

Efficient Local Shape Features Matching using CUDA

Leonardo Chang^{1,2}, Miguel Arias-Estrada¹, L. Enrique Sucar¹ and José Hernández-Palancar²

1. ABSTRACT

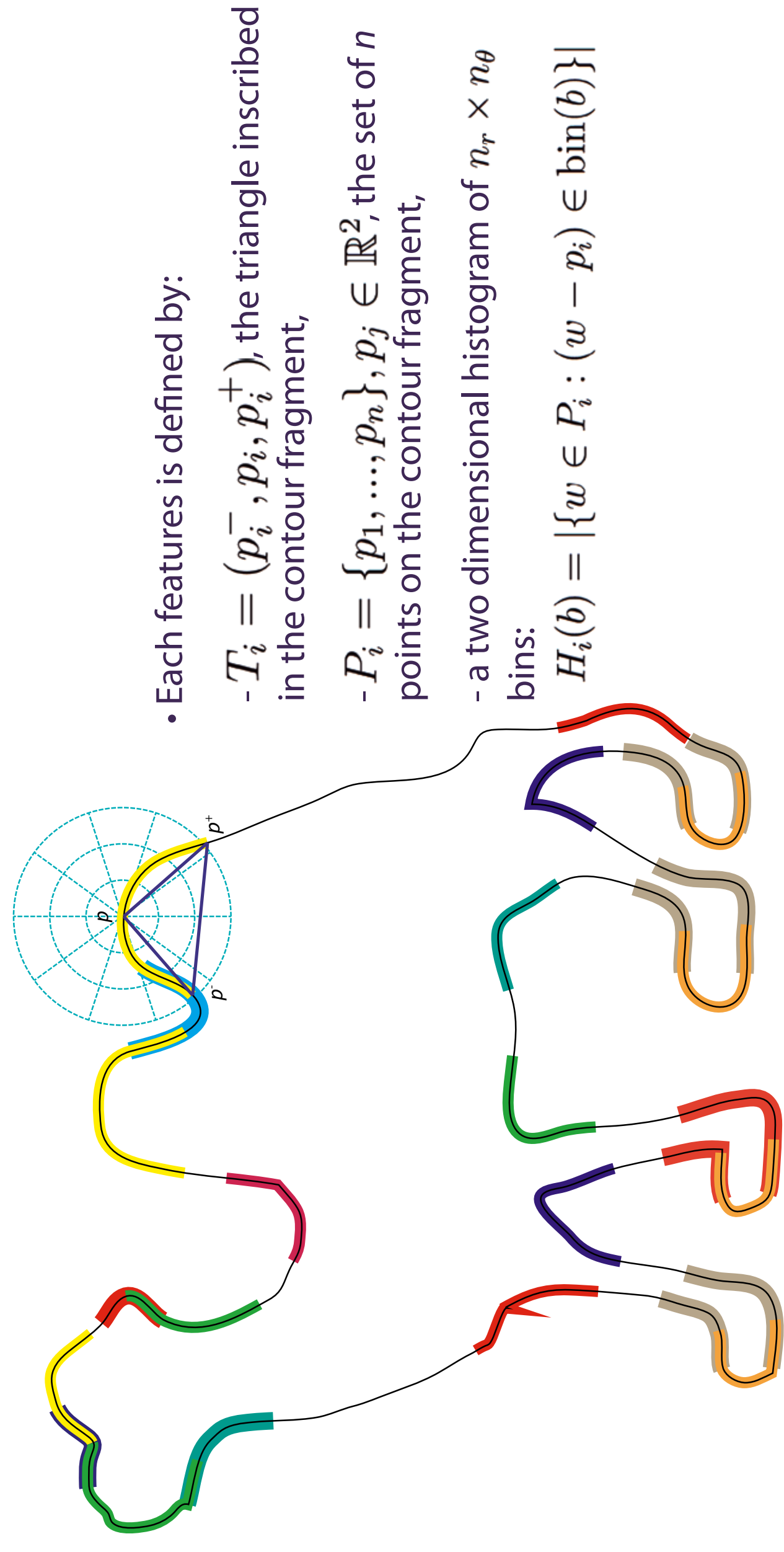
- LISF [1] is an invariant local shape features descriptor which can be used to obtain a discriminative and compact representation of an object.
- For high occlusion levels (up to 60%) LISF method outperformed other popular shape description methods, with about 20% higher bull's eye score and 25% higher precision and recall in classification.
- In this work, we present a massively parallel implementation in GPU of the most time expensive parts of LISF algorithm.
- A 34x speedup is achieved compared to the CPU implementation when matching 290 vs 290 LISF features.
- GPU implementation shows linear scaling while increasing number of features to match, in contrast to CPU implementation that shows an exponential growth.

