BOXR: Body and head motion Optimization framework for eXtended Reality

Ziliang(Johnson) Zhang, Zexin Li, Hyoseung Kim, Cong Liu
University of California, Riverside
RTSS 2024

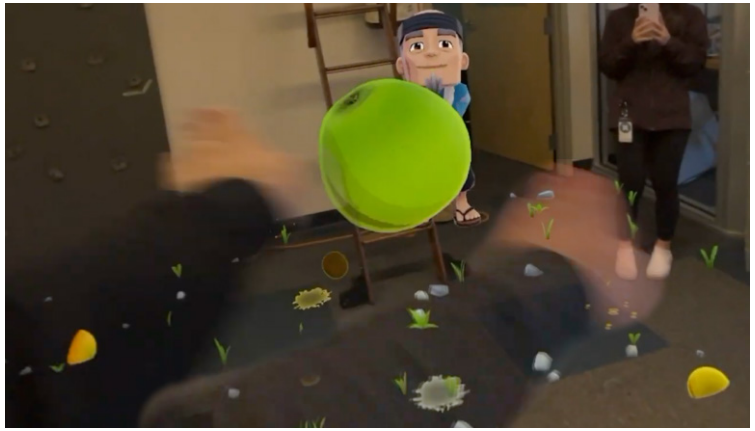
Extended Reality (XR)

What is Extended Reality (XR)?

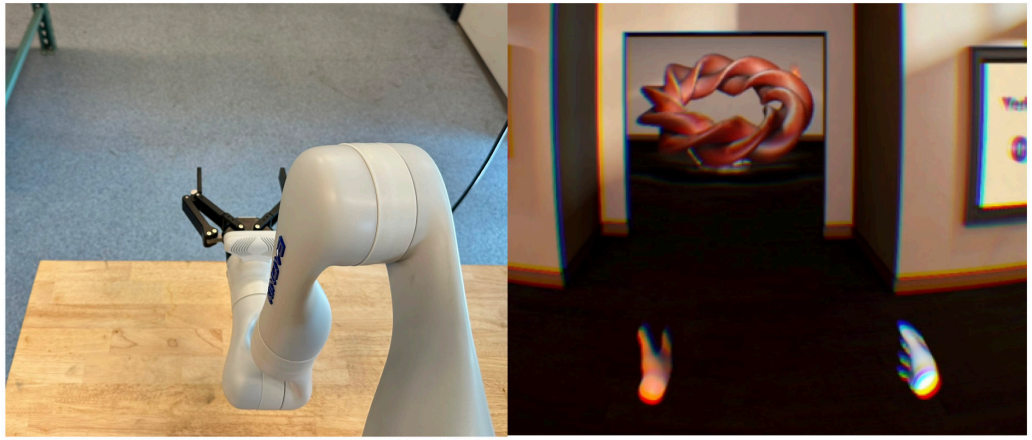
- Creates an immersive virtual-physical user experience
- Includes Virtual Reality (VR), Augmented Reality (AR), Mixed Reality (MR)

Why XR is useful?

- Intuitive control: user motion directly interact with virtual environment
- Diverse use cases: entertainment, teleoperation, accessibility, etc.



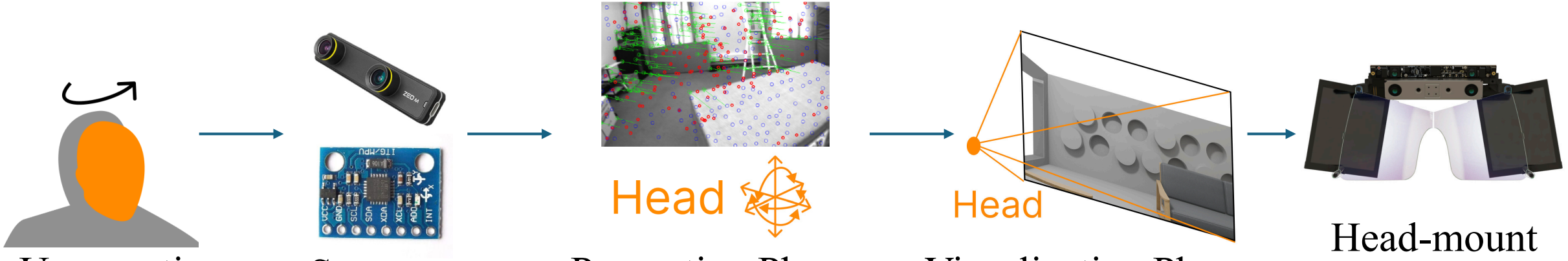
*



* iFixit, Vision Pro Teardown: Behind the Complex and Creepy Tech, https://youtu.be/JVJPAYwY8Us

XR pipeline and challenges

How XR systems work?



User motion → Sensors → Perception Phase → Visualization Phase → Head-mount Display (HMD)

Challenges of XR systems

- **Tight budget for motion display**: users can sense motion misalignment in 20ms*
- **Resource constraints**: standalone XR devices use embedded systems with limited computational resources

* Huzaifa, Muhammad, et al. "ILLIXR: Enabling end-to-end extended reality research." 2021 IEEE International Symposium on Workload Characterization (IISWC), 2021.

XR Sensors

Inertial Measurement Unit (**IMU**):

- Sample Rate: 500Hz ~ 1000Hz
- Data: acceleration and rotation

Stereo Camera (**CAM**):

- Sample Rate: 20Hz ~ 60Hz
- Data: a picture of surrounding information (With depth information)

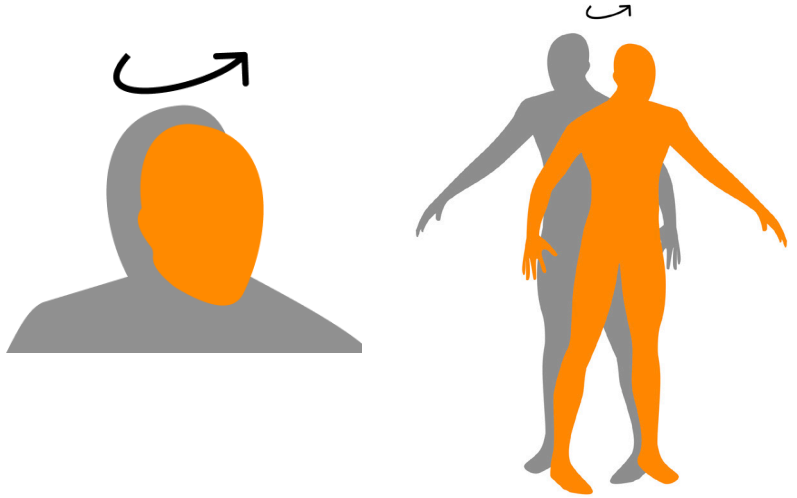
Apple Vision Pro sensors*

* Sarang Sheth, "Every Single Sensor inside the Apple Vision Pro and What It's Individually Designed To Do." https://www.yankodesign.com/2023/06/07/every-single-sensor-in-the-apple-vision-pro-and-what-its-individually-designed-to-do/

User Motion

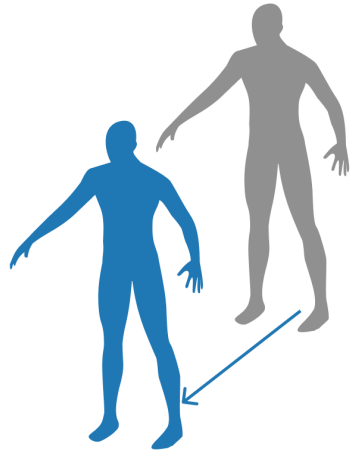
Head Motion

- Frequent rotational or small-magnitude translational movements
- i.e., head shaking, torso rotation, body leaning
- Can be accurately captured by **IMU**



