Efficient Reuse of Metric Indexes for Multi-resolution Queries

Rodrigo César Arboleda[1,2], **Agma Traina**[2], **Caetano Traina Jr.**[2]

[1]Google Brazil
Belo Horizonte – BH – Brazil

[2]Instituto de Ciências Matemáticas e de Computação (ICMC)
Universidade de São Paulo (USP)
São Carlos – SP – Brazil

`rodrigo.arboleda@usp.br, agma@icmc.usp.br, caetano@icmc.usp.br`

Abstract. *Performing similarity queries using resolutions different from those used for the data employed during the index construction is a challenge for traditional Metric Access Methods (MAMs). Varying data resolution occurs, for example, in scenarios involving reduced data fidelity, such as when reducing data resolution for efficient network transmission. This work proposes a novel way to extend a MAM, using the Slim-tree as a case study, to efficiently and adaptively support queries over transformed data using indices built using only the original, full-resolution data. The proposed approach modifies the pruning heuristic to incorporate safe upper bounds on distances in the transformed domain, guaranteeing correct query results while preserving much of the pruning power of the original MAM. Experimental results on image datasets demonstrate substantial performance improvements over sequential search, even under conditions of high compression.*

1. Introduction

The growing need to analyze large volumes of complex data, such as images, videos, and audio, often distributed across remote sites via digital networks, poses significant challenges to traditional similarity query retrieval methods. In many practical scenarios, such as efficient data transmission in bandwidth-constrained environments or on devices with limited capacity, it is common to apply transformations that reduce the resolution of this data. However, this practice introduces a critical challenge: Metric Access Methods (MAMs) built on the original high-resolution data may become incompatible with their transformed counterparts. MAMs are specialized index structures that segment a metric space into hierarchical regions to organize complex objects. The spatial segmentation takes into account properties of the data space to accelerate similarity query execution, to allow pruning entire segments that certainly do not contain query answers.

To illustrate, consider a simple case in which an image search procedure filters result candidates using color histograms with a fixed dimensionality, like 256 bins. In scenarios with high network traffic, it may be advantageous to reduce the histogram resolution to decrease data transfer. However, index structures built for high-resolution histograms are incompatible with queries posed at lower resolutions, and vice versa: the resolution of the queries must match that of the index. Similar issues arise during exploratory data analysis, where various data transformations are tested to identify the most suitable representation, again requiring resolution alignment between queries and the index structure.

In environments with limited network connectivity, such as edge or border computing, mobile applications must operate within strict bandwidth constraints. For example, in a visual similarity search task, a client may send low-resolution image descriptors to a server for product identification. A key challenge is processing these low-fidelity queries efficiently against a high-resolution data index to maintain low-latency responses. Our proposed solution directly leverages the high-resolution index to answer low-resolution queries, bypassing the need for costly index reconstruction or exhaustive sequential scans. This approach significantly mitigates performance degradation caused by limited network connectivity, ensuring faster and more efficient query handling.

As a result, current data search approaches often fall back on either costly sequential scans or complete index reconstruction for the transformed data, both of which significantly degrade search efficiency. This work directly addresses this challenge by introducing an adaptive solution that enables the efficient reuse of existing indices, even when queries target data that has been transformed, for instance, through resolution reduction via the Haar Transform. We propose an adaptation to the pruning heuristics of MAMs, exemplified using the Slim-tree [Traina Jr et al. 2000], which integrates safe upper bounds for distances in the transformed domain. This ensures result correctness while, depending on how much the original data is compressed, it is able to preserve much of the original structure's pruning ability. By bridging the gap between the need for data transformation and the efficiency of similarity queries, our approach offers a promising direction for faster and more cost-effective analysis in distributed and resource-constrained environments.

2. Background

2.1. Similarity Measurements

Similarity measurements quantify how closely objects resemble each other in similarity queries and algorithms operating in distance spaces [Zezula et al. 2006, Gupta et al. 2025]. They are crucial in Database Management Systems (DBMS), especially for managing complex datasets like image collections. Similarity can be quantified using various methods, such as similarity coefficients or distance functions [Zaki and Meira 2020]. The latter are particularly significant. They evaluate the similarity of a pair of objects, where smaller values indicate higher similarity, with zero representing identical objects. Distance functions that satisfy the properties of non-negativity, identity of indiscernibles, symmetry, and the triangle inequality are referred to as metrics [Chen et al. 2023, Deza and Deza 2016]. A typical example is the Manhattan distance, which is often used to compare color histograms in image analysis tasks [Vadivel et al. 2003].

2.2. Complex Data Signatures

A Complex Data Signature is a set of features extracted from a complex object, typically represented as a feature vector. A signature captures distinct characteristics that help identify and compare objects, enabling a more compact representation and faster and more accurate comparisons of objects. They are widely used in database systems for indexing and comparing complex data such as images, videos, and audio [Saouabe et al. 2024, Cazzolato et al. 2019]. A data signature is typically derived through mathematical or statistical techniques designed to capture the essential characteristics of the original object. A variety of methods can be employed for this purpose, including image processing algorithms [Marques and Rangayyan 2013] and techniques based on machine learning models [Elharrouss et al. 2024, Santana and Ribeiro 2023].