2. MOTIVATION

- Shape descriptors have proven to be very useful in many image processing and computer vision applications, (e.g., object detection and recognition, image retrieval, object categorization, etc.)
- However, shape representation and description remains as one of the most challenging topics in computer vision.
- Most important considerations: robustness to image scale, rotation, translation, occlusion, noise and viewpoint.
- Most research assumes an effective and flawless segmentation. Not appropriate for some real applications where there is noise or occlusion, or an accurate segmentation is unavailable.
- Most research focus on accuracy. Not appropriate for some real applications where time is a critical issue.

3. THE INVARIANT LOCAL SHAPE FEATURES

- In this work, boundary and internal edge fragments are used to represent the shape of an object.
- When using contour fragments, we do not have to deal with the inherent limitations of using global features.
- In addition, there is not restriction about only dealing with closed contours or silhouettes, the features are extracted from both the contour and the internal edges of the object.
- LISF: extraction and description methods are invariant to rotation, translation, scale and present certain robustness to partial occlusion.



¹ Instituto Nacional de Astrofísica, Óptica y Electrónica (INAOE)
Luis Enrique Erro No.1, C.P. 72840, Tonantzintla, Puebla, Mexico.
{lchang, ariasmo, esucar}@ccc.inaoep.mx

² Advanced Technologies Application Centre (CENATAV)
7thA No. 21406, Playa, C.P. 12200, La Habana, Cuba.
{lchang, jpalancar}@cenatav.co.cu

4. FEATURE MATCHING

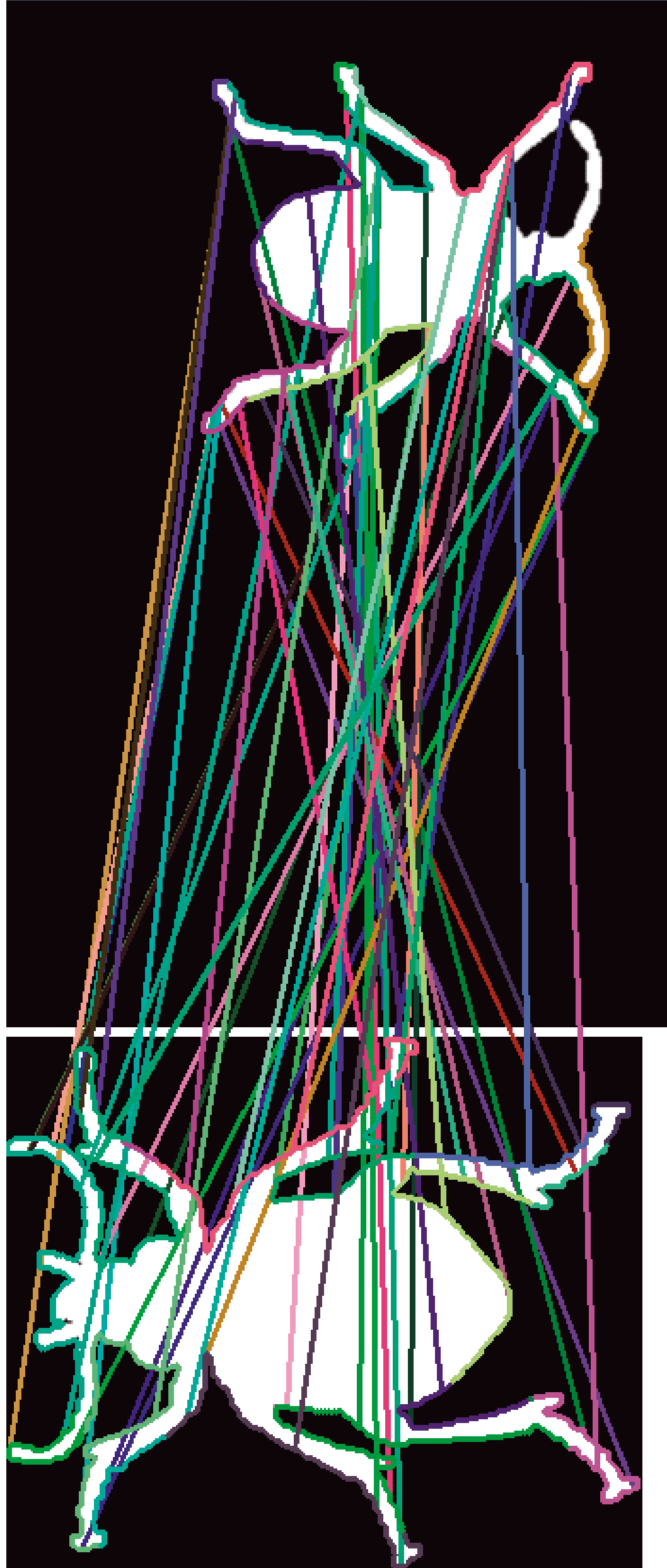
- Feature matching takes into account the structure and spatial organization of the features.
- Matches are validated by rejecting casual or wrong matches.