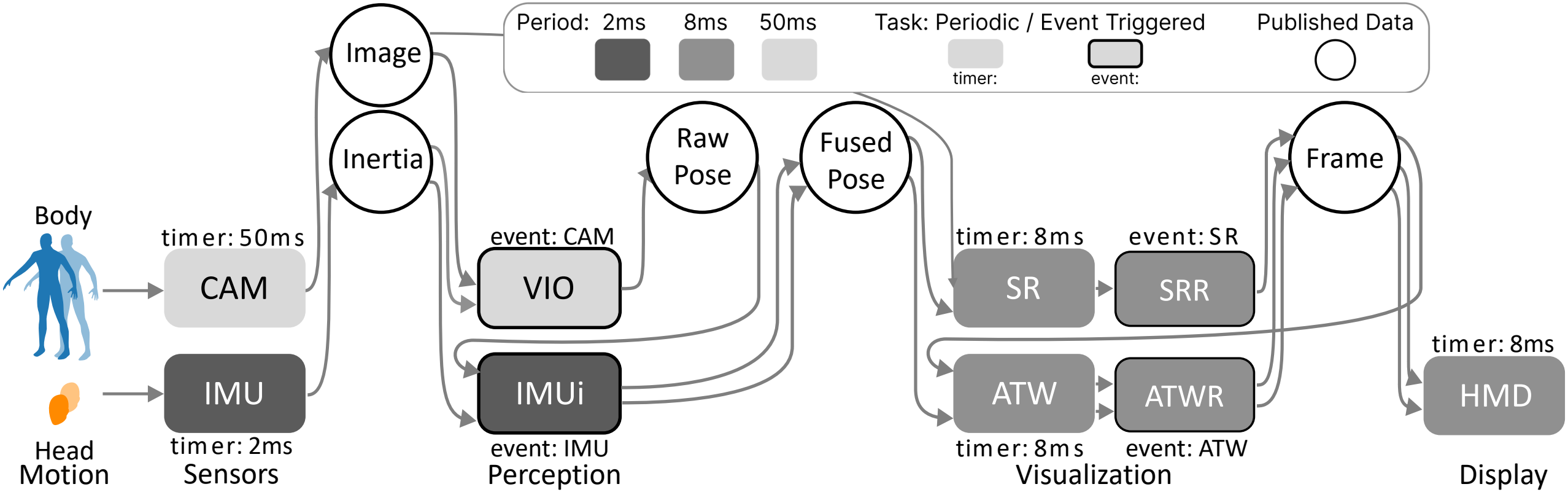
Body Motion

- Less frequent but with absolute position change in the environment
- i.e., running, walking, skiing, acceleration in car
- Requires **CAM** to perform accurate localization



State-of-the-art XR System Model: ILLIXR*

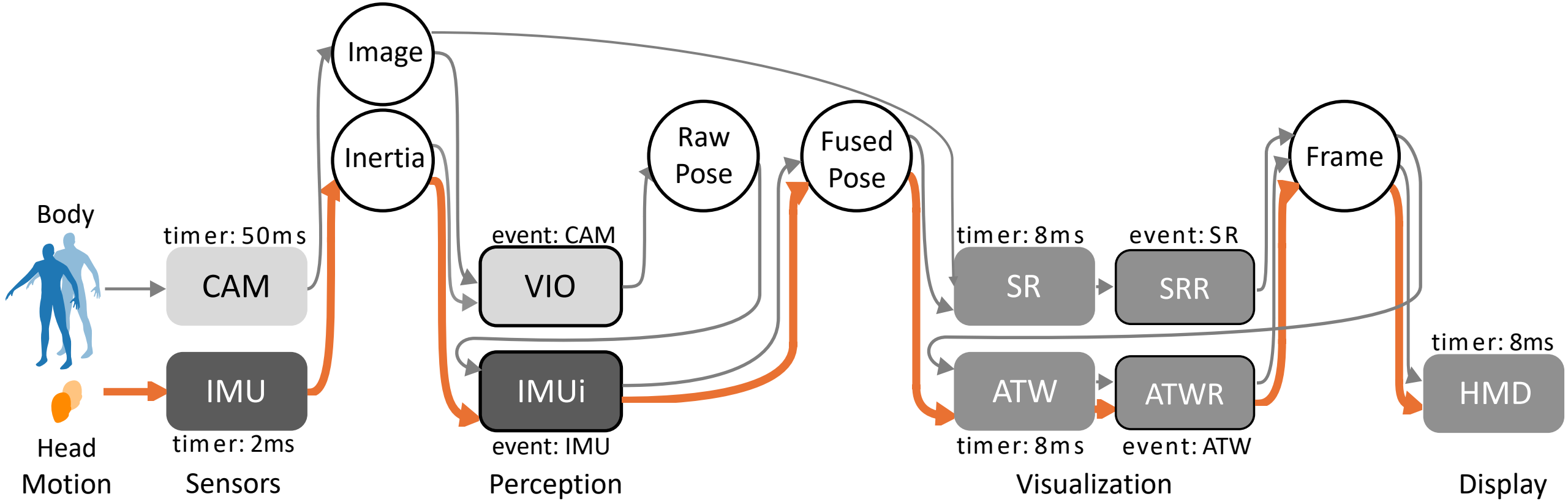
- Adopts a publisher-subscriber model
- Tasks: Visual Inertial Odometer (**VIO**), IMU integration (**IMUi**), Scene Reconstruction (**SR & SRR**), Asynchronous Timewarp (**ATW & ATWR**)



* Huzaifa, Muhammad, et al. "ILLIXR: Enabling end-to-end extended reality research." 2021 IEEE International Symposium on Workload Characterization (IISWC), 2021.

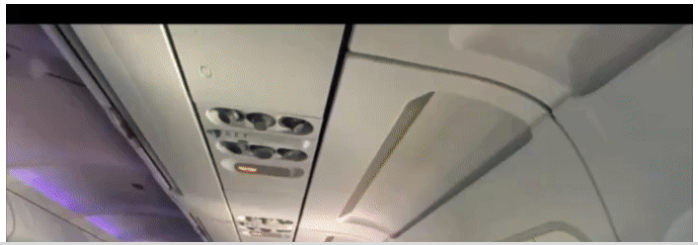
Existing metric: Motion-to-Display latency (M2D)

- Time between the **latest motion by the IMU** and its corresponding display on HMD
- **Why use M2D**? Most cases users just sit or stand idle without body motion

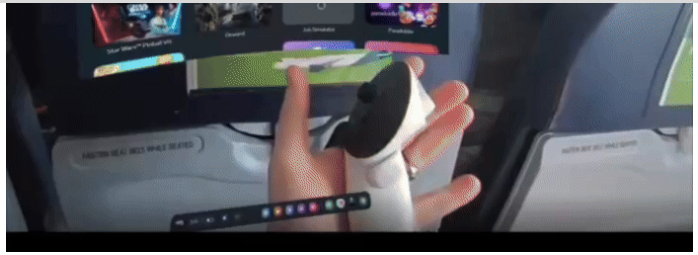


What if we only optimize M2D?

- Existing work achieves less than 20ms M2D, but when large body motion involved…



- Only **head motion** is accounted for in M2D
- **Body motion** is **NOT** considered in M2D metric and thus not optimized



