A PCA-Based Binning Approach for Matching to Large SIFT Databases

CRV 2010

Geoffrey Treen and Anthony Whitehead
Carleton University

Outline

1. **Introduction**
2. **Background: Handed-Hierarchical Matching (SIFT-HHM)**
    - 3.1 Keypoint Descriptor Properties
    - 3.2 HHM Algorithm
    - 3.3 Experimental Results
3. **PCA-Based Binning Algorithm for Matching to Large Databases**
    - 4.1 Principal Component Vector ($\underline{PCV}$), $\underline{e}_{PCV}$ and $\underline{\delta e}_{PCV}$
    - 4.2 Database Sorting
    - 4.3 Database Querying
    - 4.4 Experimental Results
4. **Conclusions and Future Work**
5. **Questions?**
6. **References**

1. Introduction: Why SIFT?

- Invariant to scale, rotation, linear illumination
- Partially invariant to 3D viewpoint change
- SIFT features used for:
  - Content-based image retrieval
  - Video event classification
  - Object recognition / tracking
  - Image classification
  - Markerless motion capture
  - Building panoramas
  - Mobile surveillance
  - Face authentication
  - etc.

1. Introduction : SIFT

- **Main drawbacks**
  - Computation speed (takes about 1s to compute 1000 SIFT features in a typical image on a standard dual-core processor)
  - Matching: Nearest-neighbour search in 128-dimensional space (for 128-byte standard SIFT vector)

1 Introduction: PCA-SIFT, SURF

- **PCA-SIFT:**
  - 3042-element SIFT input vector reduced to 36 elements through PCA
- **SURF:**
  - Scale-space maxima found using determinant of an approximated Hessian matrix
  - 64 elements, constructed from image patch Haar wavelet responses
  - Sign of the trace of the Hessian used as a 65^{th} element, to split databases in half for faster matching

1. Introduction: Approximate Nearest-Neighbour Search Techniques

- **Space-partitioning data structures (kd-tree, k-means tree)**
- **kd-tree:**
  - Data is split along axes in order of decreasing variance
- **k-means tree:**
  - Data clustered, recursively (using k-means algorithm) into k distinct groups
  - Recursion stops when the number of points in a region < k
- **Best-Bin First Algorithm (Beis and Lowe):**
  - Searches kd-tree nodes in order of distance from query point
  - Limit on number of nodes examined
- **Randomized kd-trees (Silpa-Anan and Hartley):**
  - Parallel implementation using multiple kd-trees
  - Data split randomly among D dimensions that show greatest variance
- **Approximate k-means tree search (Muja and Lowe):**
  - After a single tree traversal, unexplored nodes searched in order of distance from query point (mean value of clustered points at a branch)
  - Limit on number of nodes examined

1. Introduction: Approximate Nearest-Neighbour Search Techniques

- **FLANN (Fast Library for Approximate Nearest-Neighbours):**
  - Developed by Muja and Lowe (2009)
  - Publicly-available library of approximate nearest-neighbour search algorithms
  - Automatically chooses the best algorithm for a given dataset and desired precision
  - Achieves better than 1000 X over a linear search is possible while finding 90% of the closest matches (31-million SIFT feature dataset)
  - Current state-of-the art for large-database SIFT matching

