Integrating storage manager (TASM) and query optimizer (FiGO) in Video-DBMS

Student: Chirag Parikh (2022900005) **CSE, IIIT-H**
Instructor: Prof. Kamal Karlapalem **Data Systems Project**
Project Video: YouTube-link **Project Code: GitHub-repo**

---

1 Introduction

Video data is increasingly used across domains such as surveillance, autonomous systems, and content analysis [1]. Efficiently querying and retrieving objects from videos remains a significant challenge due to the large volume of data and the computational cost of running object detection models repeatedly. Traditional approaches either rely on storing dense object metadata extracted by running expensive object detection model ahead of time—resulting in high initial compute overhead—or run query optimizers (like FiGO [2]) using an ensemble of models during inference, leading to high latency.

To address this trade-off between initial compute overhead and per-query latency, we propose an integrated solution combining FiGO [2]—a query optimizer that uses an ensemble of object detection models—and TASM [3], a storage manager that supports semantic indexing with chunked video storage. Our system intelligently routes queries through FiGO or TASM based on prior query history and dynamically builds semantic indexes to accelerate future queries.

The proposed approach also supports automated retrieval of all object types detected in a video, eliminating redundant model executions and enabling scalable, low-latency query processing for diverse videos. By extending TASM to operate efficiently on consumer-grade GPUs and restructuring video storage into 8-frame sequential chunks, we ensure broad applicability and robust performance across hardware setups.

1.1 Main Contributions

This work makes the following key contributions:

- **FiGO and TASM Integration:** We integrate the FiGO query optimizer [2] with TASM [3], enabling significant reductions in query execution time for repeated retrievals of the same object type.
- **Automated Object Type Retrieval:** We introduce an automated mechanism to retrieve and track all object types present in a video, eliminating the need for repeated FiGO executions for previously detected types.

- **Scalable Storage and Retrieval:** The system is designed to scale efficiently for both storage and fast object retrieval, especially suited for real-world applications involving long-duration, large scale CCTV videos.
- **Multi-Attribute Object Query Support:** Both FiGO and TASM now provide support for retrieving multiple object-level attributes, including cropped object pixels, frame-level object location, and object dimensions.
- **Consumer-Grade GPU Compatibility:** TASM is extended to work effectively on consumer-grade GPUs such as the NVIDIA RTX 20- and 30-series, ensuring broader accessibility.

2 Background

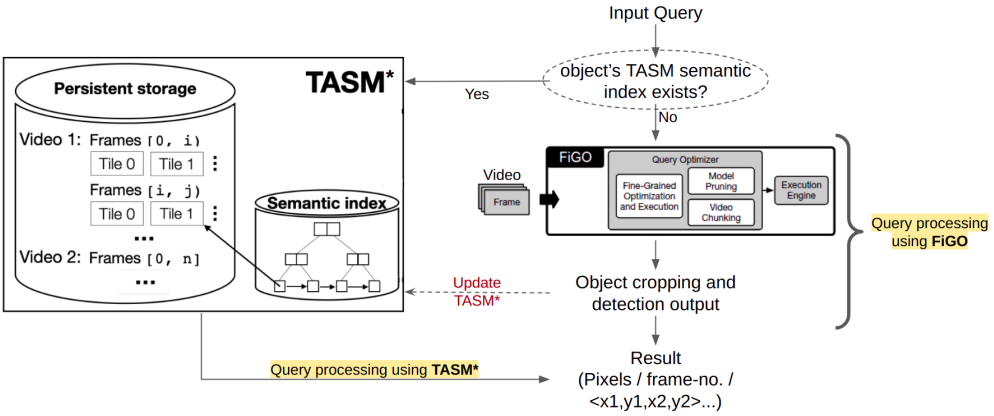
2.1 FiGO

Traditional techniques for Video Database Management Systems (VDBMSs), such as Model Specialization (MS) and Model Cascade (MC), are used to accelerate query execution by minimizing the computational load of running deep learning models on large volumes of video data. However, these approaches face significant challenges, including reduced accuracy on difficult-to-detect predicates (e.g. small object in the background), lower filtering efficacy with frequently-occurring objects in long videos, and missed optimization opportunities due to coarse-grained query planning for the entire video. These limitations hinder the ability to efficiently and accurately process video queries, particularly on large and diverse datasets, prompting the need for more robust solutions that can dynamically adapt to varying video contents and accuracy-throughput requirements.