Meta Quest 3 Taking Flight*

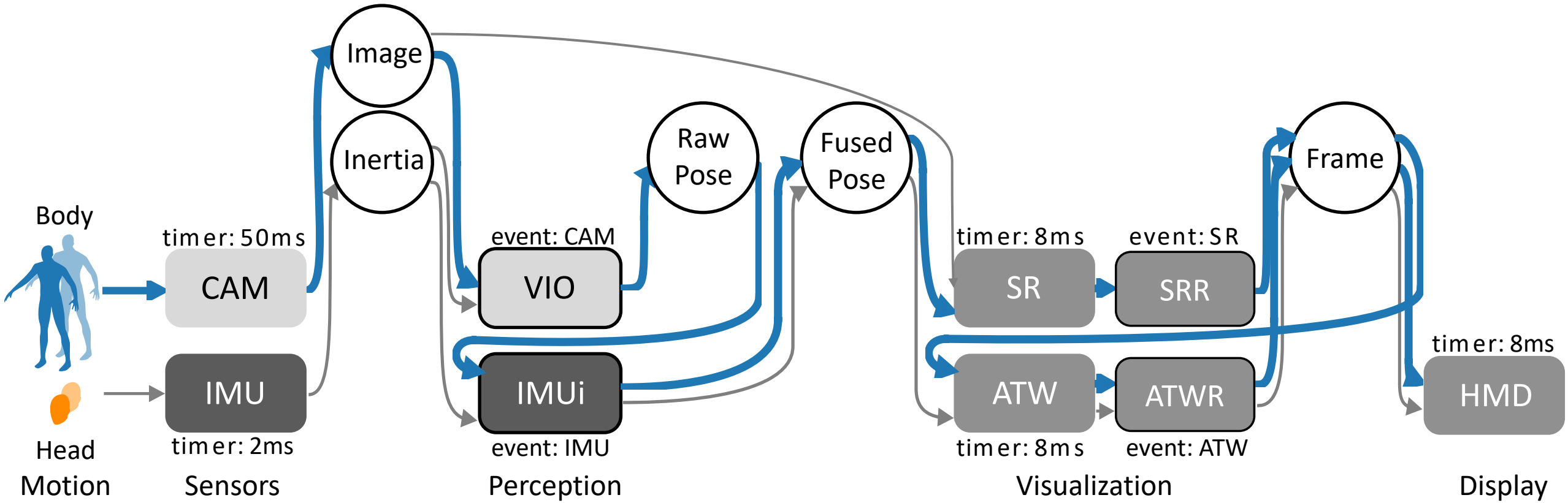
Apple Vision Pro Mountain Skiing**

- UI is supposed to move along with the user, but it drifts towards to edge of view!

* HistoricBeans, "Quest 3 in flight (passenger video)." https://www.reddit.com/r/OculusQuest/comments/17a8ctd/quest_3_in_flight_passenger_video/
** E. Cosner, "Snowboarding with apple vision pro - can this actually work? " https://youtu.be/LTVuGsYY2yc.

Introducing: Camera-to-Display latency (C2D)

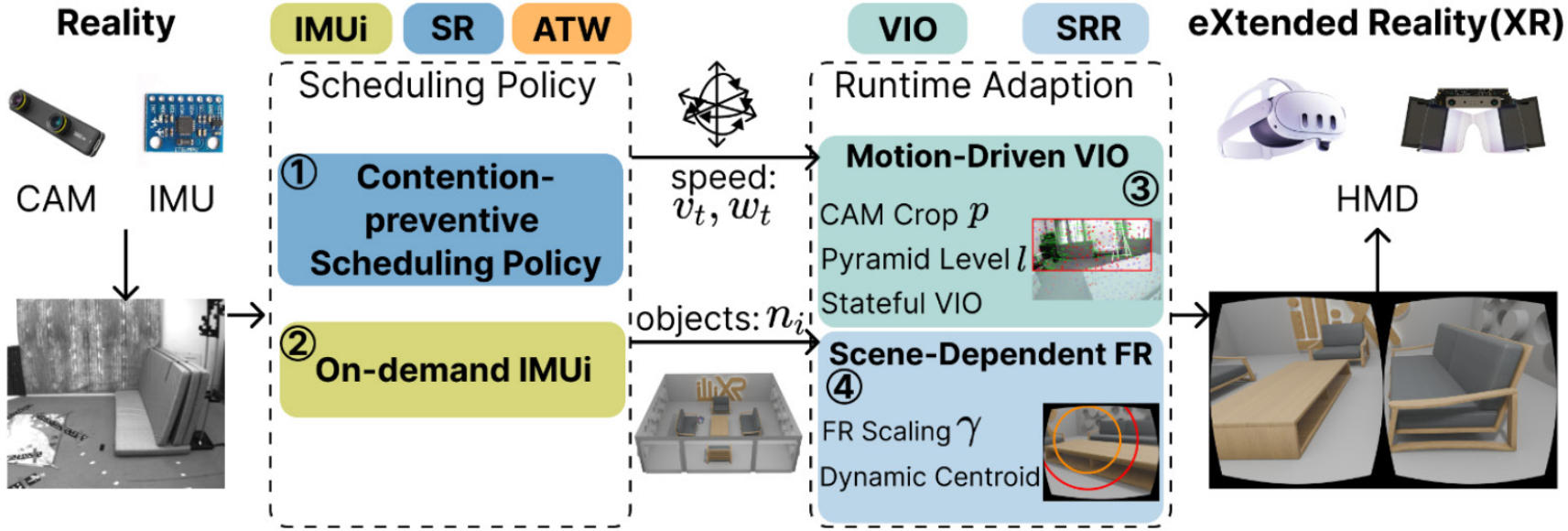
Time between the **latest body motion by CAM** and its corresponding display on HMD



Our Solution: BOXR

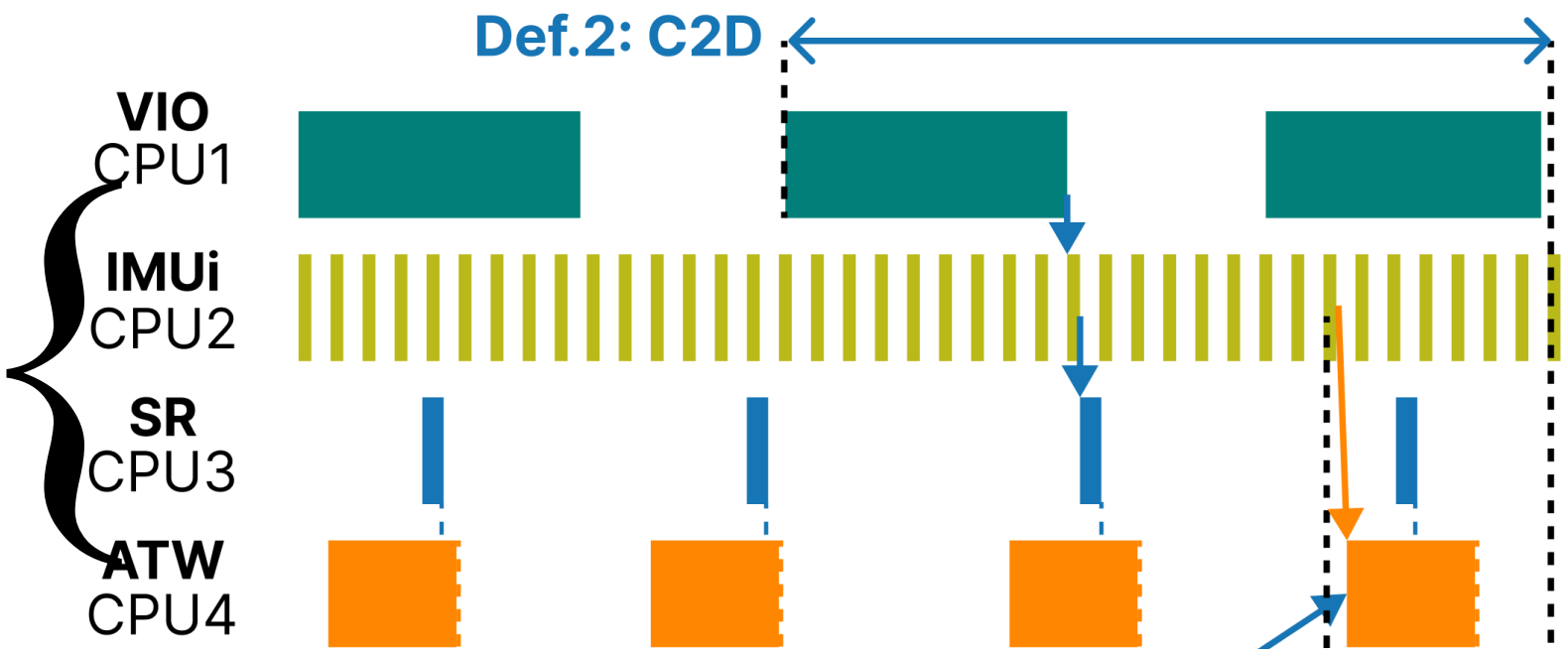
BOXR: Body and Head Motion **O**ptimization Framework for e**X**tended **R**eality

