Distributed Video Slow Motion using Scene-Based Frame Interpolation

Team Members

Anisha Priyadarshini Mohanty (apmohant@uci.edu)

Venkata Bharath Reddy Reddem (vreddem@uci.edu)

Maitreyi Sinha (maitres@uci.edu)

Background

Problem: Most videos are captured at 24-30 FPS, which is too low for smooth motion or slow-motion playback

One solution:
Frame interpolation generates new frames between existing ones to produce high-FPS, smoother video

- Modern deep learning models: RIFE create high-quality results but are **computationally expensive**

Videos have scenes with varying motion making some segments much costlier to process
- Static scenes are cheap to process
- Fast-motion scenes require more intermediate frames

→

This variability makes interpolation **slow and imbalanced** on a single machine or with naive equal splitting

Introduction

- Video processing is **computationally expensive**.
- High quality slow motion video requires frame interpolation.
- Interpolating entire video sequentially is time consuming for long videos.
- Running the entire pipeline on a single machine leads to slow processing times.
- Distributing work across multiple nodes can significantly improve throughput.
- Motion complexity varies across scenes, making load balancing across workers difficult.

Motivation

- Slow motion is widely used in sports, video editing, and visual effects.
- Scenes in a video vary widely in motion complexity.
- Motion intensity varies across each set of frames:
  - High motion → more frames needed
  - Low motion → fewer frames suffice
- Distributed processing reduces computation time.
- Goal
  - Develop a distributed system to generate slow-motion videos efficiently.
  - Adaptively interpolate different number of frames per scene based on motion intensity.

$$\text{motion\_intensity} = \frac{1}{N-1}\sum_{i=1}^{N-1} \|F_{i+1} - F_i\|$$

Related Works

- **Distributed Video Processing**
  - Simple frame splitting across workers.
  - Ignores motion complexity → severe load imbalance.

  Ref: Jeffrey, Huang, Nandra

- **Scene-Based Partitioning**
  - Splits video into scenes for semantic consistency.
  - Assumes equal cost per frame, inaccurate for interpolation tasks.

  Ref: Ahmed

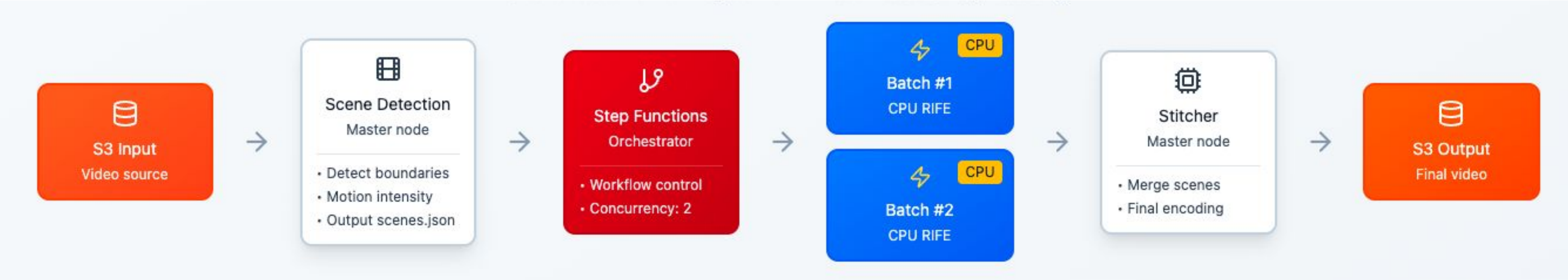
- **Load Balancing in Distributed Pipelines**
  - Traditional methods: static partitioning or round-robin
  - Often fail when task cost per frame varies significantly.