4.1. FINDING CANDIDATE MATCHES

- For each feature in the query, its K nearest neighbors in the database image are found by comparing their descriptors.
- In this work we use chi-squared distance to compare histograms. $\left(\chi^2 = \sum_i \frac{(m_i - n_i)^2}{m_i + n_i}\right)$
- To provide the method with rotation invariance the feature descriptors are normalized in terms of orientation, (i.e., all features are set to orientation zero).

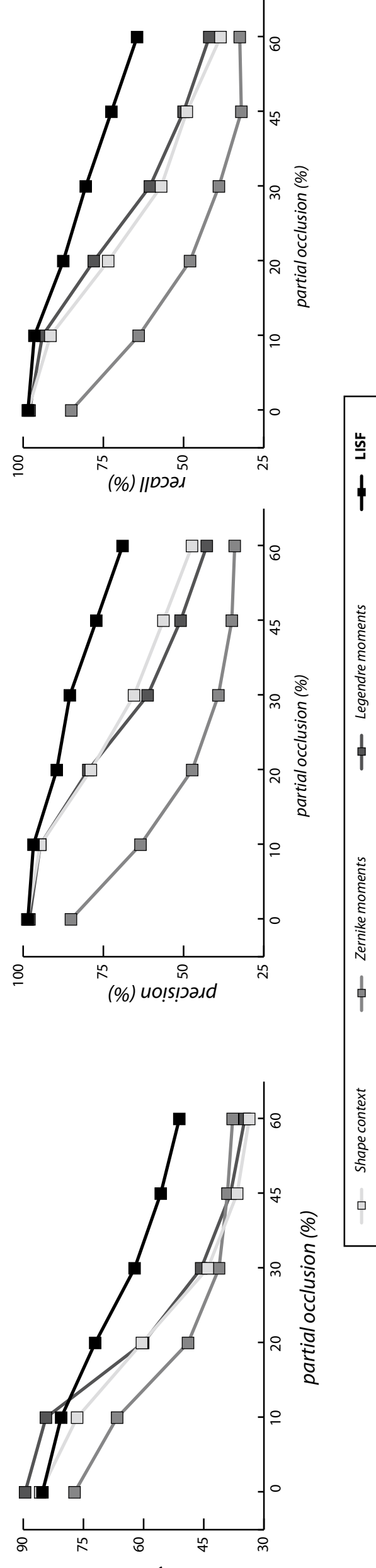
4.2. REJECTING CASUAL MATCHES

- For each candidate match, features on both images are aligned according to this correspondence scale, rotation and translation.
- For each aligned feature in the query image, its nearest neighbor in the database image is found.
- This nearest neighbor votes for a center position in the candidate image.
- If the candidate match is a correct one, most of the center votes should be concentrated around the object centroid.
- This idea is what allows us to accept or reject a candidate match, using a measure of dispersion.