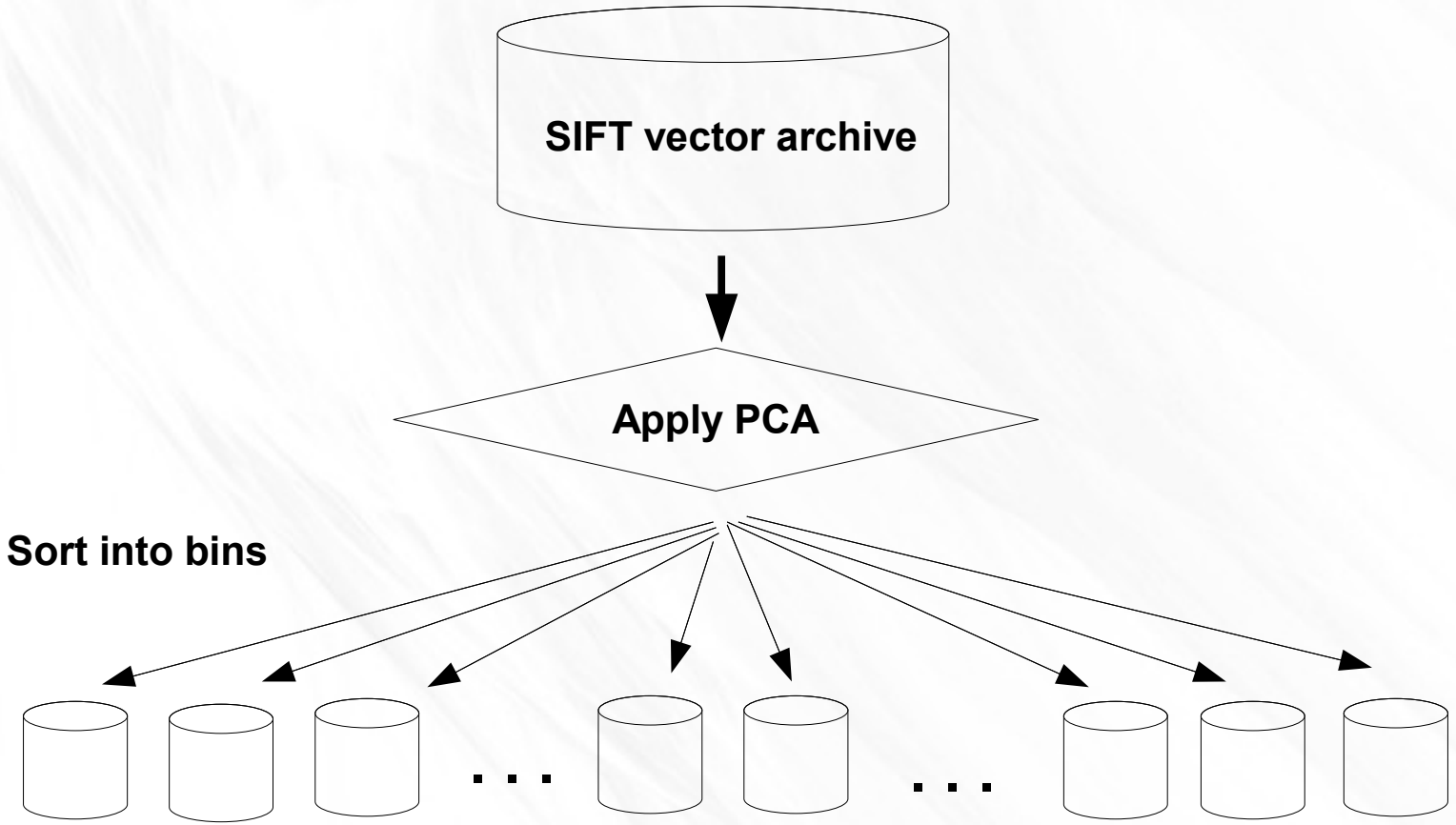
1. Introduction: Main contribution

- **Previous work:**
  - **SIFT-HHM,** which finds SIFT correspondences in image pairs in about 7% of the time it takes for a linear search (presented at the 2009 IEEE Workshop on Applications of Computer Vision (WACV)

- **Main contribution of this paper:**
  - **PCA binning algorithm**, which – using SIFT-HHM at the lowest level – can find correspondences in SIFT vector archives three times faster (for the same level of recall-precision) than FLANN