- Introduces the C2D metric to capture the delay from body motions
- Co-optimizes the delay from both body and head motions (M2D & C2D).
- Addresses unique challenges in XR systems:
  - Resource contention, optimization tradeoffs, and varying execution time due to motion and scene dynamics



System bottleneck

- XR adopts **publisher-subscriber** model
- **CPU** tasks: VIO, IMUi, SR, ATW

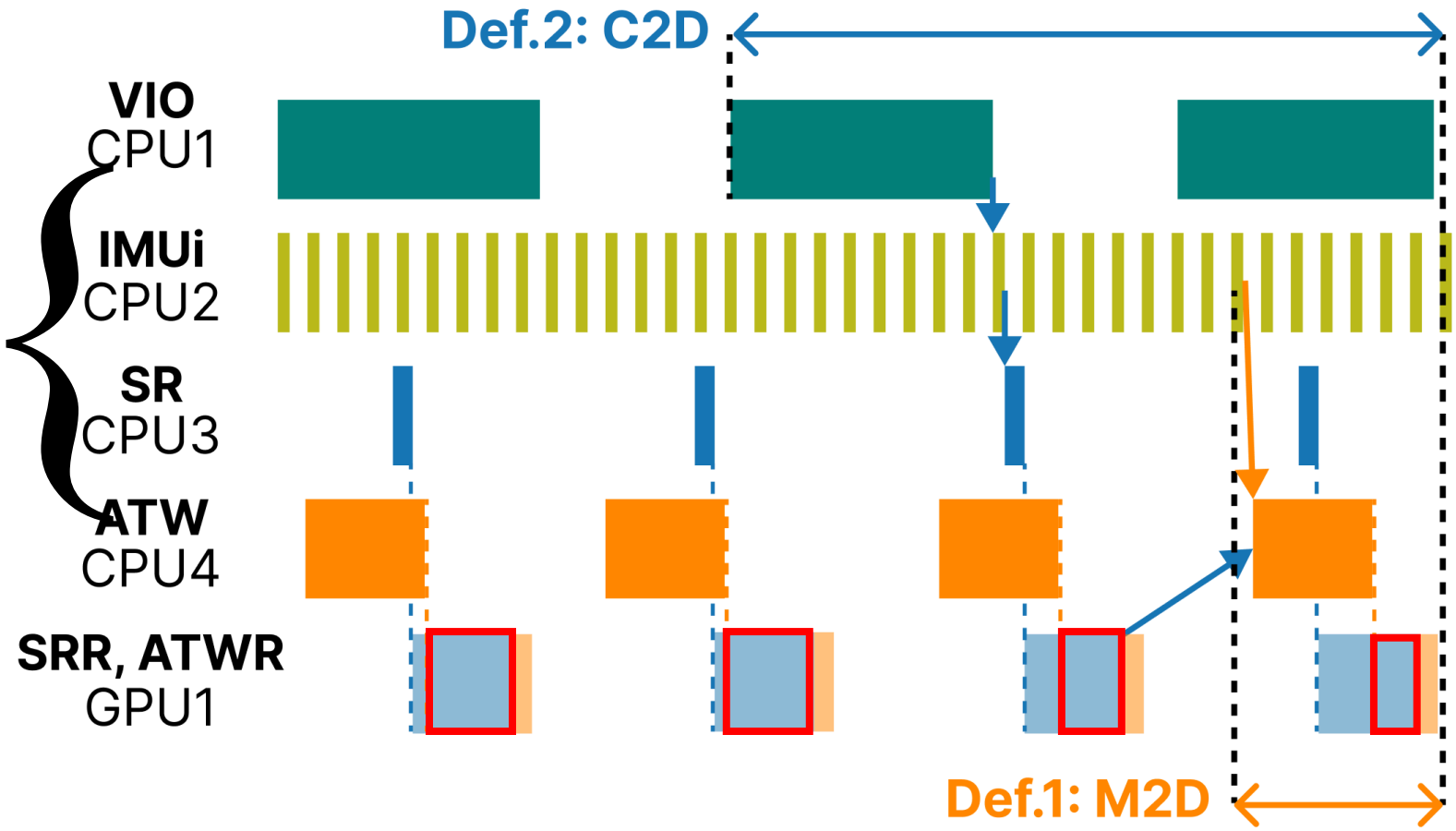


VIO	Visual Inertial Odometer
IMUi	IMU Integration

SR	Scene Reconstruction
ATW	Asynchronous Timewarp

System bottleneck

- XR adopts **publisher-subscriber** model
- **CPU** tasks: VIO, IMUi, SR, ATW
- **GPU** tasks: SRR, ATWR



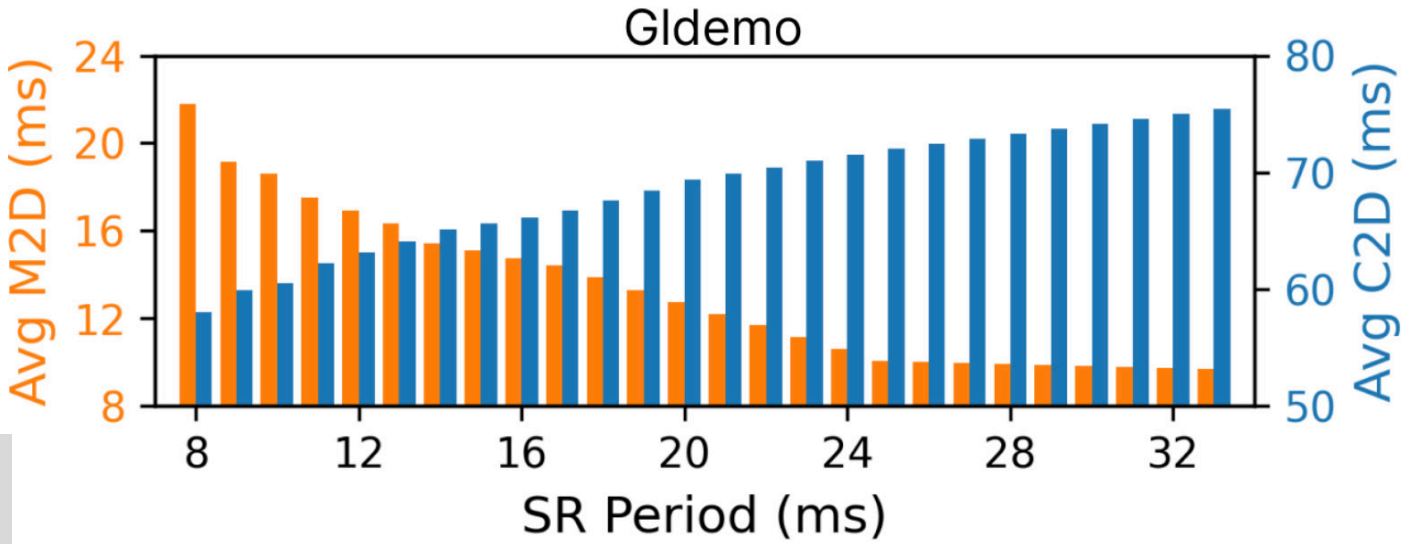
- **Contention** Between **SRR** and **ATWR**: Both tasks share the same stream and execute sequentially in a first-in-first-out order

M2D and C2D Tradeoff

- Single Optimization of M2D: increase SR period to mitigate contention

- However, C2D increases because less SR is executed: freshness of CAM data decreases