2.3. Metric Access Methods

A Metric Access Method (MAM) is an indexing technique used in databases that work with complex data, where a metric defines the notion of similarity. They are important for optimizing similarity queries in applications such as image and video retrieval [Sharma et al. 2019]. A MAM segments the data space into regions, enabling selective data pruning without analysis of individual elements. For instance, an R-tree [Guttman 1984] organizes dimensional data into a balanced tree structure, where each node represents a specific region in the data space. By calculating the minimum and maximum distances between two nodes, it becomes possible to define node intersections, enabling their use as a MAM. The primary goal when building a MAM is to minimize the number of nodes traversed to reach a leaf node. MAMs employ heuristics that, together with the distances computed within the structure, enable discarding regions in the search space (pruning), making the process significantly more efficient compared to the sequential search [Chen et al. 2023].

Many MAMs have been proposed to improve query performance in databases storing complex data. It is possible to use structures developed for multi-dimensional data associated to a metric, as well as specific structures were designed for “purely metric” data (such as the VP-tree [Yianilos 1993]), that only require the metric and the data (data do not need to be dimensional), such as texts and sequences. When a structure operates on high-dimensional objects, its performance degrades as dimensionality increases, eventually becoming worse than a sequential scan. This decline is due to frequent overlaps between node regions, requiring the evaluation of every node that overlaps during searches. As the dimensionality increases, the overlaps tend to increase, also increasing the number of nodes that must be visited, which is known as the "curse of dimensionality," which worsens in higher dimensionality. Several structures have been developed to address this challenge, such as the X-tree [Berchtold et al. 1996] and Slim-tree [Traina Jr et al. 2000].

The Slim-tree is a purely metric MAM designed to organize data based on distances in a metric space. It selects a small number of elements to serve as pivots, where each pivot, together with a radius, defines a node’s covering region. Nodes are typically stored in fixed-length blocks in secondary memory. A key advantage of the Slim-tree lies in its ability to optimize node occupancy while minimizing overlap between regions. A node splits when its storage capacity is exceeded, constructing a minimum spanning tree (MST) from the stored elements, using pairwise distances as edge weights. The longest edge in the MST is then removed, partitioning the data into two groups.

Both R-trees and Slim-trees support the dynamic insertion of new elements without requiring a complete rebuild of the structure, making them ideal for databases that experience frequent updates. In contrast, structures like the VP-tree lack this flexibility, as they require more effort to accommodate new elements, limiting their adaptability in dynamic environments.

MAM structures rely on a specific distance measurement for data comparison. If it changes, the entire structure must be rebuilt, a process that can be highly time-consuming, particularly when dealing with large datasets. In the same way, if transformations are applied to the data already stored, the MAM also needs to be reconstructed.

2.4. The Wavelet Transforms and the Haar Transform

Data transformations are widely employed in signal processing. Wavelet transforms are among the most useful ones. They enable multiresolution analysis by decomposing a signal into components at various scales [Abdel-Hamid 2024, Daubechies 1990]. They allow a detailed evaluation of both coarse and fine features, making it possible to analyze signals at multiple levels of resolution and capture localized variations at varying levels of detail.

The Haar transform is particularly effective for signal compression, and is frequently applied in image processing [Eliasof et al. 2024, Mulcahy 1997]. It operates by computing the averages and differences between pairs of elements, as follows. Given a vector $[a_1, a_2, a_3, ..., a_n]$, two arrays are calculated: the coefficients of approximation $A_h^{(a)}$ and of details $A_h^{(d)}$:

$$A_h^{(a)} = [\frac{a_1 + a_2}{2}, \frac{a_3 + a_4}{2}, ..., \frac{a_{n-1} + a_n}{2}] \quad (1)$$

$$A_h^{(d)} = [\frac{a_1 - a_2}{2}, \frac{a_3 - a_4}{2}, ..., \frac{a_{n-1} - a_n}{2}] \quad (2)$$

In the context of content-based image retrieval (CBIR – retrieval of images based on their visual content), the approximation coefficients capture the coarse structure of the image. They can be compared more efficiently, making them well-suited for initial retrieval steps. This transformation can be applied iteratively to the approximation coefficients, yielding progressively coarser representations. This multilevel decomposition is especially valuable for narrowing down the search space, enabling a fast, preliminary selection of candidate images, which can then undergo more detailed and computationally intensive comparisons using higher-resolution features.

The Haar transform can be extended to multidimensional data by applying it sequentially along each dimension, typically starting with rows and then progressing to columns. With a linear complexity of $\mathcal{O}(n)$, it is highly efficient, even for large datasets. By preserving both approximation and detail coefficients, the transform remains reversible and lossless, ensuring exact data reconstruction, an essential feature for applications like Content-Based Image Retrieval (CBIR), where both efficiency and fidelity are critical.

When used for compression, the Haar transform retains only the approximation coefficients. The compression ratio is adjustable via the number of decomposition levels, enabling a flexible trade-off between efficiency and fidelity. This adaptability makes it especially effective in contexts where storage or transmission constraints must be balanced while preserving the essential image content.

3. Methods

3.1. Description of the adopted approach

Structures designed for complex data, such as the Slim-tree and X-tree, are effective for similarity queries. However, they become unsuitable when queries involve transformed data – for instance, when accessing different resolutions or applying data transformations. In such cases, either the MAM must be entirely rebuilt using the transformed data, or the system must fall back on sequential search. This limitation arises because the precomputed distances and the resulting search space no longer align with the modified data, preventing the structure from effectively guiding the search.