FiGO addresses these challenges through a novel integration of fine-grained query optimization and a model ensemble approach. By dividing video data into variable-sized chunks, FiGO selects the most suitable model from an ensemble for each chunk, such that it delivers the lowest query execution time while meeting the target accuracy, ensuring adaptability to varying video contents and query complexity. Fine-grained query-optimization is achieved through chunking videos into different sizes until further chunking overhead exceeds the chunk execution overhead. The system also employs a Thompson sampling-based strategy to prune the ensemble for every chunk, focusing only on models that provide significant utility, which minimizes query-optimization overhead without significant loss in accuracy. This approach enhances adaptability and efficiency across different video segments, outperforming existing systems for processing queries over the entire videos by 3.3x on average across four video datasets.

2.2 TASM

The exponential growth in video data, driven by advances in camera technology and machine learning, has created new demands for efficient video data management systems. Traditional systems store videos as single encoded files, focusing mainly on frame processing post-decoding, which limits storage optimizations and significantly impacts query performance. Applications like automatic traffic analysis and retail planning require quick, efficient access to specific video content. However, the overhead involved in decoding entire video frames, even when only specific regions are needed, slows down the process and increases resource consumption, highlighting the need for a more efficient approach to video storage and query execution.



Update TASM* semantic index using detection output and store video in untiled 8-frame chunk form (unlike TASM) for faster retrieval of future queries

Figure 1: **Integrating TASM and FiGO in Video-DBMS.** As opposed to TASM, TASM* stores videos in 8-frame chunks instead of full video in untiled form. This enables faster retrieval of relevant frames as per input object query.

Leveraging modern video codec features, TASM offers a novel solution by introducing a tile-based storage manager that optimizes how videos are stored and accessed. By dividing video frames into tiles, it enables spatial random access, allowing for the retrieval and processing of only the necessary regions within frames. TASM uses a dynamic approach to adjust tile layouts according to video content and query workloads, employing a semantic index to improve layout efficiency based on detected objects. This method significantly reduces the need to decode irrelevant data and improves query performance, especially for subframe selection and object detection tasks. Through a combination of uniform and non-uniform tile layouts, incremental tiling strategies, and an initial layout based on regions of interest, TASM addresses the limitations of traditional video storage systems, enhancing both storage efficiency and query effectiveness. Their evaluations show that TASM speeds up subframe selection queries by over 50% on average and up to 94%, and doubles throughput for the full scan phase in object detection queries.

3 Integrating TASM and FiGO

We process an input query (template shown in 2) to retrieve objects of a particular type from a given video using either TASM or FiGO. If the object type in the current query has been queried previously, the query is processed through TASM, as all associated metadata is already available in the TASM semantic index. Otherwise, the query is processed using FiGO.

```
SELECT (object, frame, x1, y1, width, height) FROM video
```

Figure 2: Query template

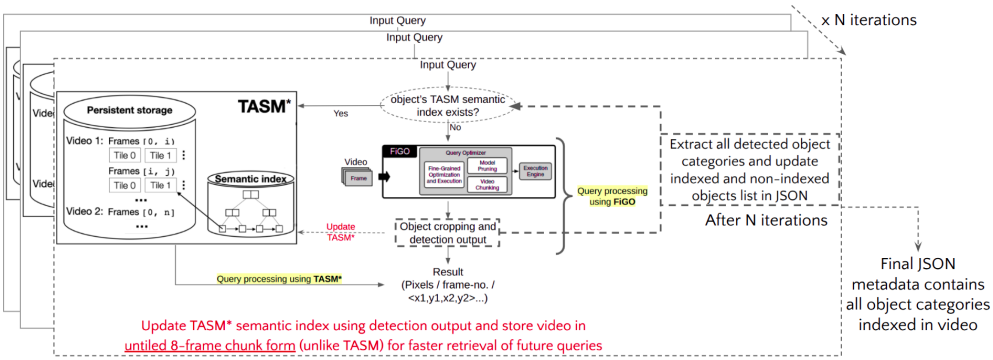


Figure 3: **Automated Retrieval of All Objects in Video.** Value of N is iteratively determined based on the number of object categories present in video