SR	Scene Reconstruction
ATW	Asynchronous Timewarp

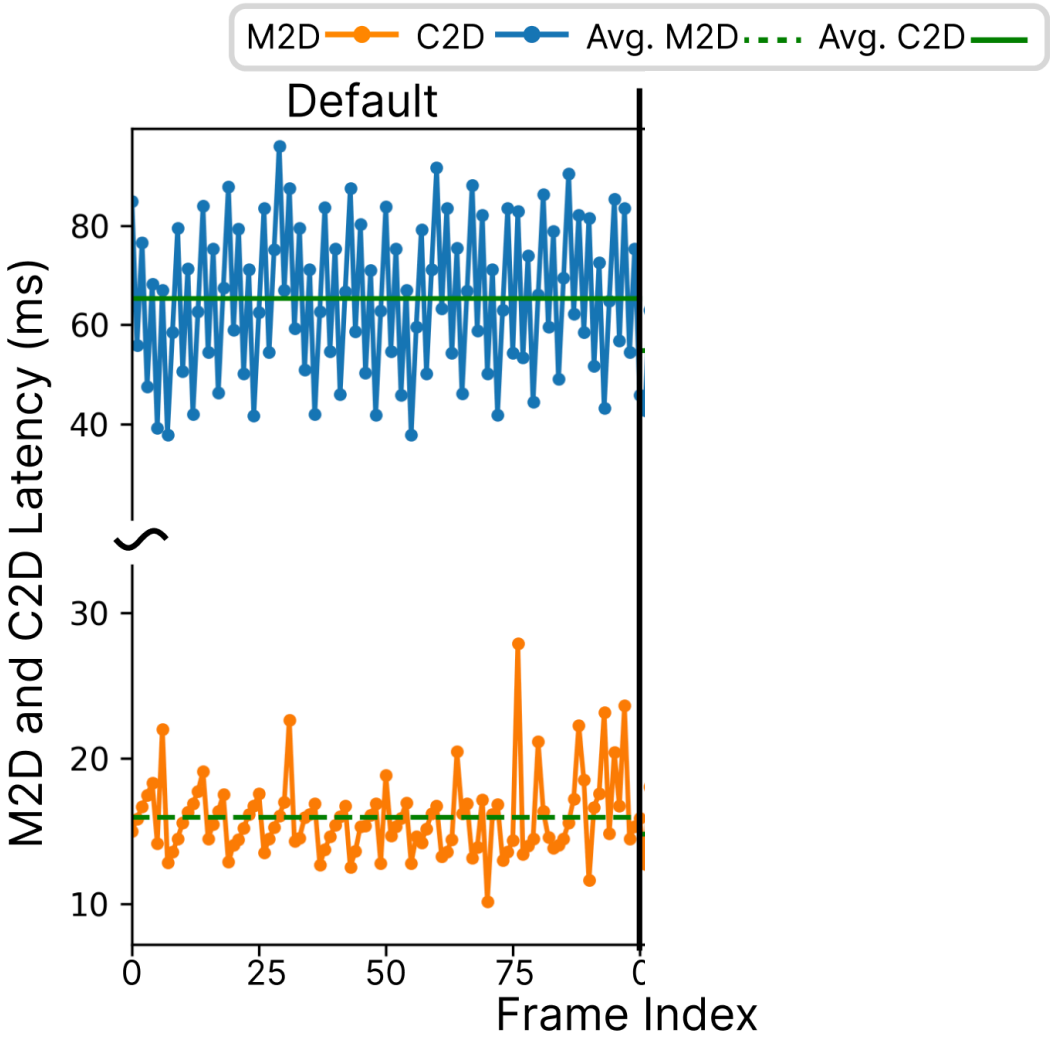


We need to consider both M2D and C2D during optimization

Simple Optimization of M2D and C2D does not work

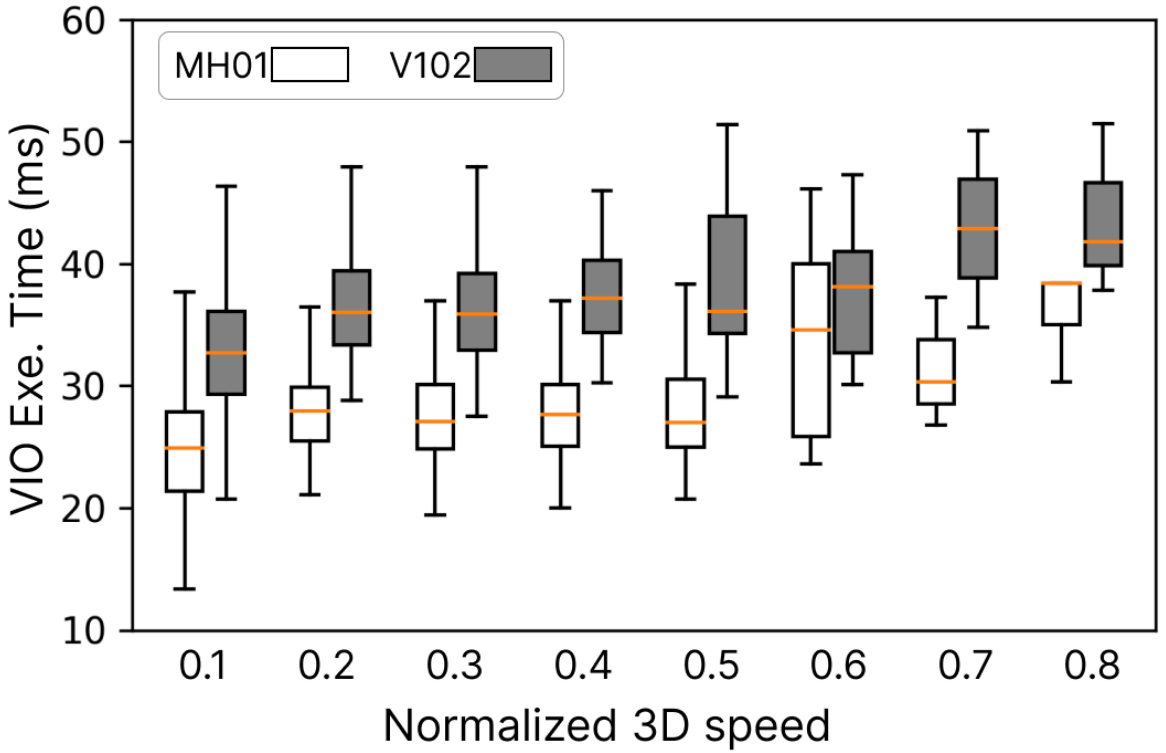
- Adjusting both SR and ATW period to minimize both average M2D and C2D
- But M2D and C2D fluctuations **increase**!
- Some Jobs have longer blocking time due to **contention**

Need to have scheduling policy that **prevents contention**!

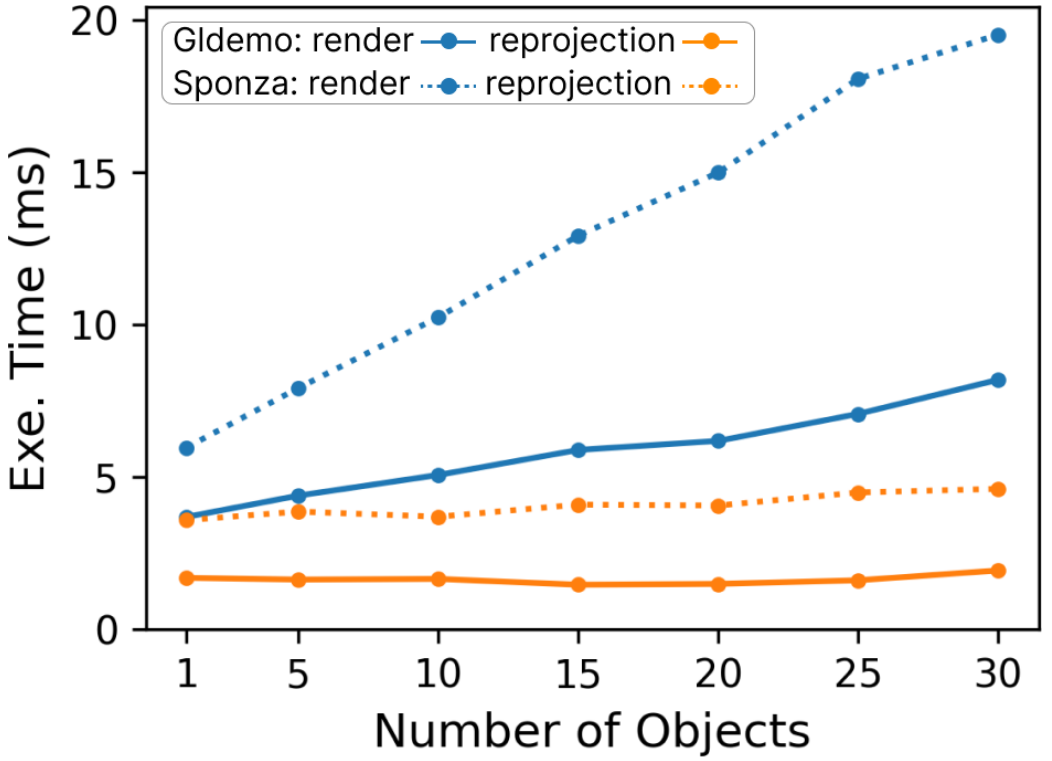


XR runtime characteristics

- **C1:** Motion-dependent Tracking: Larger speed increases VIO exec. time
- **C2:** Scene-dependent Rendering: More objects increase SRR exec. time (render)



C1: VIO Execution time increases with motion



C2: SRR Exe. time increases with Obj.