Consider, for instance, a set of complex objects represented by image feature vectors, such as 256-bin color histograms. If a structure like the Slim-Tree is built on these vectors, it can support efficient similarity queries searching for 256-dimensional vectors. However, when a query requires comparisons using a resolution of 128 bins, the original structure is useless, because the previously computed distances are no longer the same for the new representation. The workaround is to perform a sequential search, because the transformation can be applied to each object immediately before performing the similarity computation. While this ensures correctness, it sacrifices the efficiency provided by the original indexing structure.

A potential transformation suitable for this scenario is the Haar Transform, whose reversibility can be leveraged to maintain efficiency and flexibility. By employing only the approximation coefficients, similarity comparisons can be performed more quickly on a lower-dimensional representation. At the same time, by storing the coefficients of details, it remains possible to reconstruct the original data when needed. In the case of image histograms, for example, the Haar Transform enables, in a single step, a reduction in dimensionality from 256 bins to 128 bins.

Indexing structures rely on heuristics to optimize computations and enable subspace pruning during query processing, using previously calculated distances to avoid unnecessary comparisons. However, when a transformation is applied to the data, these precomputed distances become invalid, rendering the indexing structure ineffective. This work proposes a method that allows continued use of the index even after the data has been transformed. When estimating a bound on how much the new distances deviate from the original ones is possible, the indexing heuristics can be adapted accordingly. By incorporating these estimated distances into the pruning mechanism, the index can still guide the search process effectively. As a result, our approach can deliver performance superior to that of a full sequential search, retaining part of the efficiency of the original index without requiring its reconstruction.

3.2. Implementation Details

This work investigates an indexing framework for querying data subjected to resolution-reducing transformations. We combine the Slim-tree with the Haar transform and use Manhattan distance as a representative similarity measure, focusing on range queries as the primary query type.

When determining whether a search operation should descend into a child, the Slim-tree (like most hierarchical MAMs) considers three objects: the query center O_c, the representative of the current node O_n, and the representative of the candidate child O_v. Additionally, four relevant distance values are involved: the search radius R_b, the distance between the query center and the current node representative D_{cn}, the distance between the current node and the child node representatives D_{nv}, and the covering radius of the child node R_v, as shown in Figure 1. Notice that not all of these distances require approximation: the search radius R_b is provided directly by the user and is exact, and D_{cn} is computed before entering the current node, so it is precisely known.

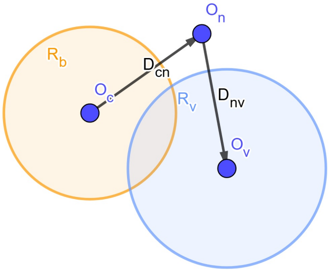

Figure 1. Representation of objects and distances in the query

However, D_{nv} and R_v are precomputed and stored within the MAM structure. After a transformation is applied to the data, these values are no longer valid in the transformed space. As a result, it becomes necessary to employ a heuristic to ensure that the pruning process remains safe; that is, it must guarantee that no potentially relevant data is mistakenly excluded. A heuristic that satisfies this condition can be formulated as follows:

$$D_{cn} - D_{nv} \leq R_b + R_v \tag{3}$$

Therefore, if R_v is overestimated, the pruning condition remains valid in all correct cases, but potentially admits additional false positives that increase processing. The same rationale applies to D_{nv}. Thus, by computing guaranteed over-approximations of R_v and D_{nv}, we ensure that no relevant results are missed, thereby preserving query integrity of the query response while safely operating in the transformed data space.

Under those conditions, the less accurate the approximation, the more false positives will be introduced, and thus the higher the likelihood of failing to prune nodes that could otherwise be safely discarded. Therefore, the quality of the approximation is critical to the efficiency of this approach.

To find a good approximation to the Haar transform used with the Manhattan distance, consider two vectors of dimensionality e:

$$A = [a_1, a_2, a_3, a_4, \ \ldots \ a_{e-1}, a_e] \tag{4}$$

$$B = [b_1, b_2, b_3, b_4, \ \ldots \ b_{e-1}, b_e] \tag{5}$$

The Manhattan distance (D_{manh}) between A and B is given by Equation 6:

$$D_{manh}(A, B) = |a_1 - b_1| + |a_2 - b_2| + \ldots + |a_e - b_e| \tag{6}$$

After applying the Haar transform, two new vectors A_h and B_h are obtained as the transformed versions of the original vectors A and B, concatenating both approximation (a) and detail (d) coefficients.

$$\begin{aligned} A_h = [A_h^{(a)}; A_h^{(d)}] = [&\frac{a_1 + a_2}{2}, \frac{a_3 + a_4}{2}, \ldots, \frac{a_{e-1} + a_e}{2}; \\ &\frac{a_1 - a_2}{2}, \frac{a_3 - a_4}{2}, \ldots, \frac{a_{e-1} - a_e}{2}] \end{aligned} \tag{7}$$

$$\tag{8}$$

$$\begin{aligned} B_h = [B_h^{(a)}; B_h^{(d)}] = [&\frac{b_1+b_2}{2}, \frac{b_3+b_4}{2}, ..., \frac{b_{e-1}+b_e}{2}; \\ &\frac{b_1-b_2}{2}, \frac{b_3-b_4}{2}, ..., \frac{b_{e-1}-b_e}{2}] \end{aligned} \tag{9}$$