To determine whether a particular object type has already been queried for a given video, we maintain a JSON file that records a list of unique object types previously queried for each video.

We modified TASM to store videos in 8-frame chunks. The TASM semantic index is used to fetch and decode only those video chunks that contain the queried object type. Additionally, we extended the TASM semantic index to store not only object type metadata but also the temporal and spatial locations of objects (i.e., frame number, `x1`, `y1`, `width`, `height`).

Similarly, FiGO was modified to return cropped object images along with their temporal and spatial locations. When a query is processed through FiGO, the returned output is used to update the TASM semantic index so that future queries involving the same object type can be directly processed through TASM, thereby eliminating the need to re-run FiGO.

Figure 1 illustrates the integration method discussed above. Figures 4 and 5 show sample query outcomes from integrated TASM and FiGO database system.

3.1 Automated Retrieval of All Objects in Video

Figure 3 illustrates the automated retrieval method discussed above. In addition to returning results for the queried object type, FiGO's query execution also yields detections of other object types present in the video. Although FiGO optimizes its execution based on the queried object type—potentially making detections of other object types less reliable—the list of these additional detections is still used to update the set of all object types identified so far in the video. Unreliable detections or incorrectly detected object types are removed from the set when their corresponding FiGO queries are processed. This information is maintained in a JSON file.
For each video, the JSON file tracks both:

1. the set of all object types detected so far, and
2. the subset of those object types that have already been processed through FiGO.

This enables an automated and iterative query execution strategy. During each iteration, a query is issued for an unprocessed object type using FiGO. The metadata obtained from FiGO's output is then used to update the TASM semantic index, and the object type is marked as processed in the JSON file. This process continues until

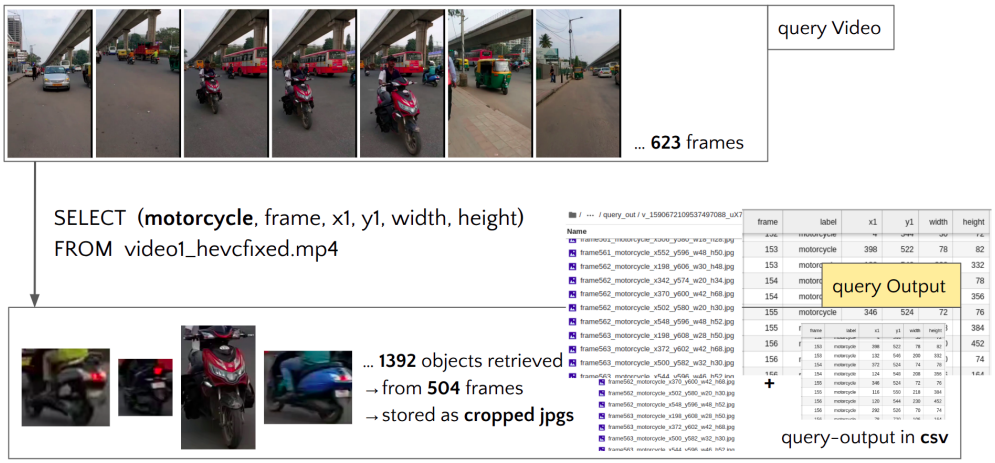


Figure 4: Sample query to retrieve all motorcycles along with their metadata from given video using TASM and FiGO integrated system.