BOXR Overview

Why design a scheduler and runtime adaption

- Scheduling policy can be generated during setup to avoid large runtime overhead
- Motion and scene objects can only be acquired during runtime

BOXR Scheduling policy

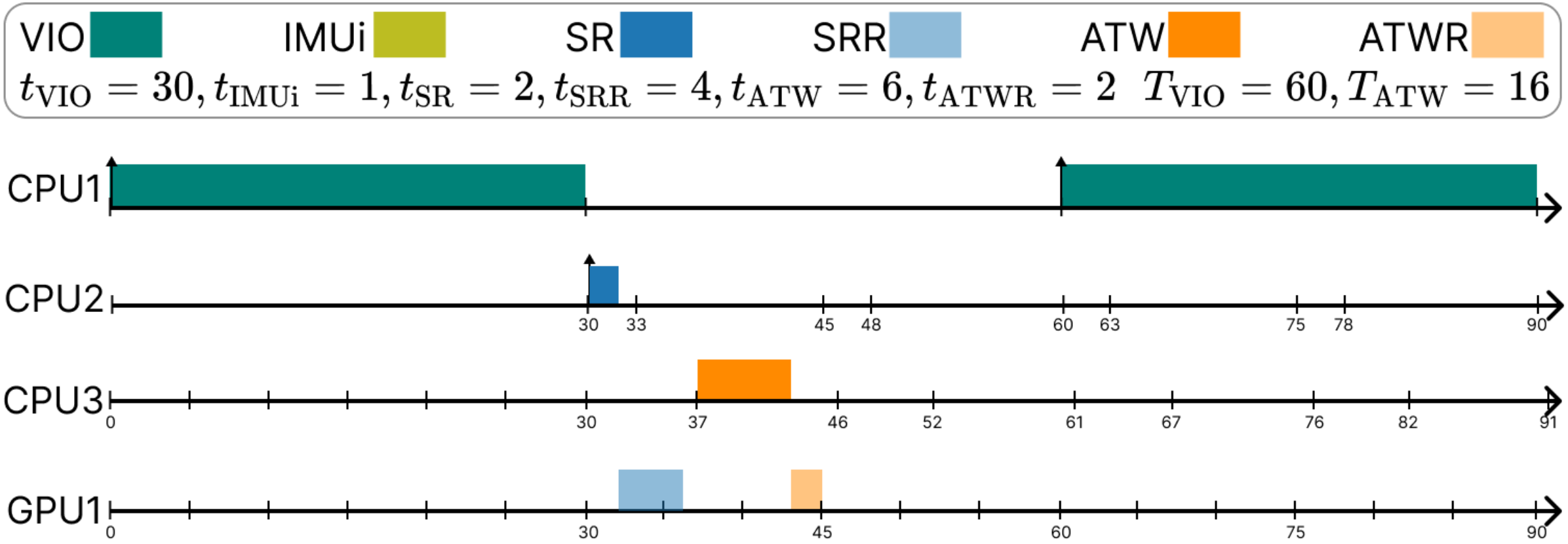
- Contention-preventive scheduling: Prevents contention between SRR and ATWR while maintain the fresh raw pose information
- On-demand IMUi: Executes IMUi only when fused pose is needed

BOXR Runtime Adaption

- Motion-Driven VIO: Addresses **C1:** motion-dependent tracking
- Scene-Dependent Foveated Rendering: Addresses **C2:** scene-dependent rendering

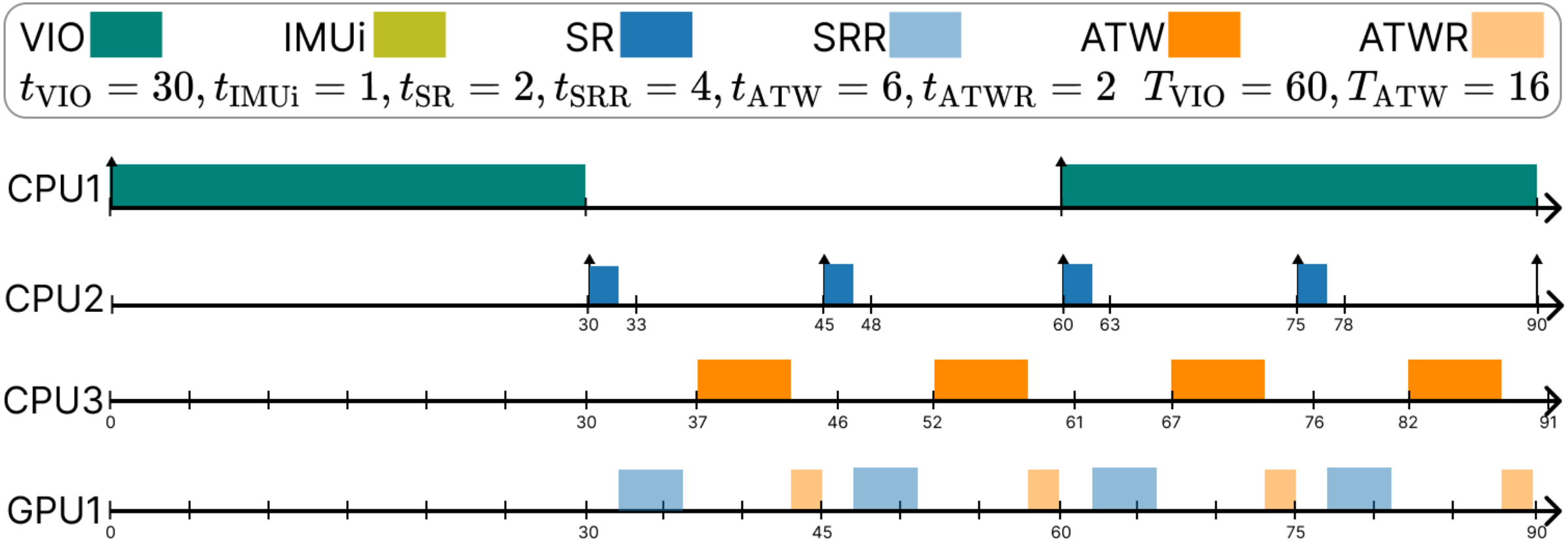
BOXR Scheduling policy: Contention-preventive

- To avoid contention between ATWR and SRR, BOXR schedules ATW and ATWR jobs synchronously to SR and SRR jobs



