = \left(\frac{a_0 x_{min}^0 + a_1 x_{min}^1}{a_0 + a_1}, \frac{b_0 y_{min}^0 + b_1 y_{min}^1}{b_0 + b_1}\right)$$

03 *Experimental Result*

Test Conditions	Result	Future work

- **Implemented on VVC reference software VTM 10.0**
- **QP set**
  22, 27, 32, 37
- **Coding Structure : Low-delay P**
- **Test sequences**
  - Sequences from A1, A2, B, C and D in common test condition (CTC) of VVC
- **Metric**
  - Anchor **VTM 10.0**
  - Coding performance : BD-BR
  - Complexity reduction : Time reduction ratio (TR)

$$\mathrm{TR} = \frac{T_{Ref} - T_{Test}}{T_{Ref}} \times 100\%$$

03 *Experimental Result*

Test Conditions | ***Result*** | **Future work**

- **Two comparisons**
  - ***MS*** : An algorithm that further reduced the number of SPs in TZSearch by 1/3
  - ***ASR*** : An algorithm that performs FS within an adaptive search range
- **Overall performance**
  - Time reduced by ***81%***
  - Performance loss ***1.20%***
- **Individual performances**

＼	PMVP	MRS	MVI	All
BD-BR	0.03%	0.66%	0.51%	1.20%
TR	–	63.32%	99.85%	81.26%

Class	Sequence	MS [4]		ASR [7]		Ours	
		BD-BR	TR	BD-BR	TR	BD-BR	TR
A1	*Tango2*	1.01%	48%	1.21%	-1134.31%	**0.95%**	**78%**
	FoodMarket4	0.91%	61%	0.81%	-1051.49%	**0.87%**	**37%**
	Campfire	0.49%	63%	0.52%	-621.32 %	**0.52%**	**92%**
A2	*CatRobot1*	1.04%	37%	0.94%	-1440.71%	**1.02%**	**75%**
	DaylightRoad2	1.88%	41%	1.87%	-1241.47%	**1.87%**	**80%**
	ParkRunning3	0.71%	35%	0.82%	-1188.99%	**0.72%**	**88%**
B	*MarketPlace*	1.46%	23%	1.26%	-1496.47%	**1.41%**	**78%**
	RitualDance	1.26%	43%	1.16%	-1132.26%	**1.26%**	**84%**
	Cactus	0.69%	22%	0.49%	-1597.64%	**0.62%**	**81%**
	BasketballDrive	2.01%	35%	2.00%	-1149.04%	**1.98%**	**86%**
C	*RaceHorses*	1.66%	30%	1.56%	-581.98%	**1.66%**	**89%**
	BQMall	0.99%	12%	1.01%	-791.88%	**1.01%**	**81%**
	PartyScene	1.01%	26%	0.91%	-884.36%	**0.92%**	**85%**
	BasketballDrill	2.52%	21%	2.82%	-721.05%	**2.21%**	**85%**
D	*RaceHorses*	1.63%	21%	1.43%	-605.63%	**1.60%**	**88%**
	BQSquare	0.13%	19%	0.14%	-865.15%	**0.24%**	**78%**
	BlowingBubbles	0.94%	25%	0.96%	-845.36%	**1.04%**	**82%**
	BasketballPass	1.77%	20%	1.87%	-614.15%	**1.79%**	**87%**
	Average	1.23%	32%	1.21%	-997.96%	**1.20%**	**81%**

03 *Experimental Result*

Test Conditions | **Result** | ***Future work***

- ***Improvement on performance loss***
  - ***From MRS*** : Inaccuracy MV on down-sampled picture
  - ***From MVI*** : Inaccurate modeling of error surface
- ***Further hardware implementations***

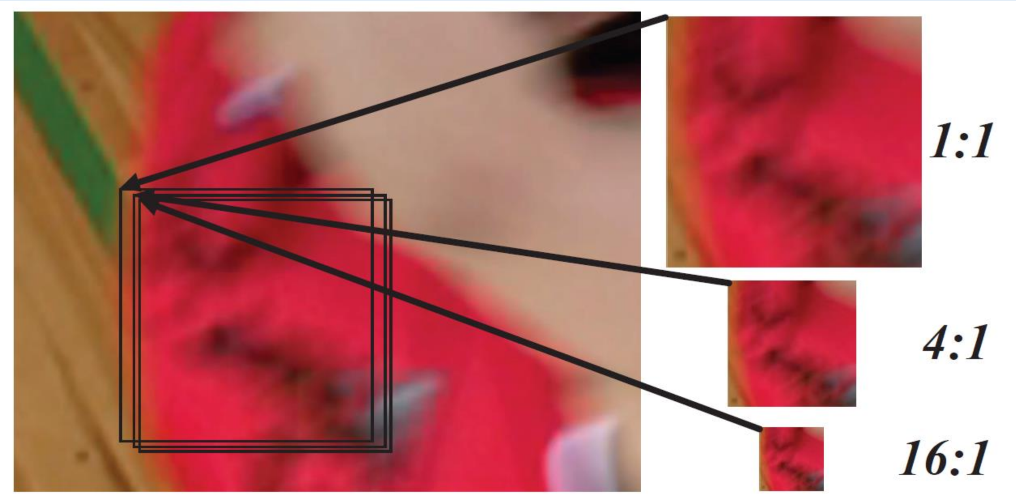

Fig. 1 Example of inaccurate search

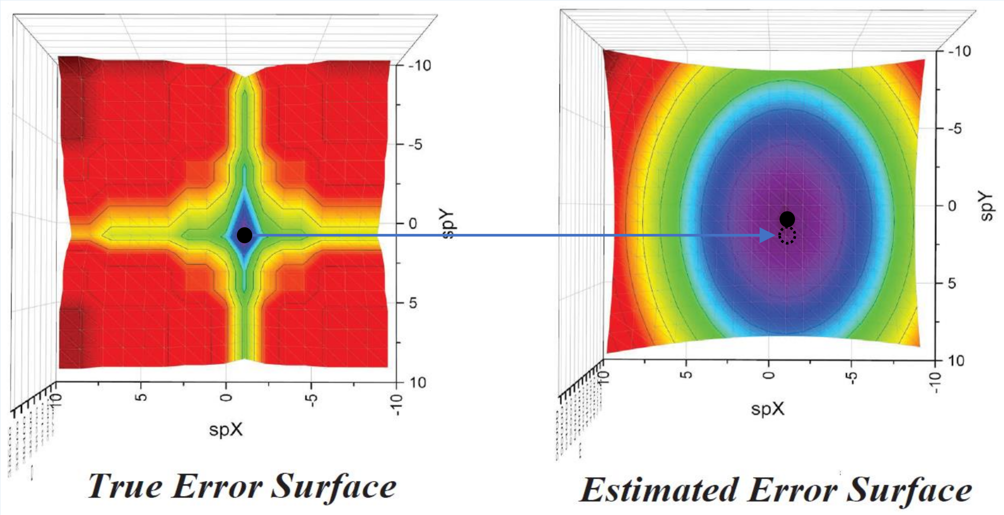

Fig. 2 Example of inaccurate modeling

Thanks

Looking forward to your questions and opinions .