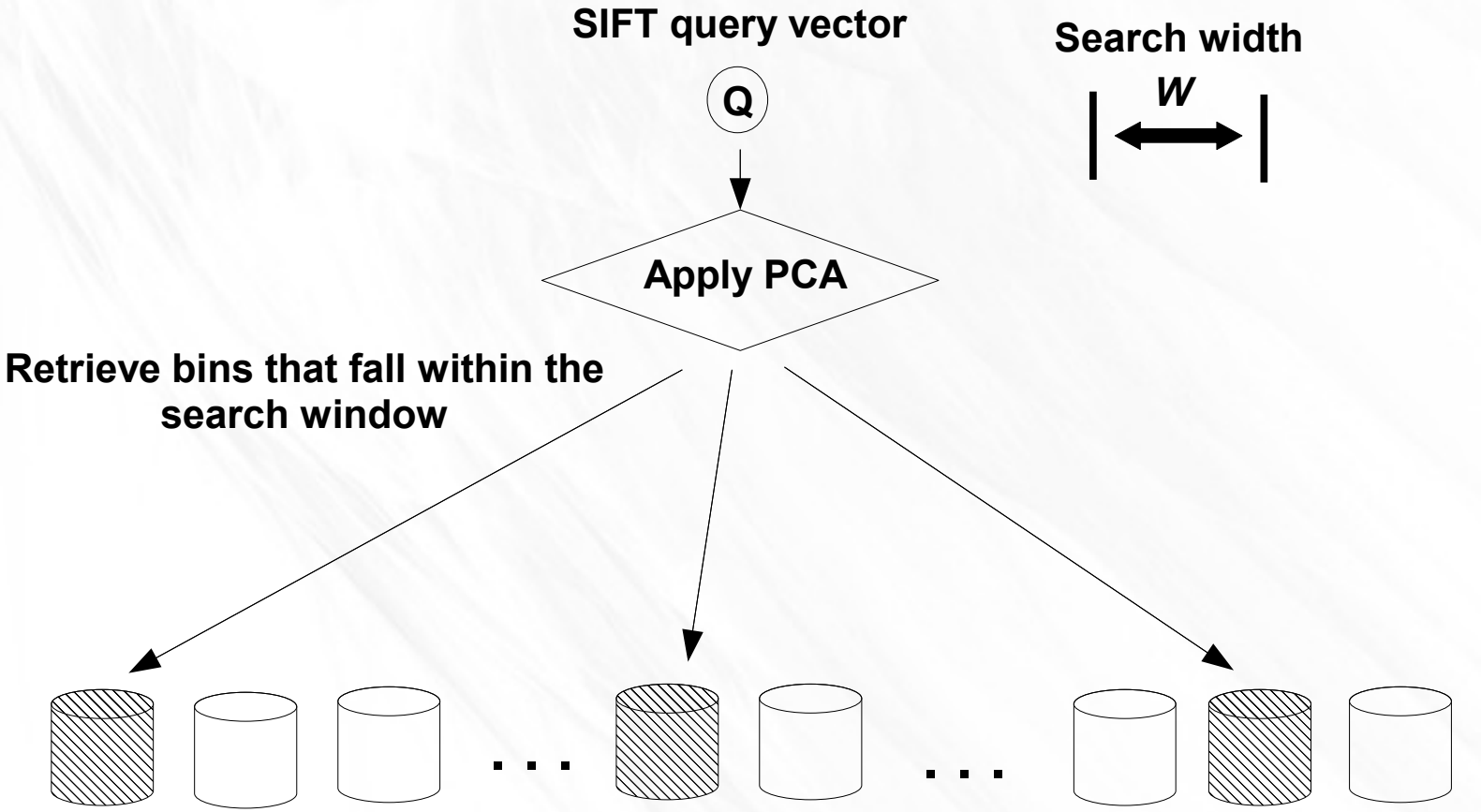
1. Introduction: Overview of binning approach

- Basic concept – sorting:



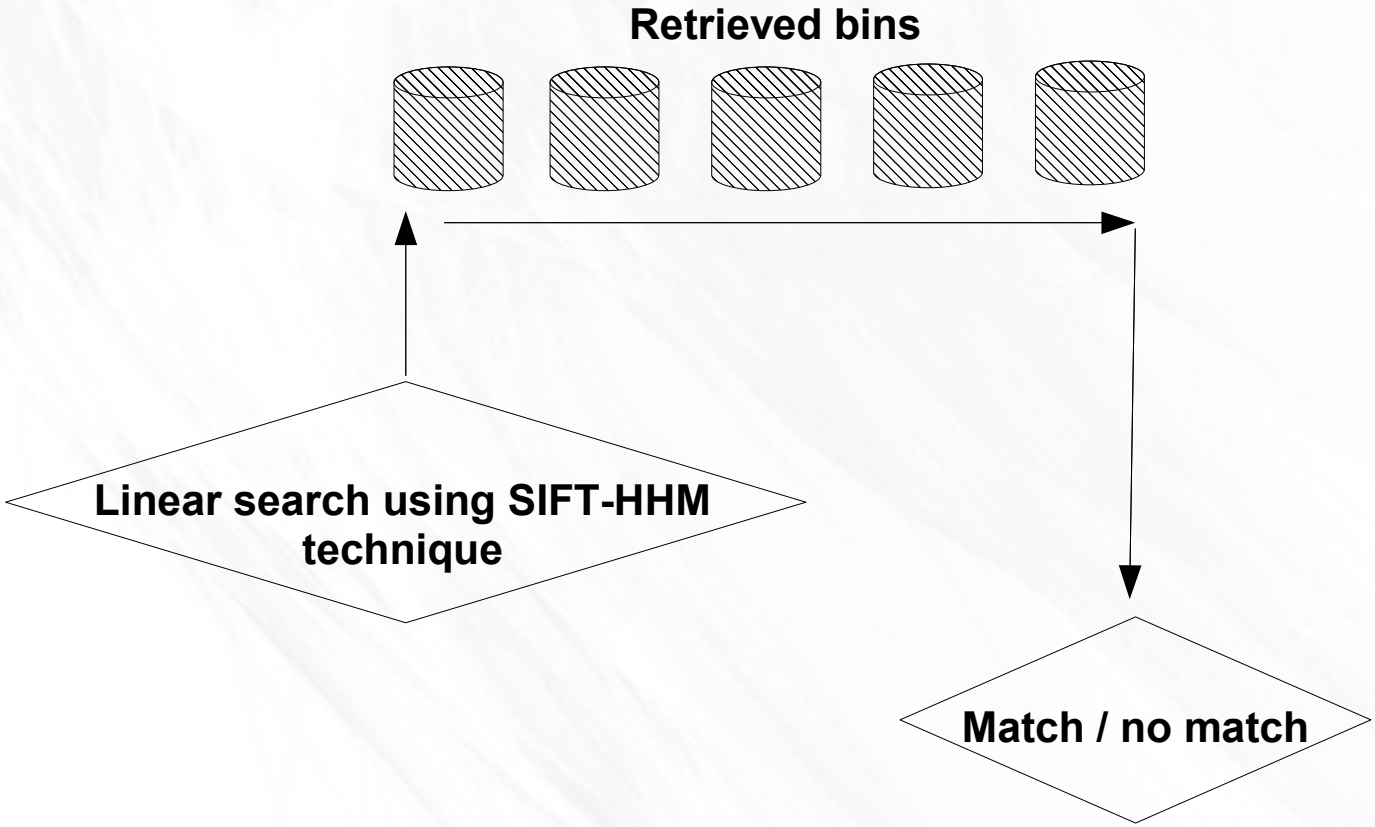
1. Introduction: Overview of binning approach

- Basic concept – querying:



1. Introduction: Overview of binning approach

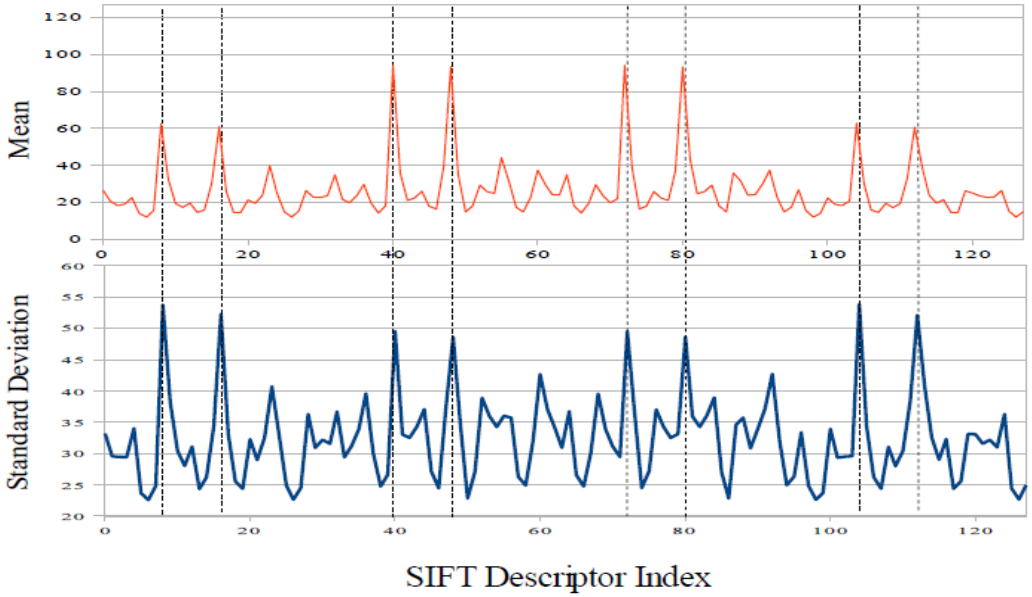
- Querying (cont'd):