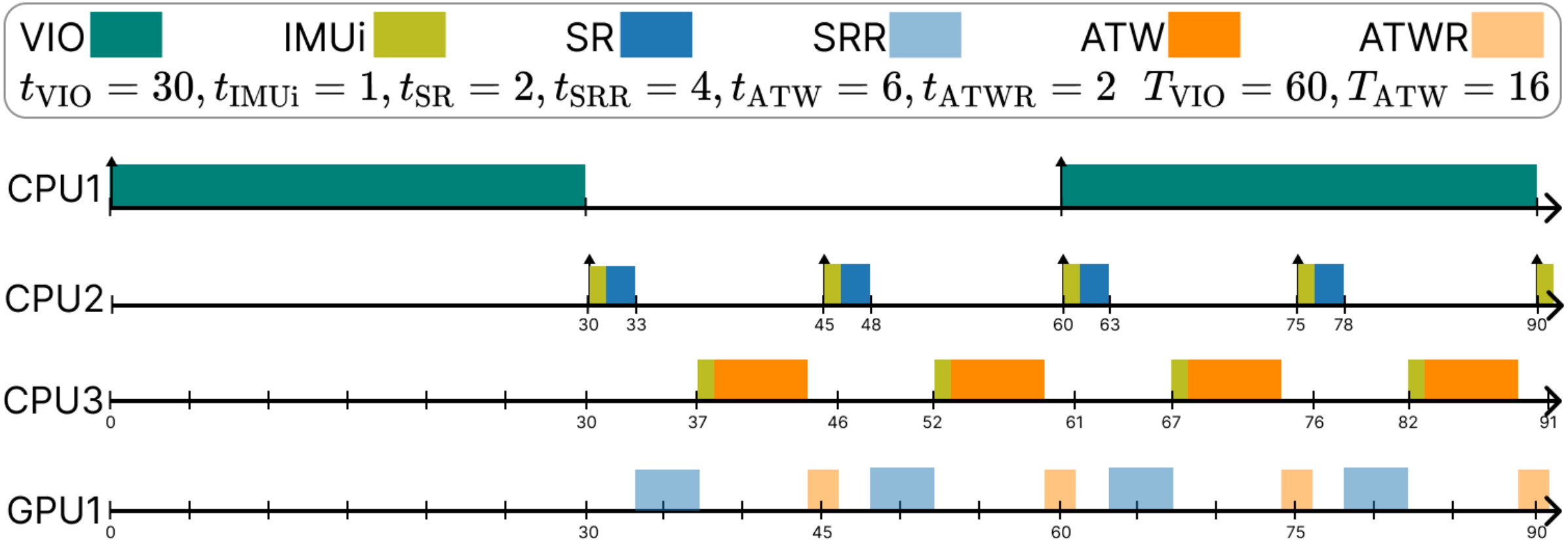
BOXR Scheduling policy: Contention-preventive

- To maintain raw pose data freshness, BOXR determines the SR period that ensures each C2D sequence (VIO→IMUi→SR→SRR→ATW→ATWR) completes with no temporal overlap



BOXR Scheduling policy: On-demand IMUi

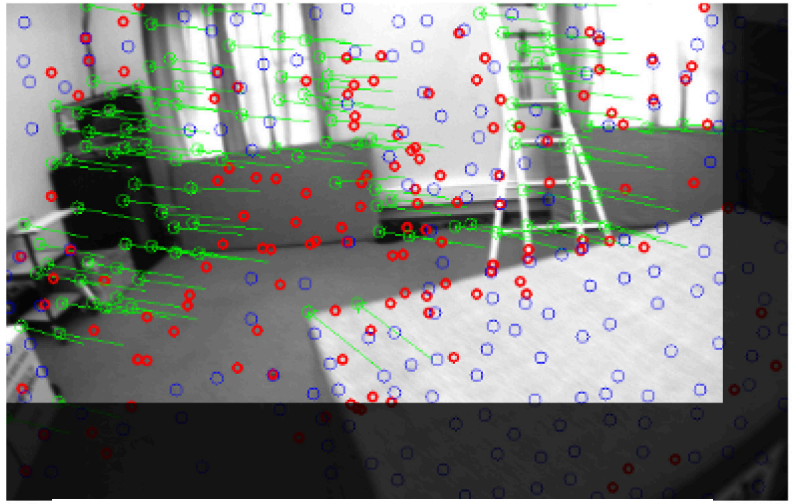
- To avoid wasted IMUi jobs, BOXR schedules IMUi to run only at the beginning of SR or ATW jobs

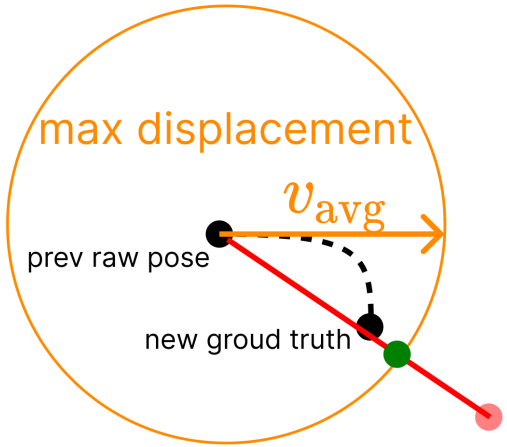


Runtime Adaption: Motion-Driven VIO

Goal: Manage the varying execution time of VIO while minimizing pose quality degradation

- **Image cropping**: when motion is large, crop the image input
- **Level of pyramid adjustment**: adjust the pyramid level in the VIO's Multi-state Kalman Filter w.r.t motion
- **Error bounding**: leverage the previous raw pose to calculate the maximum possible displacement. Bound the output pose within the displacement.

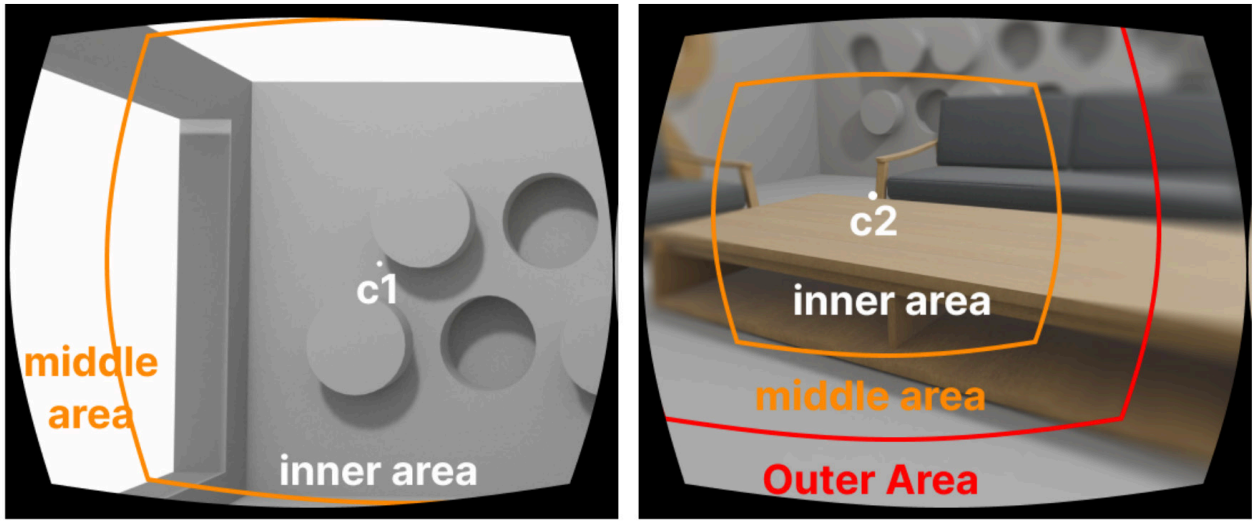




Runtime Adaption: Scene-Dependent Foveated Render

Goal: Control varying SRR execution time by scene-adaptive foveation

- **Dynamic foveation area**: reduce the peripheral resolution when more objects are in the viewport
- **Dynamic Objects Centroid**: fix the center of foveation to the objects' centroid to effectively reduce render time



Evaluation and Baselines Setup

Setup:

- Hardware Platform: PC, NVIDIA AGX Xavier, NVIDIA Orin Nano
- XR applications: Table I
- Trajectory: Table II

TABLE I: Evaluated XR Applications

	Sponza(Spon)	**Materials**(Mat)	**Gldemo**(Gl)	**Platformer**(Plat)
Object	32	81	7	3014
Vertex	192870	62826	54760	26168
Texture	33	24	8	4