For compression using the Haar transform, only the approximation coefficients are used. Thus, considering only the approximation coefficients from $A_h^{(a)}$ and $B_h^{(a)}$, we can compute a safe upper bound for the Manhattan distance between A and B.

$$\begin{aligned} D_{\text{manh}}(A_h^{(a)}, B_h^{(a)}) &= \left|(\frac{a_1+a_2}{2} - \frac{b_1+b_2}{2})\right| + \left|(\frac{a_3+a_4}{2} - \frac{b_3+b_4}{2})\right| \\ &+ \cdots + \left|(\frac{a_{e-1}+a_e}{2} - \frac{b_{e-1}+b_e}{2})\right| \\ &= \frac{1}{2}\big(\, |a_1 - b_1 + a_2 - b_2| \quad + \cdots + |a_{e-1} - b_{e-1} + a_e - b_e|\,\big) \end{aligned} \tag{10}$$

For any real numbers x and y, the triangle inequality states:

$$|x+y| \leq |x| + |y| \tag{11}$$

Applying this inequality to each term of 10. For each pair of indices $(2i-1, 2i)$, let $x = a_{2i-1} - b_{2i-1}$ and $y = a_{2i} - b_{2i}$. Applying the inequality gives:

$$\big|(a_{2i-1} - b_{2i-1}) + (a_{2i} - b_{2i})\big| \leq |a_{2i-1} - b_{2i-1}| + |a_{2i} - b_{2i}| \tag{12}$$

Summing all the pairs from $i = 1$ to $e/2$, we have:

$$\begin{aligned} &|a_1 - b_1 + a_2 - b_2| + ... + |a_{e-1} - b_{e-1} + a_e - b_e| \leq \\ &\leq |a_1 - b_1| + |a_2 - b_2| + ... + |a_{e-1} - b_{e-1}| + |a_e - b_e| \end{aligned} \tag{13}$$

This implies that:

$$\begin{aligned} &2 \times D_{manh}(A_h, B_h) \leq D_{manh}(A, B) \Leftrightarrow \\ &\Leftrightarrow D_{manh}(A_h, B_h) \leq \frac{D_{manh}(A, B)}{2} \end{aligned} \tag{14}$$

Since $D_{manh}(A_h, B_h) \geq 0$, we have shown that the distance in the transformed domain – considering only the Haar transform's approximation coefficients – is upper-bounded by half of the original distance. This property allows us to safely estimate upper bounds in the transformed space using the original distances stored in the index, enabling an adaptation of the Slim-tree's pruning heuristic. As a result, we ensure the heuristic remains safe, because no relevant data is mistakenly discarded during pruning. This approach provides a practical method to reuse an existing Slim-tree structure for multi-resolution similarity queries, eliminating the need for index rebuilding.

4. Experiments

4.1. Experimental Setup

We experimentally validated our proposed multiresolution indexing strategy by performing range similarity queries using the Manhattan distance (L_1) across multiple image datasets. In this Section, we show the results obtained only for 256-bin grayscale image histogram vectors derived from the complete Stanford Dogs Dataset [Khosla et al. 2011, Cazzolato et al. 2022] (20,580 images), as results from the other datasets are similar. Each object is composed of a name (a string attribute with a maximum of 200 bytes) and its full-resolution histogram (256 64-bit floats, 2048 bytes). Query objects were represented at various resolutions, generated by applying the Haar Wavelet Transform to reduce dimensionality before each search execution.

The experiments take the dataset's sequential scan as the baseline compared to our proposed approach. The latter utilizes a single index structure (an adapted Slim-tree) built using the original, full-resolution data (256 bins). For queries posed at lower resolutions, the same index is reused, employing safe pruning logic based on estimated distance upper bounds derived from the multiresolution properties of the Haar transform. Performance was evaluated by averaging results over 500 queries for each experimental configuration, with each query centered on a randomly selected object from the dataset. Key performance indicators measured are: average query time, average disk page accesses, average number of distance calculations, and average number of objects returned satisfying the query radius.

A preliminary analysis was conducted to optimize disk page size, evaluating pages sizes of 32, 64, 128, and 256 KB, measuring the number of disk accessed to answer queries at full resolution (256 bins) and at reduced resolution (128 bins, 1 Haar application) over all pages, using a radii selected to return approximately 15 objects. As shown in Figure 2, increasing page size yields diminishing returns after 128 KB. Thus, a 128 KB page size was adopted for all subsequent experiments to optimize I/O efficiency.