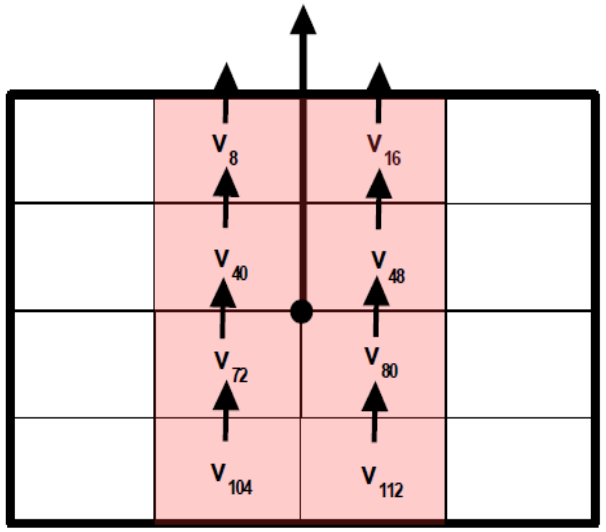


2. Background: Handed-Hierarchical Matching (SIFT-HHM)

2.1 SIFT-HHM: Keypoint Descriptor Properties

- SIFT descriptor means and standard deviations:





- 8 most-variant elements (8, 16, 40, 48, 72, 80, 104, 112) labelled “primary elements”
- 4 primary elements closest to keypoint centre labelled (40, 48, 72, 80) “inner primaries”

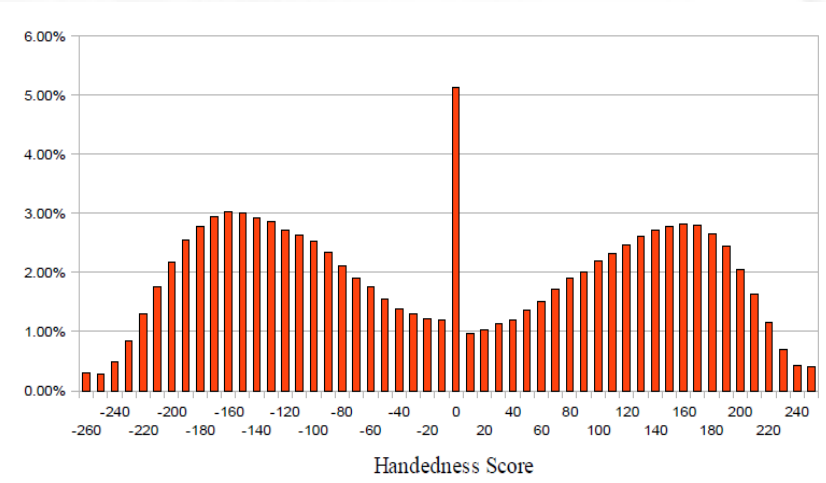
SIFT-HHM: Keypoint Descriptor Properties (cont'd)

- Pearson's correlation coefficient for inner primaries:

	v_{40}	v_{48}	v_{72}	v_{80}
v_{40}	1.000	-0.123	0.786	-0.186
v_{48}	-0.123	1.000	-0.180	0.802
v_{72}	0.786	-0.180	1.000	-0.123
v_{80}	-0.186	0.802	-0.123	1.000