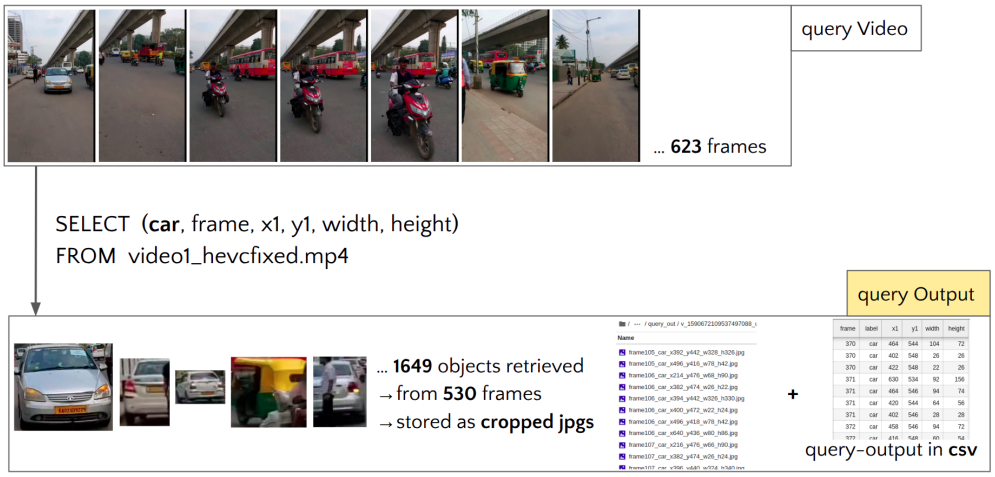


Figure 5: Sample query to retrieve all cars along with their metadata from given video using TASM and FiGO integrated system.

all detected object types for the video have been processed through FiGO once and their metadata stored in TASM.

After this initial bootstrapping, all subsequent queries for any of these object types are served using TASM alone. If a query is issued for an object type not present in the final set recorded in the JSON file, it is assumed to be absent from the entire video, and the query returns a `null` result.

This automated retrieval pipeline eliminates the need to repeatedly invoke FiGO's object detection models for queries targeting the same object types. As a result, such queries are answered instantly using the integrated TASM system, significantly reducing overall query execution time.

Please refer to the provided Project Code for full implementation details.