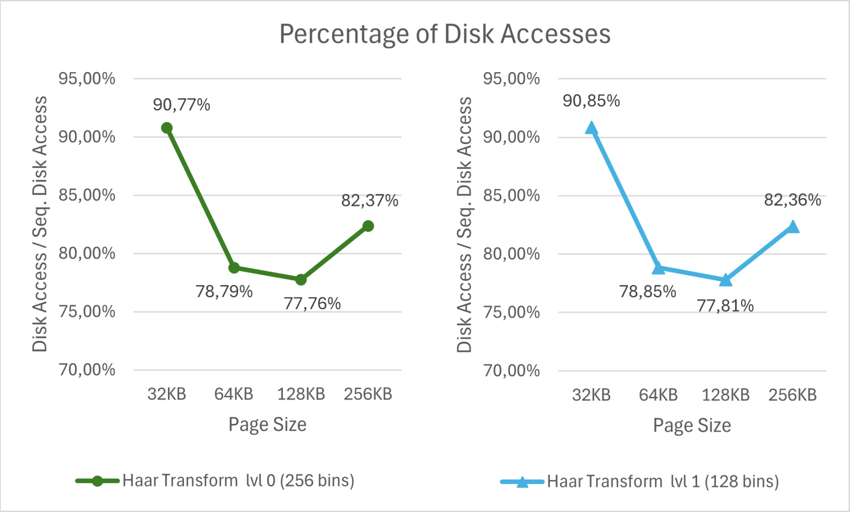

Figure 2. Percentage of disk accesses

4.2. Performance Evaluation

We evaluated the core performance aspects of the proposed index against the sequential scan baseline across different query radii and data resolutions.

Disk Accesses: A critical factor in database performance is the I/O cost. Our indexed approach substantially reduces the average number of disk pages accessed per query compared to a full sequential scan, which must read the entire dataset. As illustrated in Figure 3, the reduction exceeds 34% in favorable scenarios. The overhead compared to a hypothetical Slim-tree used only at its native resolution is minimal, demonstrating the efficiency of the multiresolution adaptation for I/O.

Distance Calculations: This metric counts the number of times a full distance comparison is performed between the query object and either a data object or an index node's representation. This metric counts the number of comparisons but does not weigh

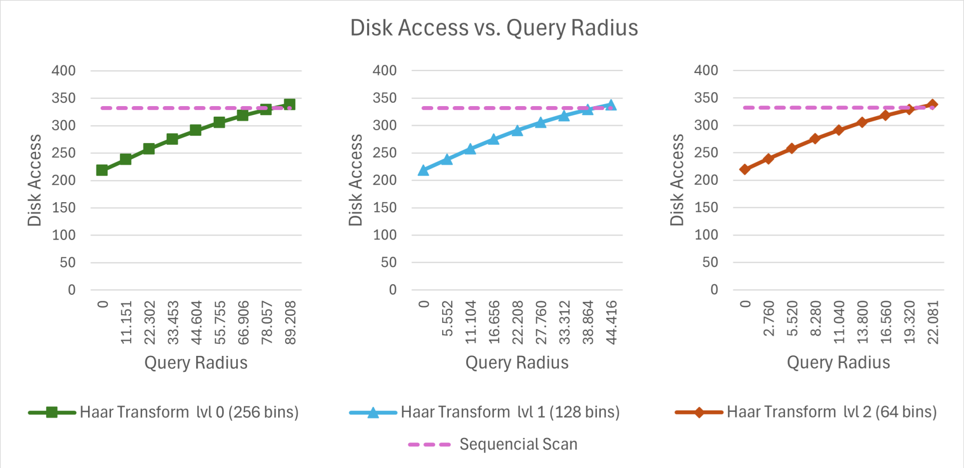

Figure 3. Average disk accesses vs. query radius at different Haar transform levels.

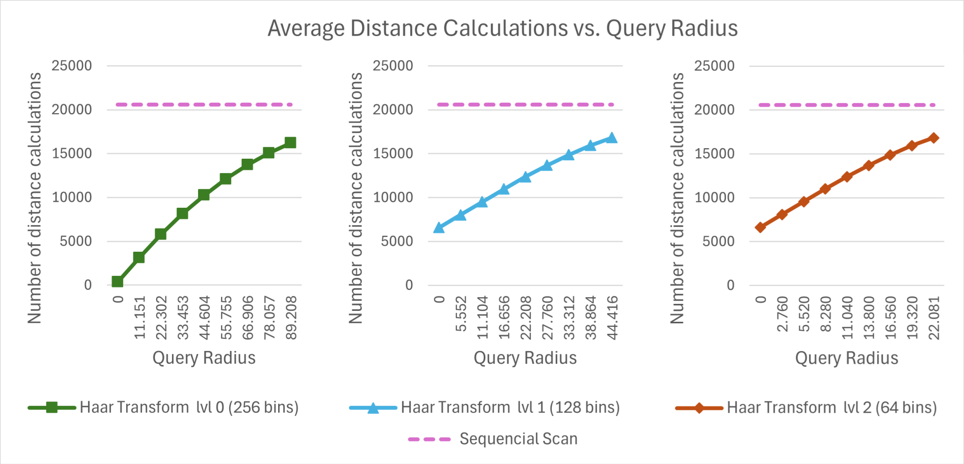

Figure 4. Average distance calculations vs. query radius at different Haar transform levels.

them by their computational cost; the cost of applying the necessary Haar transform steps to match the query resolution before the L_1 calculation increases with the compression level. The impact of this varying computational cost is captured with the Total Execution Time analysis, as follows. Figure 4 shows that the index's pruning mechanism significantly reduces the number of these calculations compared to the sequential scan (over 68% reduction observed), which must compare the query against every object. While the count is higher than an ideal single-resolution index would achieve, the gain over the baseline clearly shows the effectiveness of the pruning strategy.