- "Handedness" score and distribution:

$$h = (v_{48} + v_{80}) - (v_{40} + v_{72})$$

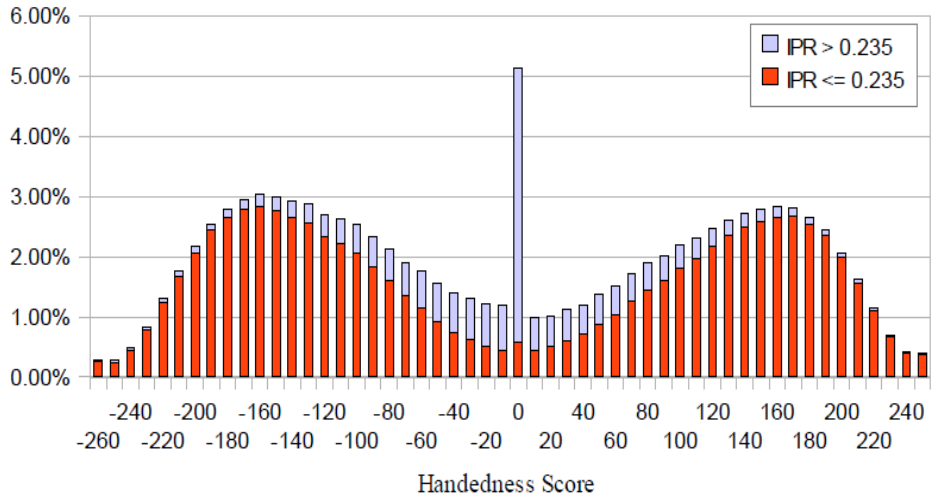


2.1 SIFT-HHM: Keypoint Descriptor Properties (cont'd)

- Inner primary ratio (IPR):

$$IPR = \frac{(v_{40}^2 + v_{48}^2 + v_{72}^2 + v_{80}^2)}{(\sum_{0}^{127} v_i^2)}$$

- Handedness score distribution (with and without IPR filtering):



2.2 SIFT-HHM: Algorithm

1. **IPR filtering:**
   - Remove SIFT vectors that have IPR > threshold
   - Will improve overall matching performance
2. **Handedness split:**
   - Split SIFT databases in two based on handedness score
   - If h < 0, sort vectors into “left” bin, if h >=, sort into “right” bin
3. **Hierarchical Euclidean-distance matching:**
   - Treat SIFT vector hierarchically, matching primary elements first
   - If primary Euclidean distance is above a threshold, reject keypoint as a potential match (eliminates 97% of vectors on first pass)
   - If primary Euclidean distance is below the threshold, save vector (and distance value) into a cache and match on the second pass using the rest of the descriptor vector

2.3 SIFT-HHM: Experimental Results

	Recall	Precision	F1 Score	Normalized Matching Time (s)
SIFT (Linear Search)	0.684	0.972	0.803	1.210
PCA-SIFT	0.721	0.940	0.816	0.366
SURF	0.700	0.962	0.797	0.294
SIFT-HHM	0.682	0.975	0.803	0.081

	Recall	Precision	F1 Score	Normalized Matching Time (s)
SIFT (Linear Search)	0.775	0.987	0.869	1.092
PCA-SIFT	0.776	0.982	0.867	0.339
SURF	0.760	0.953	0.846	0.269
SIFT-HHM	0.765	0.987	0.862	0.072

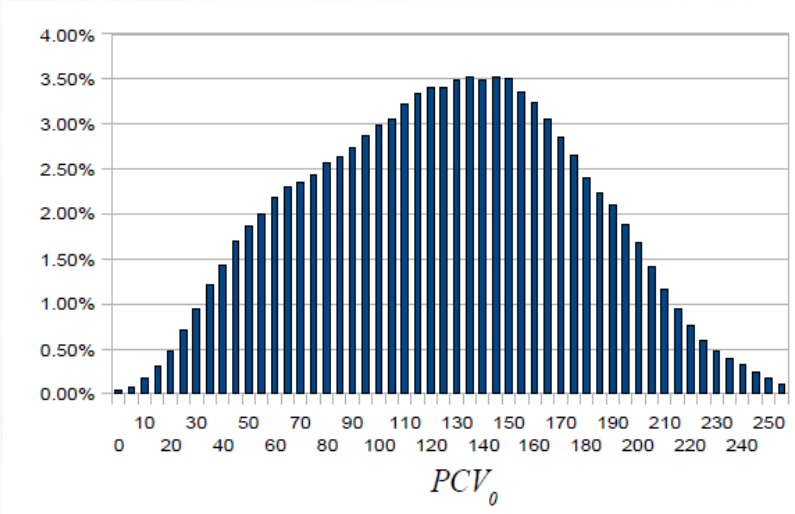
3. PCA-Based Binning Algorithm for Matching to Large Keypoint Databases