  Ref: Aghasoleymani

- **Video Frame Interpolation Systems**
  - Focuses on model accuracy/quality, not distributed scheduling.
  - Computational cost highly depends on motion intensity

  Ref: Jiang et al. (2018), Bao et al. (2019), Huang et al. (2022)

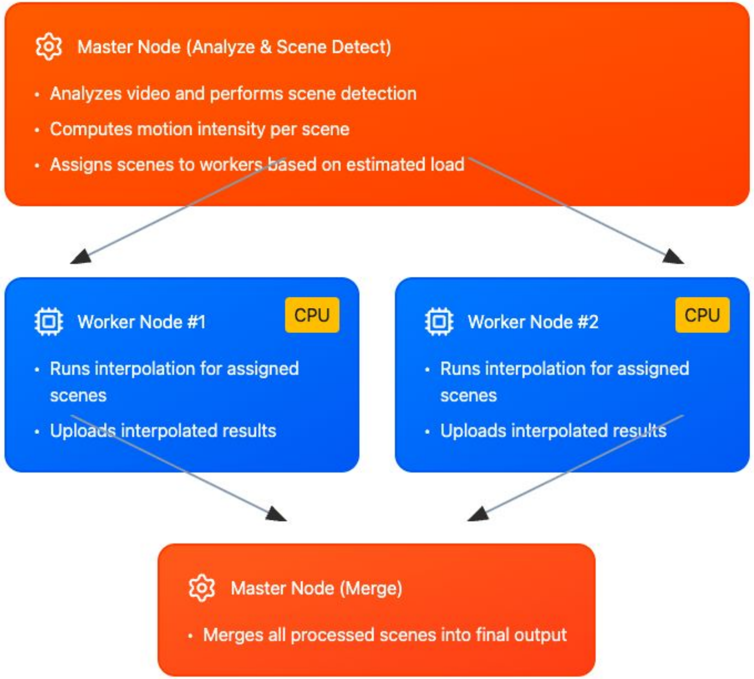
System Architecture: AWS Distributed Pipeline



- **Scene Detection:** Split video into multiple scenes based on scene change.
- **Motion Analysis:** Compute motion intensity per scene.
- **Adaptive Frame Interpolation:** More frames for high-motion scenes; fewer frames for low-motion scenes
- **Distributed Execution:** Assign scenes to worker nodes based on estimated load and run interpolation in parallel.

System Architecture Overview

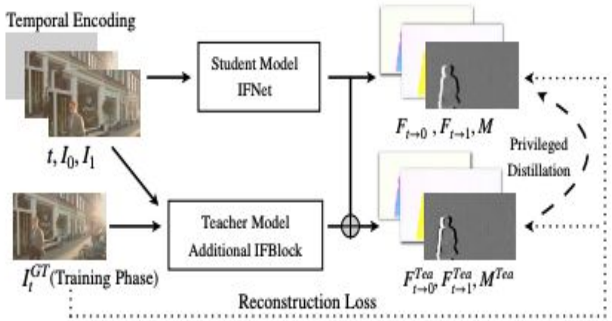
- Master node analyzes video and performs scene detection.
- Motion intensity computed per scene.
- Scenes assigned to worker nodes based on estimated load.
- Workers run interpolation and upload results.
- Master merges all processed scenes.



Key Features

- Adaptive interpolation per scene based on motion intensity.
- Parallel processing to reduce total computation time.
- Use RIFE interpolation model which uses IFNet for estimating intermediate flows.
- S3 integration for seamless distributed workflow.
- Dynamic load balancing ensures optimal utilization of all worker nodes.
- Modular pipeline design makes the system easily scalable to larger video datasets.