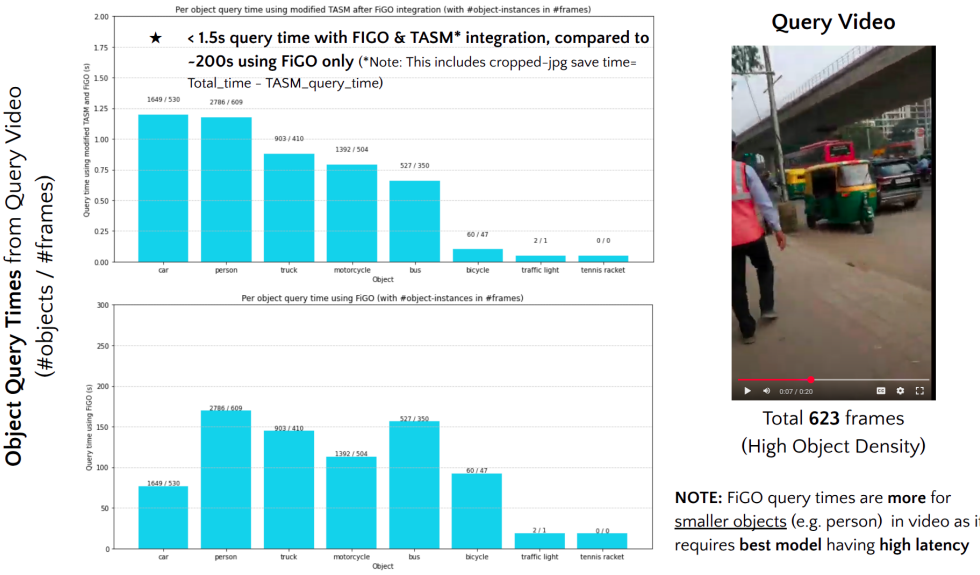


Figure 6

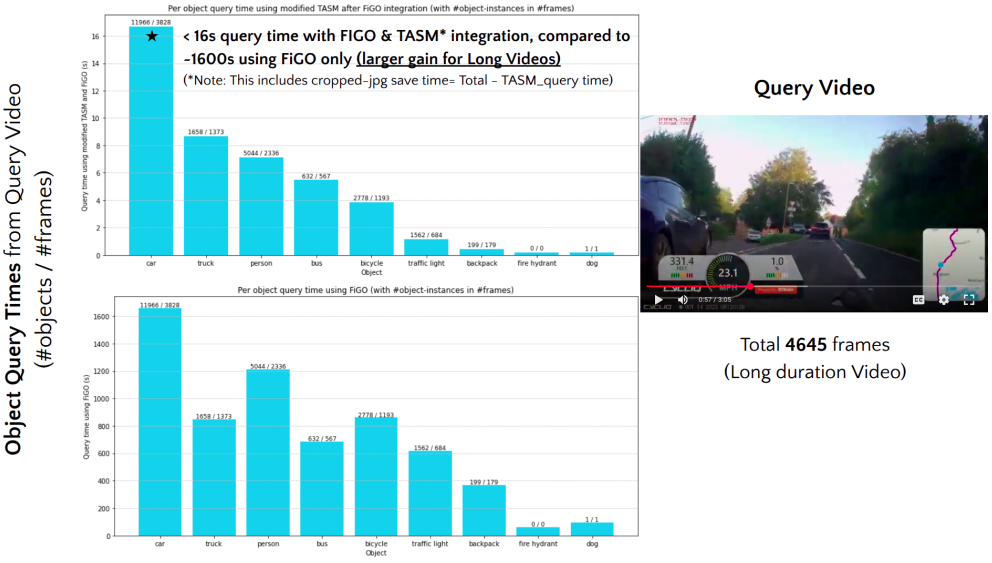


Figure 7

4 Experiments and Results

We evaluate the proposed improvements by comparing query execution times between a FiGO-only Video-DBMS and our integrated FiGO+TASM system across diverse video types, including those with dense object occurrences (Figures 6, 7, 8, 9), sparse object distributions (Figures 10, 11), and long-duration videos (Figure 7).