Total Execution Time: This metric offers a comprehensive view of performance, accounting for both I/O costs (e.g., disk access) and CPU costs (e.g., distance computations, Haar transformations, and index traversal). Although execution time can be affected by system-level factors, which introduce some variability, Figure 5 clearly shows substantial average time savings achieved by the proposed approach compared to sequential search. These gains become even more significant at higher compression levels (i.e., more Haar transform steps). This is because sequential search must apply the computationally expensive Haar transform to every object in the dataset to match the query resolution, leading to high overhead. In contrast, our indexed method performs transformations selectively – only for relevant nodes and candidate objects identified through pruning – resulting in significantly faster query execution. The proposed method consistently achieves around a 3 times speedup for queries retrieving up to 3% of the data when using compression rates of 2 and 4 times.

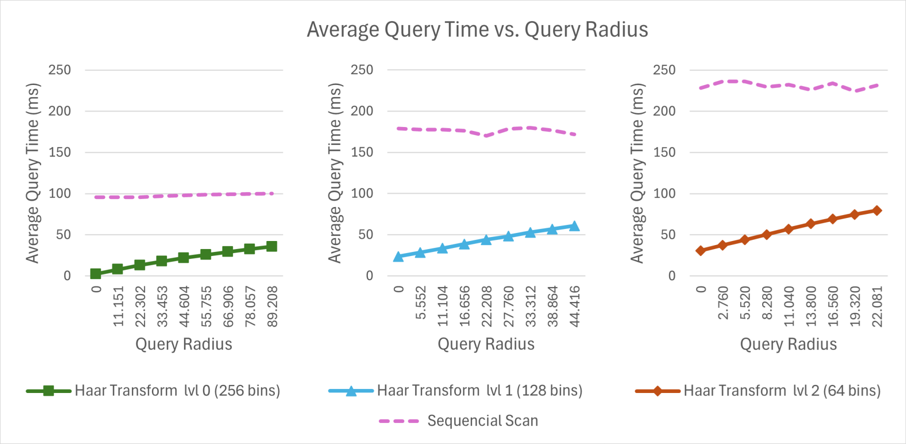

Figure 5. Average query time vs. query radius at different Haar transform levels.

4.3. Performance at High Compression Levels

A key aspect of our proposal is the ability to reuse an index built on full-resolution data to support queries at potentially much lower resolutions. To assess the robustness of the index under such stress conditions, we evaluated its performance across multiple levels of Haar transform application, ranging from 0 to 7 steps, corresponding to dimensionality reduction factors from 1 to $2^7 = 128$.

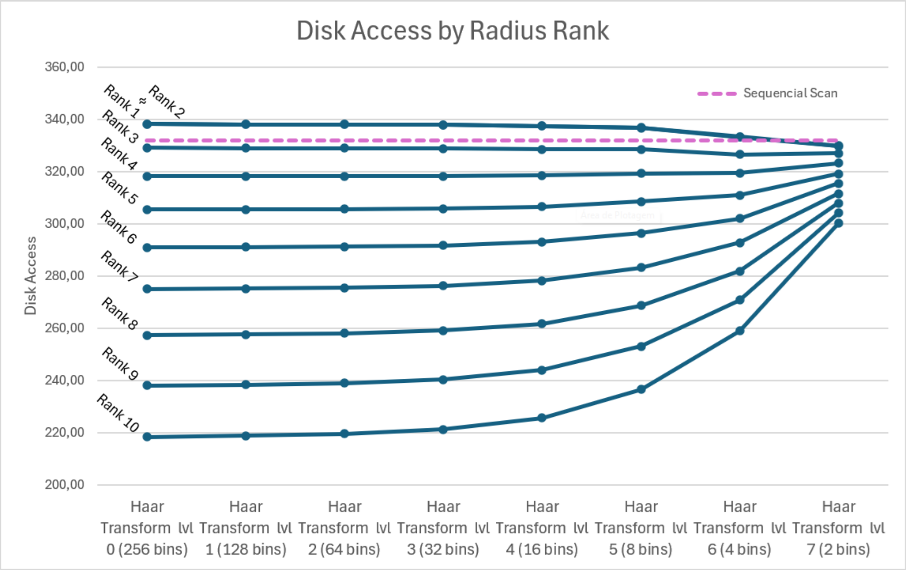

Figure 6. Disk accesses vs. radius rank across increasing Haar transform levels.

To enable fair comparisons across different resolutions – where absolute distance values can vary significantly – we used query radii normalized by selectivity. Instead of fixed radii, we defined 10 rank levels, each representing a relative query selectivity. For each compression level c, we first determine a reference radius r_1^c that, on average, retrieves 10% of the dataset; this corresponds to Rank 1. Then, for each compression level c, we define 10 radii r_k^c, where k is the Rank from 1 to 10, varying linearly between radius 0 (corresponding to Rank 10) and radius r_1^c. For example, if $r_1^c = 90$ for a given compression level c, then the radii for Ranks 1 through 10 would be $r_1^c = 90, r_2^c = 80$ and so on.