RIFE model architecture

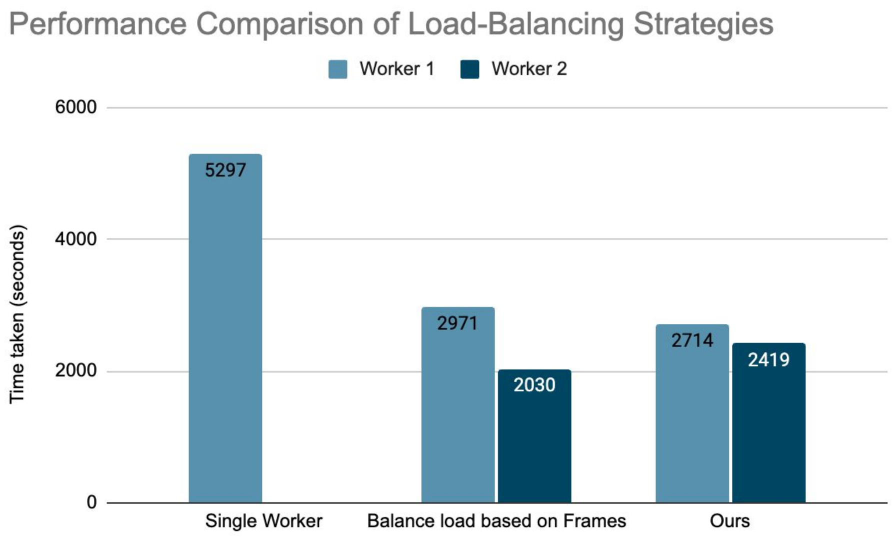


Understanding Our Processing Pipeline

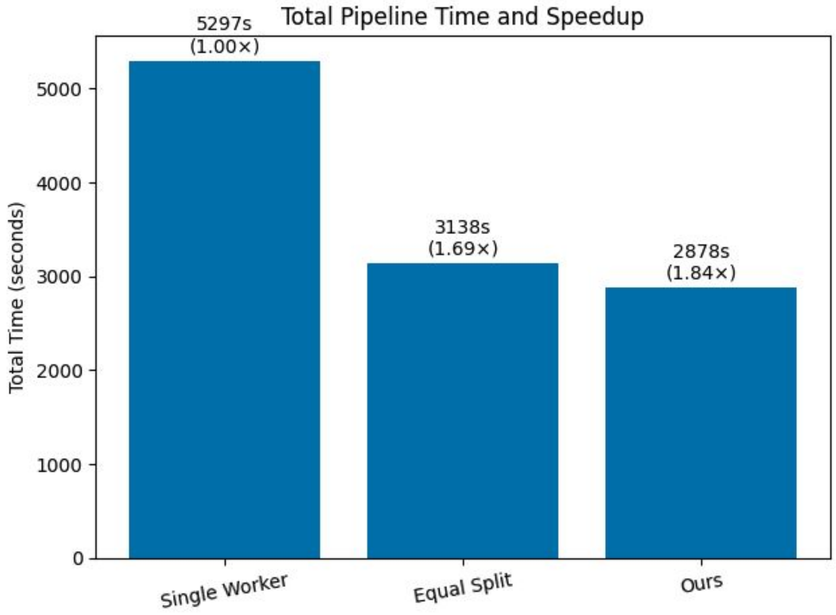
- Scene Detection
  - Histogram based scene boundary detection.
  - Automatically finds changes in scene transitions.
  - Outputs timestamps where new scenes begin.
  - Reduces unnecessary computation on static segments.

- Motion-Adaptive Load Balancing
  - Compute motion intensity for each scene.
  - Estimate number of interpolated frames needed.
  - Greedy load assignment ensures better balance.
  - Workers process nearly equal workload.