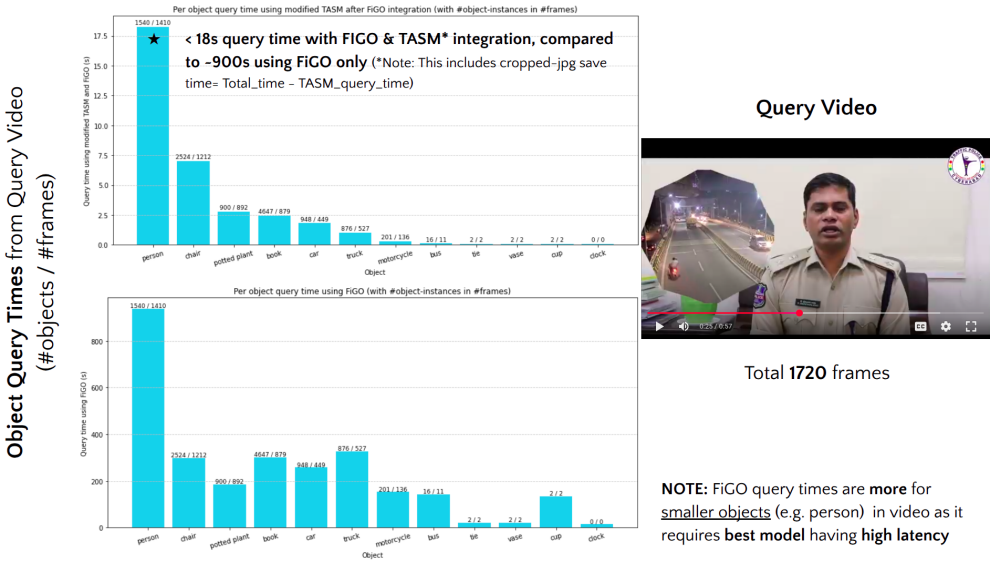


Figure 8

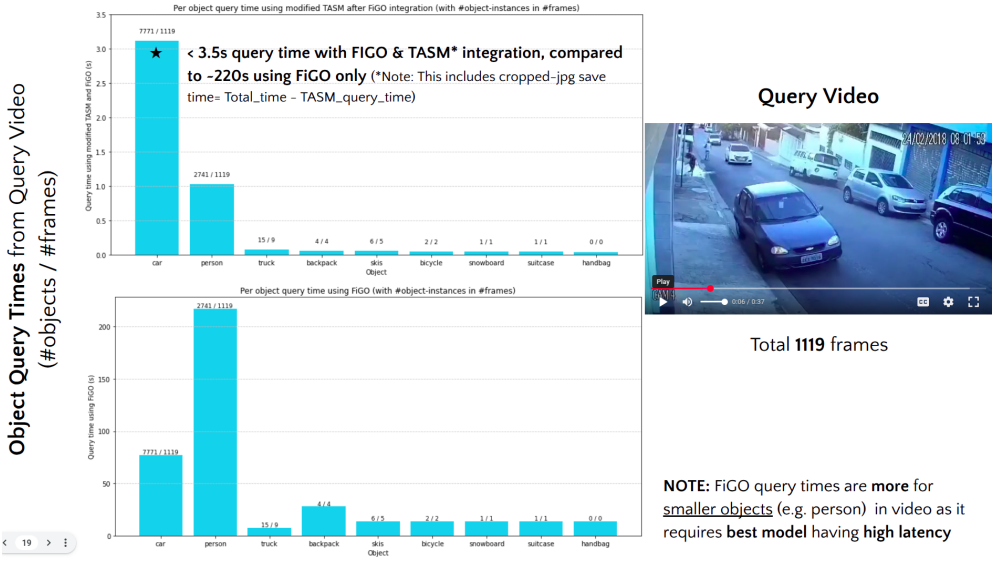


Figure 9

In each figure, the top graph represents the query execution time using the integrated FiGO+TASM system, while the bottom graph corresponds to execution time using a FiGO-only Video-DBMS. The x-axis denotes the list of all object types detected in the video. The value displayed above each bar indicates the number of objects found and the number of frames in which they appear. Columns labeled `0/0` indicate that the queried object type is not present in the corresponding video. For visualizations of these videos and their query results, please refer to the accompanying Project Video.