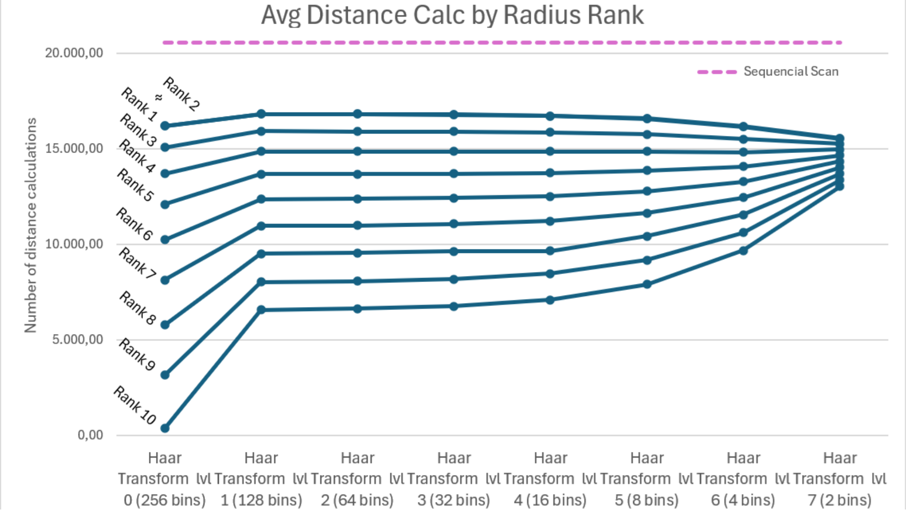

Figure 7. Distance calculations vs. radius rank across increasing Haar transform levels.

Figures 6 (disk accesses) and 7 (distance calculations) show that index efficiency, in terms of these metrics, begins to degrade more noticeably after approximately four Haar transform steps (i.e., 16× dimensionality reduction). However, the total execution time analysis in Figure 8 provides a more comprehensive view. As compression increases, each distance computation becomes costlier, as additional Haar transformation steps are required. Sequential search must perform these costly computations for all 20,580 objects, leading to a significant runtime increase. In contrast, while the indexed approach also incurs some increase in distance calculations and node accesses, its pruning capability allows avoiding most of the unnecessary computations. As a result, our method achieves significant speedups – often exceeding 50% – even at high compression levels and for queries retrieving a substantial portion of the dataset. This confirms the practicality and effectiveness of reusing the index across wide variations in data resolution.

The implemented software files are available at GitHub: https://github.com/rodrigo-arboleda/reuse_of_metric_indexes.

5. Conclusion

This work targets the challenge of performing efficient similarity queries on complex data after they have undergone resolution transformations, a common scenario in distributed systems and resource-constrained applications. The incompatibility of similarity measurements between indices built over the original data and their transformed versions represents a significant obstacle, often forcing inefficient sequential scans. We demonstrated that it is possible to overcome this limitation by reusing the original indexing structure

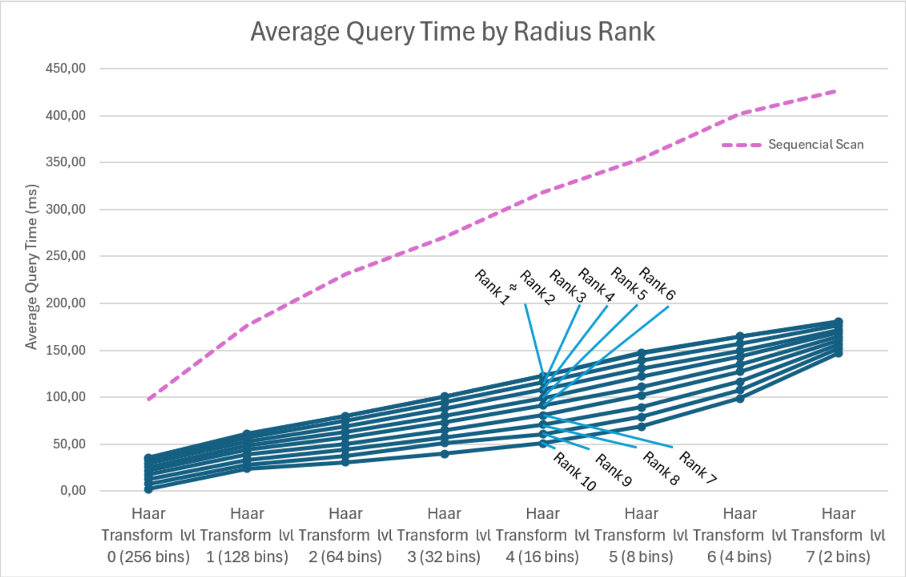

Figure 8. Average query time vs. radius rank across increasing Haar transform levels.

(exemplified using a Slim-tree in this case study) by adapting its pruning heuristic. By employing safe upper bounds for distances in the transformed domain (using the Haar Transform and Manhattan distance in this study), our approach ensures the correctness of query results, avoiding index reconstruction or performing sequential searches. Experimental results confirmed the effectiveness of the proposed solution, revealing substantial performance gains over the sequential scan method in terms of disk accesses, distance calculations, and total execution time. The proposed approach remains efficient even at high compression levels, validating the robustness of adaptively reusing metric indices for multi-resolution queries. We conclude that this strategy offers a practical and reliable means to balance resource savings in data transmission and storage with the need for efficient similarity search. Significantly, the proposed method is general and flexible. Its core idea is not tied to the specific use of the Haar transform or the Manhattan distance, but rather to the principle that adaptive index reuse is feasible whenever a safe mathematical bound can be established between distances in the transformed and original spaces. This bound is the only requirement for adapting the pruning heuristic, ensuring both correctness and efficiency of the search. Future work will explore generalizing this strategy to other transformation techniques and indexing structures, as well as investigating the use of deep learning models to learn mappings between transformed and original distances, extending the approach beyond the Haar transform.