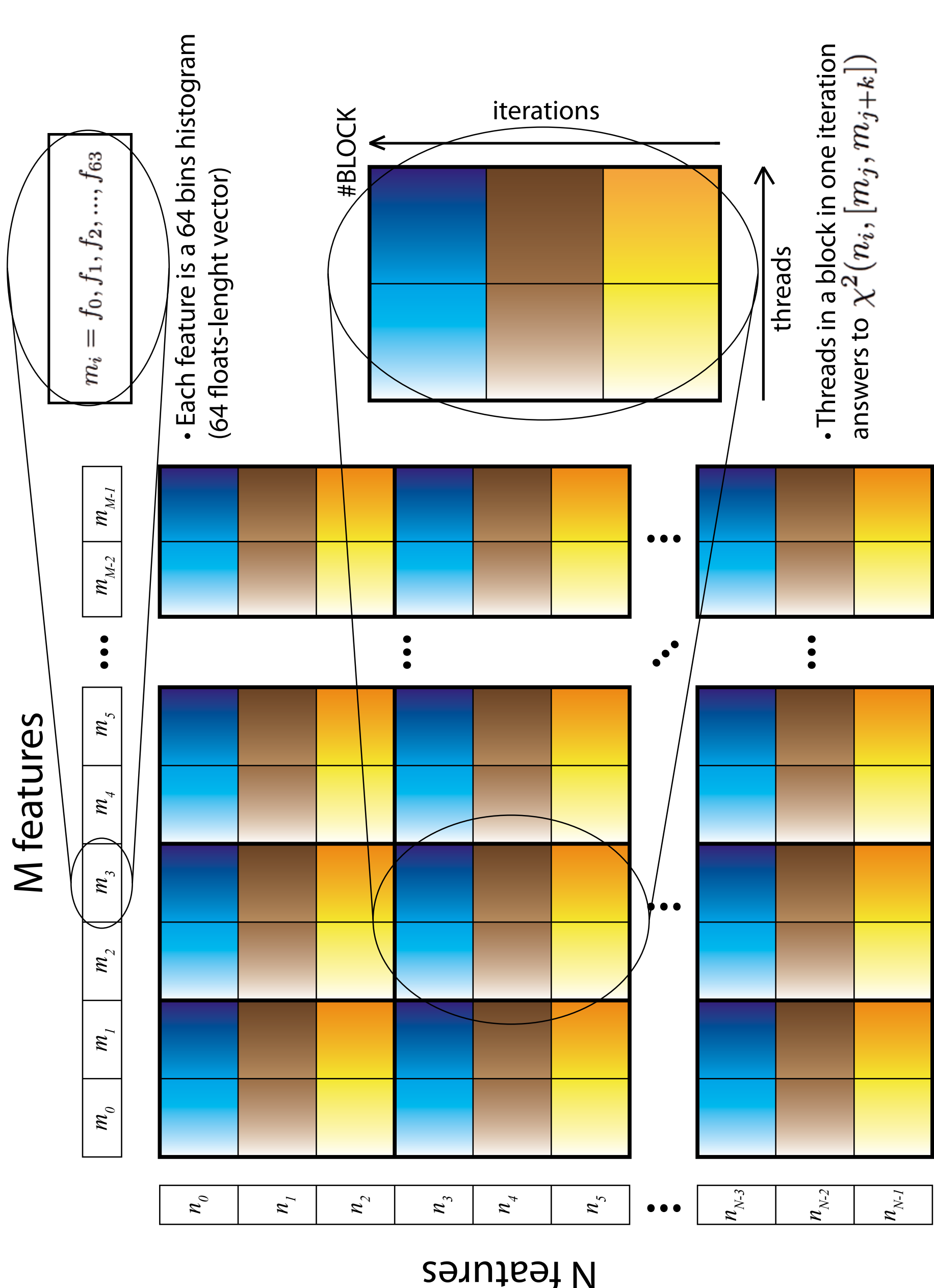
5. EXPERIMENTAL RESULTS

- MPEG-7 CE-Shape-1 dataset
- 10 classes, 20 images per class.
- Artificially introduced different levels of partial occlusion (0%, 10%, 20%, 30%, 45%, 60%).
- 2 experiments, shape retrieval and classification..
- Comparison with Zernike moments [2], Legendre moments [3] and shape context [4].