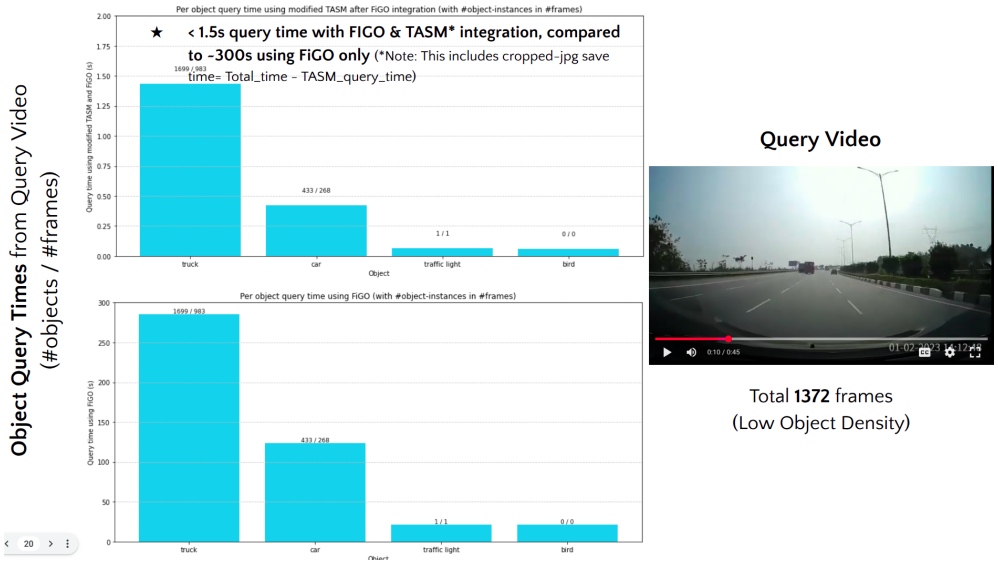


Figure 10

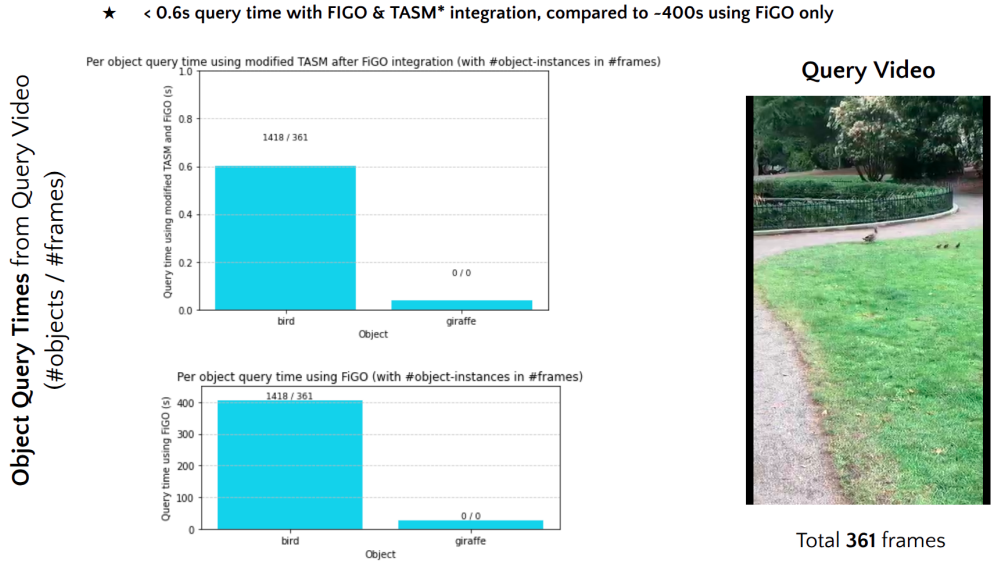
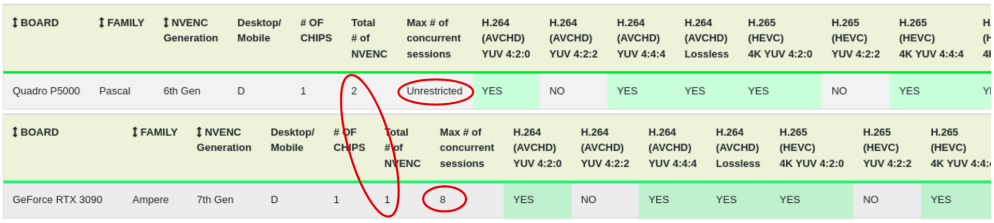


Figure 11

As evident from the graphs in each figure, significant reductions in query execution time were observed across all tested videos, with times dropping from approximately 200–1600 seconds (FiGO-only) to just 0.6–18 seconds using the integrated FiGO+TASM system. These execution times include the overhead of saving cropped object images files to disk.