Acknowledgments

This work was supported by the São Paulo Research Foundation (Fundação de Amparo à Pesquisa do Estado de São Paulo – FAPESP) Grants 2023/18026-8 and 2024/13328-9, the National Council for Scientific and Technological Development (CNPq), and the Coordination for the Improvement of Higher Education Personnel (CAPES).

References

Abdel-Hamid, L. (2024). Multiresolution analysis for covid-19 diagnosis from chest ct images: wavelet vs. contourlet transforms. *Multimedia Tools and Applications*, 83(1):2749–2771.

Berchtold, S., Keim, D. A., and Kriegel, H.-P. (1996). The x-tree: An index structure for high-dimensional data. In *Very large data-bases*, pages 28–39.

Cazzolato, M. T., Rodrigues, L. S., Ribeiro, M. X., Gutierrez, M. A., Traina Jr, C., and Traina, A. J. M. (2022). Sketch+ for visual and correlation-based exploratory data analysis: A case study with COVID-19 databases. *JIDM – Journal of Information and Data Management*, 13(2):308.

Cazzolato, M. T., Rodrigues, L. S., Scabora, L. d. C., Zabot, G. F., Vasconcelos, G. Q., Chino, D. Y. T., Jorge, A. E. S., Cordeiro, R. L. F., Traina Jr, C., and Traina, A. J. M. (2019). A DBMS-based framework for content-based retrieval and analysis of skin ulcer images in medical practice. In *Proceedings of SBBD 2019*, pages 109–120, Fortaleza, CE, Brazil. SBC.

Chen, L., Gao, Y., Song, X., Li, Z., Zhu, Y., Miao, X., and Jensen, C. S. (2023). Indexing metric spaces for exact similarity search. *ACM Comput. Surv.*, 55(6):39.

Daubechies, I. (1990). The wavelet transform, time-frequency localization and signal analysis. *IEEE Transactions on Information Theory*, 36(5):961–1005.

Deza, M. M. and Deza, E. (2016). *Encyclopedia of Distances*. Springer, Heidelberg, 4th edition.

Elharrouss, O., Akbari, Y., Almadeed, N., and Al-Maadeed, S. (2024). Backbones-review: Feature extractor networks for deep learning and deep reinforcement learning approaches in computer vision. *Computer Science Review*, 53:22.

Eliasof, M., Bodner, B. J., and Treister, E. (2024). Haar wavelet feature compression for quantized graph convolutional networks. *IEEE Transactions on Neural Networks and Learning Systems*, 35(4):4542–4553.

Gupta, S., Thakar, U., and Tokekar, S. (2025). A comprehensive survey on techniques for numerical similarity measurement. *Expert Systems with Applications*, 277:127235.

Guttman, A. (1984). R-tree : A dynamic index structure for spatial searching. In *ACM SIGMOD International Conference on Management of Data*, pages 47–57, Boston, MA. ACM PRess.

Khosla, A., Jayadevaprakash, N., Yao, B., and Fei-Fei, L. (2011). Novel dataset for fine-grained image categorization. In *IEEE Conference on Computer Vision and Pattern Recognition First Workshop on Fine-Grained Visual Categorization*, Colorado Springs, CO.

Marques, P. M. d. A. and Rangayyan, R. M. (2013). *Content-based Retrieval of Medical Images: Landmarking, Indexing, and Relevance Feedback*. Synthesis Lectures on Biomedical Engineering. Morgan & Claypool Publishers, San Rafael, California, USA.

Mulcahy, C. (1997). Image compression using the haar wavelet transform. *Spelman Science and Mathematics Journal*, 1(1):22–31.

Santana, D. R. and Ribeiro, L. A. (2023). Approximate similarity joins over dense vector embeddings. In *Proceedings of SBBD 2023*, pages 51–62. SBC.

Saouabe, A., Tkatek, S., Oualla, H., and Mourtaji, I. (2024). Image indexing approaches for enhanced content-based image retrieval: An overview.

Sharma, S., Gupta, V., and Juneja, M. (2019). A survey of image data indexing techniques. *Artificial Intelligence Review*, 52(2):1189–1266.

Traina Jr, C., Traina, A., Seeger, B., and Faloutsos, C. (2000). Slim-trees: High performance metric trees minimizing overlap between nodes. In *International Conference on Extending Database Technology*, pages 51–65. Springer.

Vadivel, A., Majumdar, A., and Sural, S. (2003). Performance comparison of distance metrics in content-based image retrieval applications. In *International Conference on Information Technology (CIT), Bhubaneswar, India*, pages 159–164.

Yianilos, P. N. (1993). Data structures and algorithms for nearest neighbor search in general metric spaces. In *Fourth Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms (SODA)*, pages 311–321, Austin, TX.

Zaki, M. J. and Meira, W. (2020). *Data Mining and Machine Learning: Fundamental Concepts and Algorithms (https://dataminingbook.info/book_html/).* Cambridge University Press, 2nd edition.

Zezula, P., Amato, G., Dohnal, V., and Batko, M. (2006). *Similarity Search: The Metric Space Approach.* Advances in Database Systems. Springer New York, New York, NY, USA.