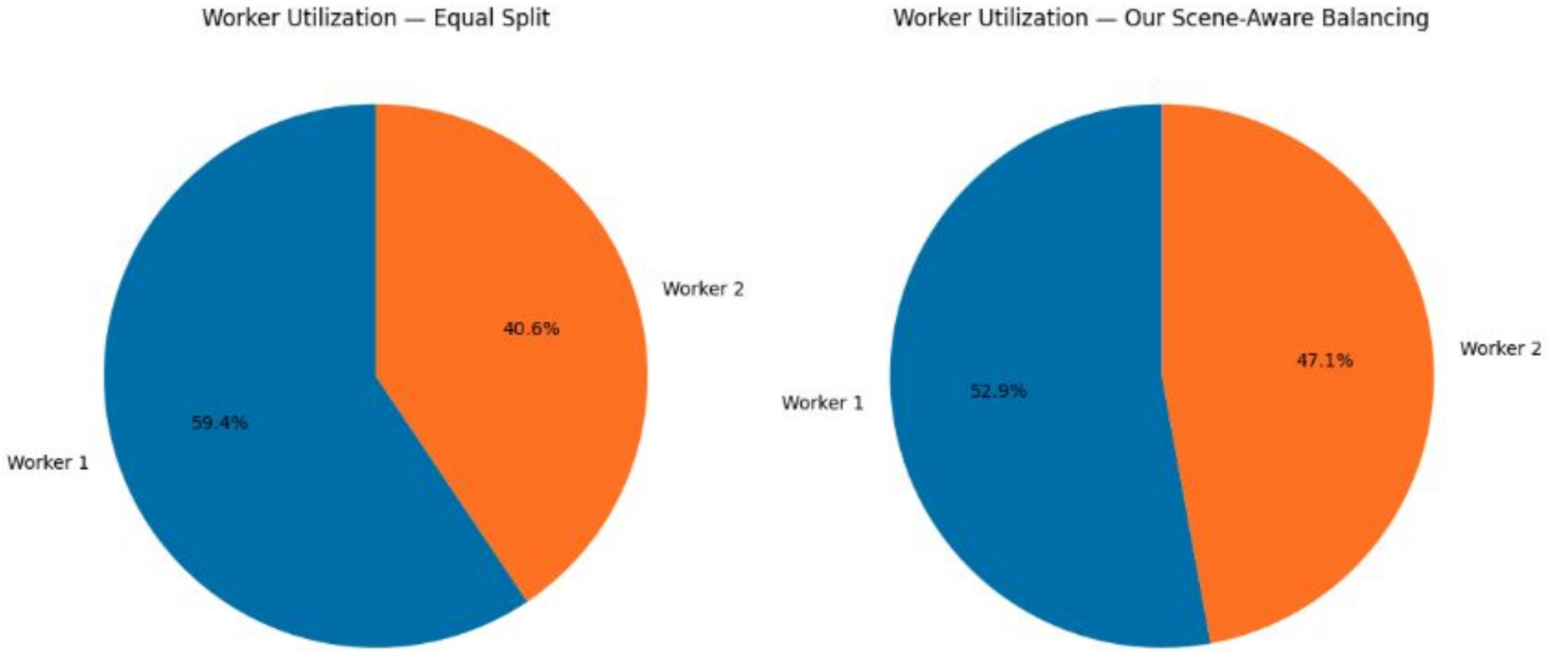
Results



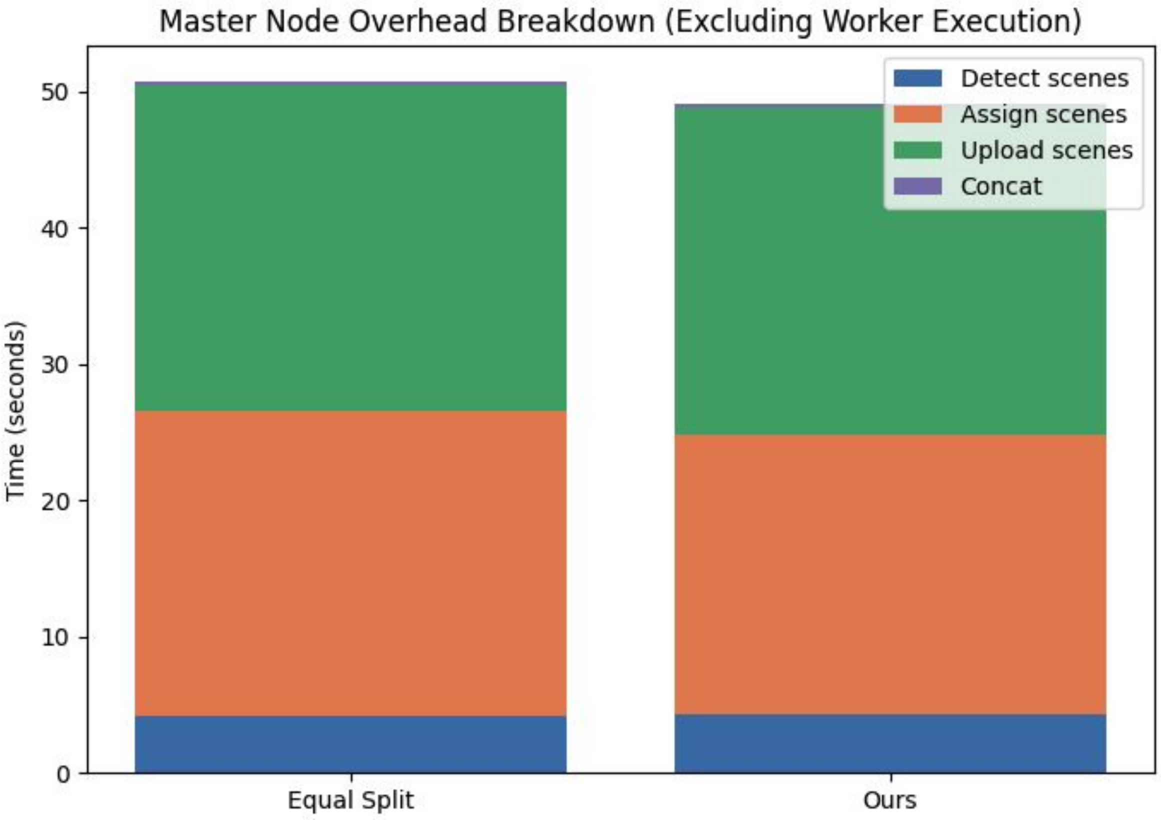
Performance Comparison of Load-Balancing Strategies
Worker 1
Worker 2
6000
4000
2000
0
Time taken (seconds)
5297
2971
2030
2714
2419
Single Worker
Balance load based on Frames
Ours