TABLE II: EuRoc MAV Dataset Categorization

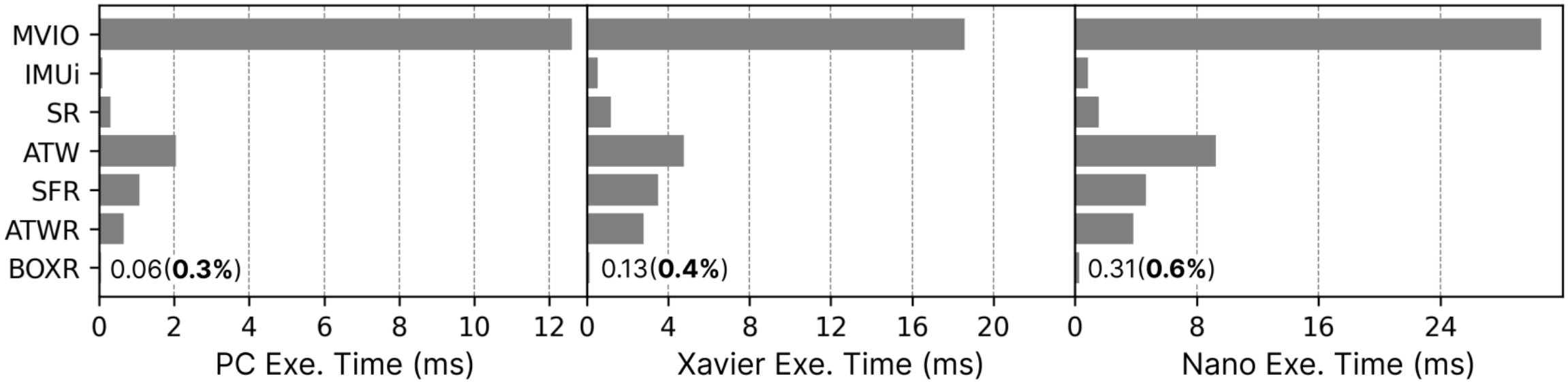
	No Motion	**Small Motion**	**Med. Motion**	**Large Motion**
$v_{3D}(m/s)$	$[0, 0.1)$	$[0.1, 1)$	$[1, 2)$	$[2, \infty)$
Perc(%)	10.62	66.75	20.30	2.33

Baselines:

- ILLIXR: Pub-Sub model
- ILLIXR-OP: Optimized with SR and ATW periodicities changes
- BOXR-S: Static BOXR only implements the BOXR scheduler
- BOXR: Complete framework with both scheduler and runtime adaption

Efficiency: Low Runtime Overhead

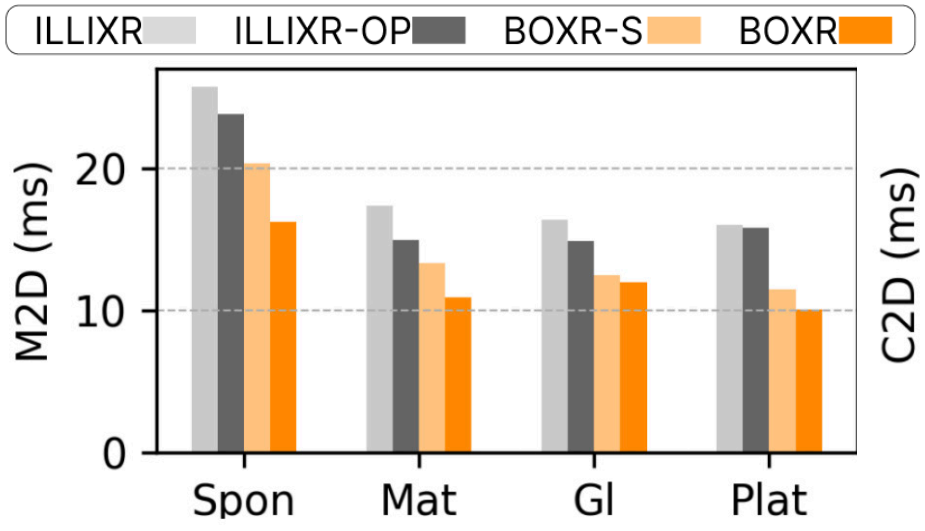
- BOXR runtime adaption only results **up to 0.6%** of system overhead

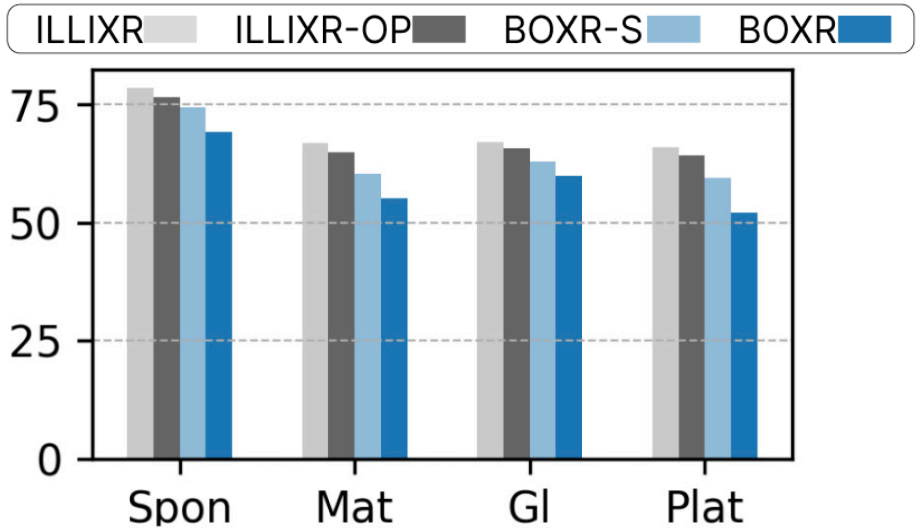


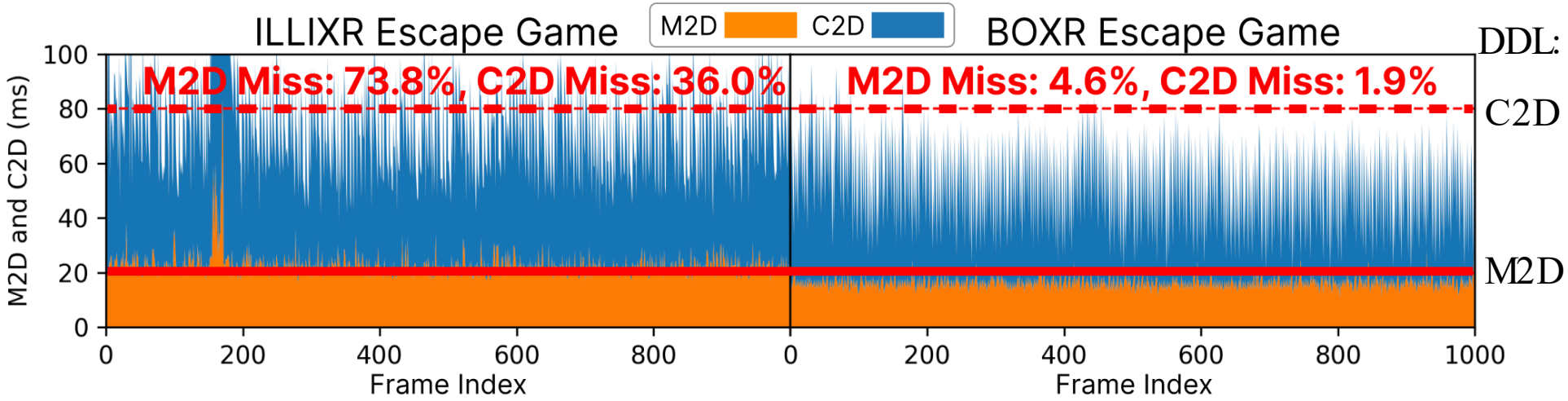
Effectiveness: M2D and C2D Reduction

BOXR General **Effectiveness**:

- Xavier: on average **37%** M2D and **27%** C2D reduction
- Real-world Gaming: BOXR reduces M2D and C2D miss rate by **69.2%** and **34.1%** respectively







Conclusion

- Discovers the C2D metric and co-optimize both M2D and C2D for both body and head motion
- Addresses resource contention and increased execution time due to motion and objects

GitHub Link

Paper Link