BOARD	FAMILY	NVENC Generation	Desktop/ Mobile	# OF CHIPS	Total # of NVENC	Max # of concurrent sessions	H.264 (AVCHD) YUV 4:2:0	H.264 (AVCHD) YUV 4:2:2	H.264 (AVCHD) YUV 4:4:4	H.264 (AVCHD) Lossless	H.265 (HEVC) 4K YUV 4:2:0	H.265 (HEVC) YUV 4:2:2	H.265 (HEVC) 4K YUV 4:4:4	H
Quadro P5000	Pascal	6th Gen	D	1	2	Unrestricted	YES	NO	YES	YES	YES	NO	YES	Y

BOARD	FAMILY	NVENC Generation	Desktop/ Mobile	# OF CHIPS	Total # of NVENC	Max # of concurrent sessions	H.264 (AVCHD) YUV 4:2:0	H.264 (AVCHD) YUV 4:2:2	H.264 (AVCHD) YUV 4:4:4	H.264 (AVCHD) Lossless	H.265 (HEVC) 4K YUV 4:2:0	H.265 (HEVC) YUV 4:2:2	H.265 (HEVC) 4K YUV 4:4:
GeForce RTX 3090	Ampere	7th Gen	D	1	1	8	YES	NO	YES	YES	YES	NO	YES

Source: https://developer.nvidia.com/video-encode-and-decode-gpu-support-matrix-new

Figure 12: The two major differences between the GPUs Quadro P5000 and GeForce RTX 3090 is highlighted in red ellipses.

5 Challenges

One of the main challenges encountered during system development was due to the differences in GPU hardware. While the original TASM implementation was developed for NVIDIA Quadro P5000 GPUs, our experiments were conducted on RTX 20- and 30-series GPUs. The differences between the two GPUs, as shown in Figure 12, led to several unexpected issues, particularly in how video tiles were handled during storage and retrieval.

Video Storage Layout

- **Uniform and Non-Uniform Tile Layouts:** Initially, differences in tiling mechanisms across GPU architectures caused compatibility issues. These were resolved by adopting a consistent storage format, wherein videos are tiled and stored sequentially in 8-frame chunks.
- **Tiled storage layout** requires videos to have their dimensions divisible by 32 and to have HEVC codec format (TASM limitations). Any custom video that does not satisfy these criteria is first converted to the appropriate format and then processed further.

Object Retrieval from Tiled Video Storage

- While tiled storage was achieved, object retrieval from chunked *and* tiled video storage remains problematic. The retrieval process in this format significantly differs from retrieval in purely tiled video storage, leading to failures in extracting objects accurately.
- Addressing this issue would require a substantial restructuring of the TASM codebase and a fundamental rethinking of its approach to retrieval from tiled layouts.
- However, even with such restructuring, the expected performance gains in the TASM+FiGO integrated system would remain marginal. This is because the use of chunked video storage already avoids full video decoding by loading only the relevant chunks during query execution.

Final Design Choice

Due to the above considerations, the current system design opts for **untiled video storage split into 8-frame sequential chunks**, which achieves a balance between compatibility, performance, and implementation complexity.

6 Acknowledgement

We thank Maureen Daum for her support in modifying TASM to improve its compatibility across different GPU hardwares.

References

[1] Elharrouss, O., Almaadeed, N., Al-Maadeed, S.: A review of video surveillance systems. Journal of Visual Communication and Image Representation **77**, 103116 (2021) https://doi.org/10.1016/j.jvcir.2021.103116

[2] Cao, J., Sarkar, K., Hadidi, R., Arulraj, J., Kim, H.: Figo: Fine-grained query optimization in video analytics. In: Proceedings of the 2022 International Conference on Management of Data. SIGMOD '22, pp. 559–572. Association for Computing Machinery, New York, NY, USA (2022). https://doi.org/10.1145/3514221.3517857 . https://doi.org/10.1145/3514221.3517857

[3] Daum, M., Haynes, B., He, D., Mazumdar, A., Balazinska, M.: TASM: A Tile-Based Storage Manager for Video Analytics . In: 2021 IEEE 37th International Conference on Data Engineering (ICDE), pp. 1775–1786. IEEE Computer Society, Los Alamitos, CA, USA (2021). https://doi.org/10.1109/ICDE51399.2021.00156 . https://doi.ieeecomputersociety.org/10.1109/ICDE51399.2021.00156