Total Pipeline Time and Speedup
5297s
(1.00×)
3138s
(1.69×)
2878s
(1.84×)
Total Time (seconds)
5000
4000
3000
2000
1000
0
Single Worker
Equal Split
Ours



Workers are utilized evenly with scene-based load assignment



Scene detection and dynamic assignments add almost no overhead.
(< 2% of processing time)

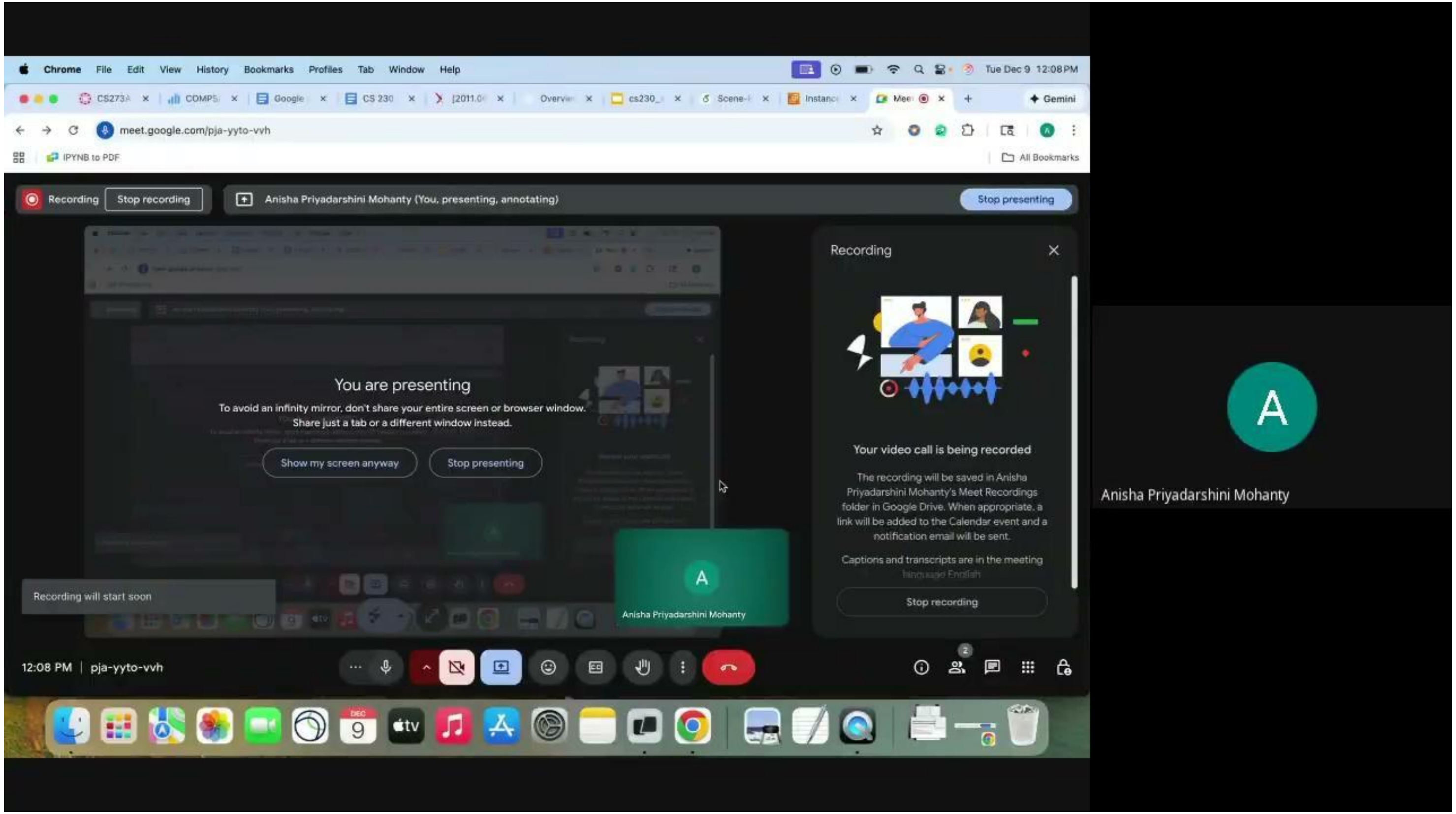
Conclusion

- Designed and implemented a distributed system that splits tasks across multiple worker nodes for faster processing
- Demonstrated that parallel execution significantly improves throughput, with measurable gains even on a two-node setup
- Identified key system challenges, including load imbalance, synchronization overhead, and worker heterogeneity

Future Work

- Extend support for heterogeneous hardware (CPUs, GPUs, accelerators)
- Add auto scaling based on workload intensity
- Integrate detailed monitoring (metrics, logs, tracing) for deeper performance analysis
- Implement fault-tolerance features such as task reassignment, checkpointing, and worker health checks

Demo



Chrome File Edit View History Bookmarks Profiles Tab Window Help
Tue Dec 9 12:08 PM
meet.google.com/pja-yyto-vvh
IPYNB to PDF
All Bookmarks
Recording
Stop recording
Anisha Priyadarshini Mohanty (You, presenting, annotating)
Stop presenting
You are presenting
To avoid an infinity mirror, don't share your entire screen or browser window.
Share just a tab or a different window instead.
Show my screen anyway
Stop presenting
Recording will start soon
Anisha Priyadarshini Mohanty
Recording
Your video call is being recorded
The recording will be saved in Anisha Priyadarshini Mohanty's Meet Recordings folder in Google Drive. When appropriate, a link will be added to the Calendar event and a notification email will be sent.
Captions and transcripts are in the meeting
Stop recording
12:08 PM | pja-yyto-vvh
A
Anisha Priyadarshini Mohanty

Thank You! Questions?