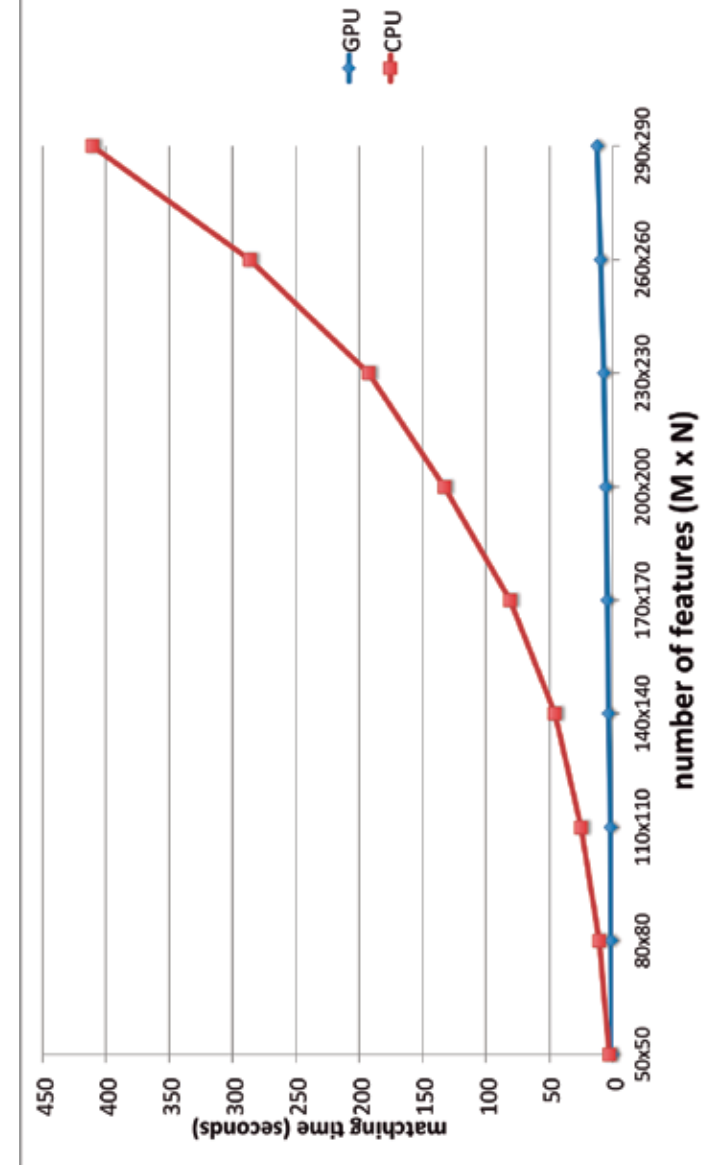
6. IMPLEMENTATION OF FEATURE MATCHING USING CUDA

- Finding candidate matches involves $M \times N$ chi-squared comparisons of feature descriptors.
- Also, rejecting casual matches needs $M \times N$ chi-squared comparisons after alignment.
- Inputs: m , descriptors of query image features (M histograms of 64 bins concatenated).
- n , descriptors of database image features (N histograms of 64 bins concatenated).
- Each value in m is used N times.
- In order to increase data reutilization and decrease global memory accesses, m and n are tiled into the shared memory.
- For values of M and N , such that the features and comparison results do not fit in the device memory, the matrix could be partitioned and the kernel launched several times.



7. GPU PERFORMANCE

- Computed on a NVIDIA GeForce GTX 480.
- So, grid size is $\langle (M+7)/8, (N+7)/8 \rangle$ and the number of threads per block is $8^3 \times 64$.



8. CONCLUSIONS

- GPU implementation linear scaling vs. CPU exponential scaling.
- A 34x speedup is achieved compared to the CPU implementation when matching 290 vs 290 LISF features.

REFERENCES

[1] L. Chang, M. Arias-Estrada, L. E. Sucar and J. Hernández-Palancar. LISF: An Invariant Local Shape Features Descriptor Robust to Occlusion. *Submitted to 26th IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2013)*.

[2] A. Khotanzad and Y. H. Hong. Rotation invariant pattern recognition using zernike moments. *9th International Conference on Pattern Recognition*, 1988, pages 326–328 vol.1, 1988.

ACKNOWLEDGEMENTS

This project was supported in part by CONACYT grant No. 103878. L. Chang was supported in part by CONACYT scholarship No. 240251.