References

- Wenbo Bao, Wei-Sheng Lai, Chao Ma, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang. Depth-aware video frame interpolation. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 3703–3712, 2019.
- Zhewei Huang, Tianyuan Zhang, Wen Heng, Boxin Shi, and Shuchang Zhou. Real-time intermediate flow estimation for video frame interpolation. In European Conference on Computer Vision, pp.624–642. Springer, 2022.
- Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan Kautz. Super slomo: High quality estimation of multiple intermediate frames for video interpolation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 9000–9008, 2018.
- Jeffrey Dean and Sanjay Ghemawat. MapReduce: Simplified data processing on large clusters. In *Proceedings of the USENIX Symposium on Operating Systems Design and Implementation*, pp. 137–150, 2004.
- Qi Huang, Petchean Ang, Peter Knowles, Tomasz Nykiel, Iaroslav Tverdokhlib, Amit Yajurvedi, Paul Dapolito IV, Xifan Yan, Maxim Bykov, and Chuen Liang. SVE: Distributed video processing at Facebook scale. In Proceedings of the 26th ACM Symposium on Operating Systems Principles (SOSP), pp. 87–103, 2017.
- Nandra, Constantin, Victor Bacu, and Dorian Gorgan. Distributed, workflow-driven rendering of 3D object scenes on a big data processing platform. In Proceedings of the 22nd International Conference on System Theory, Control and Computing (ICSTCC), pp. 94–99, 2018.
- Aghasoleymani Najafabadi, Saeid.Performance Assessment of Load Balancing Methods in Cloud Computing: Analysis of Round Robin, Equally Spread, and Throttled Strategies Using Cloud Analyst. arXiv preprint arXiv:2507.11899, 2025.Ahmed Hassanien, Mohamed Elgharib, Ahmed Selim, Sung-Ho Bae, Mohamed Hefeeda, and Wojciech Matusik. Large-scale, fast and accurate shot boundary detection through spatio-temporal convolutional neural networks. arXiv preprint arXiv:1705.03281, 2017.