3.1 PCA Binning: $\underline{PCV}$ and $\underline{e}_{PCV}$

- Principal Component Vector (PCV):

$$\underline{PCV} = Norm\{KLT\{\underline{v}\}\}$$

SIFT descriptor vector



- PCV error vector ($\underline{e}_{PCV}$):

$$\underline{e}_{PCV} = \underline{PCV} - \underline{PCV'}$$

Before random transformation

After random transformation

3.1 PCA Binning: $\underline{\delta e}_{PCV}$

- Standard deviation of $\underline{e}_{PCV}$ (first 16 elements):

$\underline{e}_{PCV}$ Index	δ	3δ	$\underline{e}_{PCV}$ Index	δ	3δ
0	7.70	23.10	8	11.09	33.27
1	6.72	20.16	9	12.65	37.95
2	7.60	22.80	10	12.71	38.13
3	9.03	27.09	11	11.84	35.52
4	7.95	23.85	12	13.19	39.57
5	9.72	29.16	13	11.28	33.84
6	12.10	36.30	14	12.64	37.92
7	10.24	30.72	15	12.16	36.48

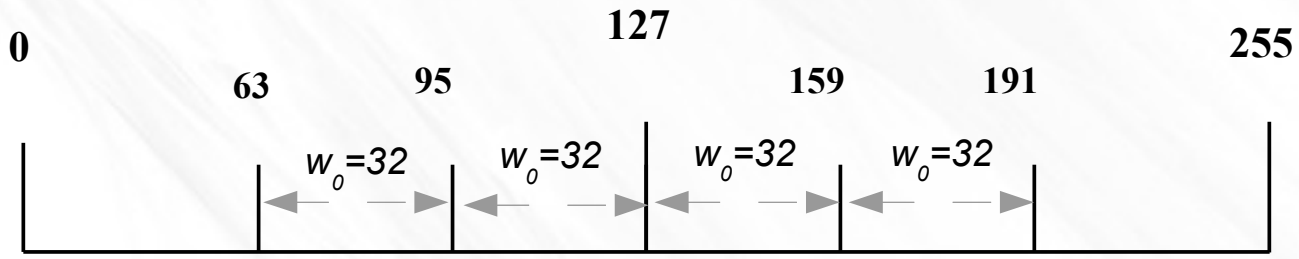
$$\underline{\delta}_{ePCV} = \{ \delta_{ePCV0}, \delta_{ePCV1}, \delta_{ePCV2}, \ldots, \delta_{ePCVN} \}$$

3.2 PCA Binning: Database Sorting

- Bin widths calculated from $\underline{\delta e}_{PCV}$:

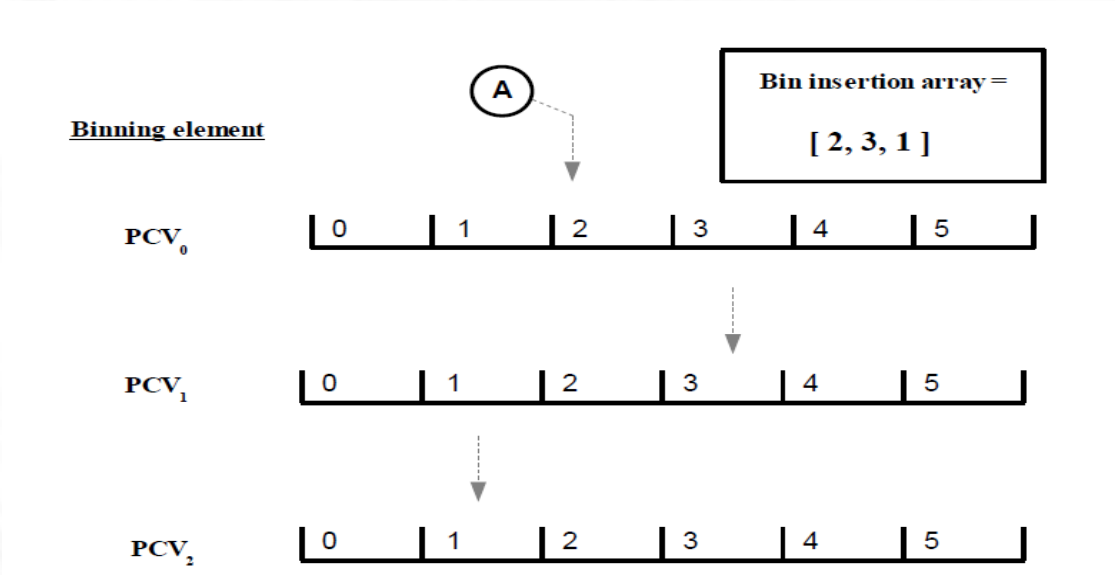
$$w_i = 4 * roundup(\underline{\delta}_{ePCVi})$$

- For PCV element 0, $w_0 = 32$



3.2 PCA Binning: Database Sorting

- Keypoint insertion example:



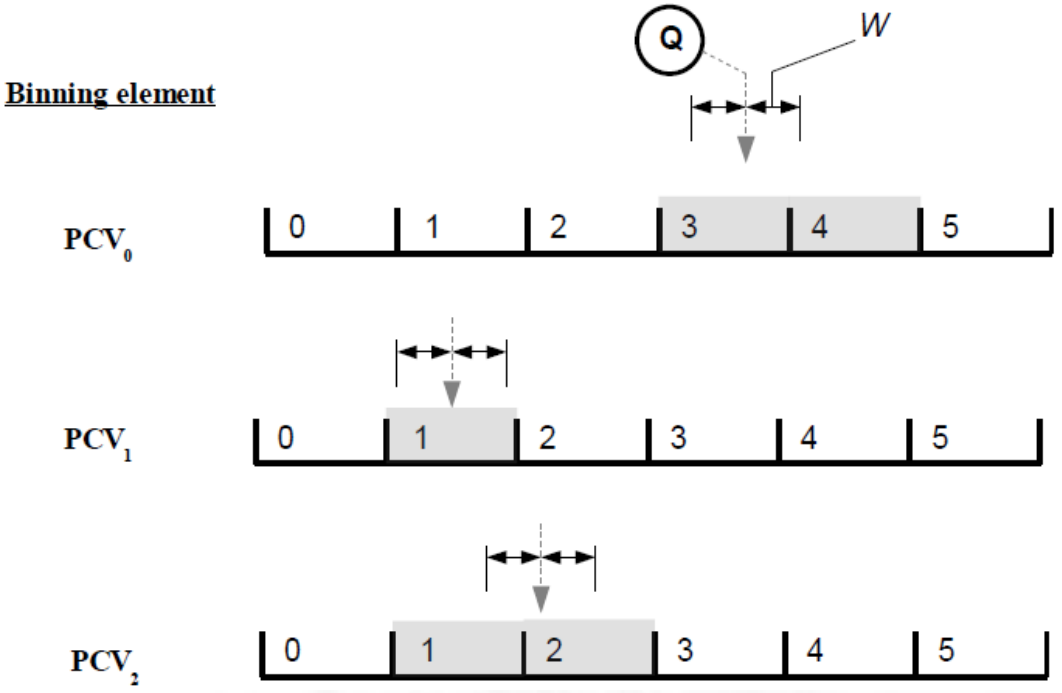
$$unique\ bin\ key = \sum_{i=0}^{N-1} b_i \left(\prod_{j=i+1}^{N-1} n_j \right)$$

Bin insertion array index

Number of bins at index j

3.3 PCA Binning: Database Querying

- Querying example:

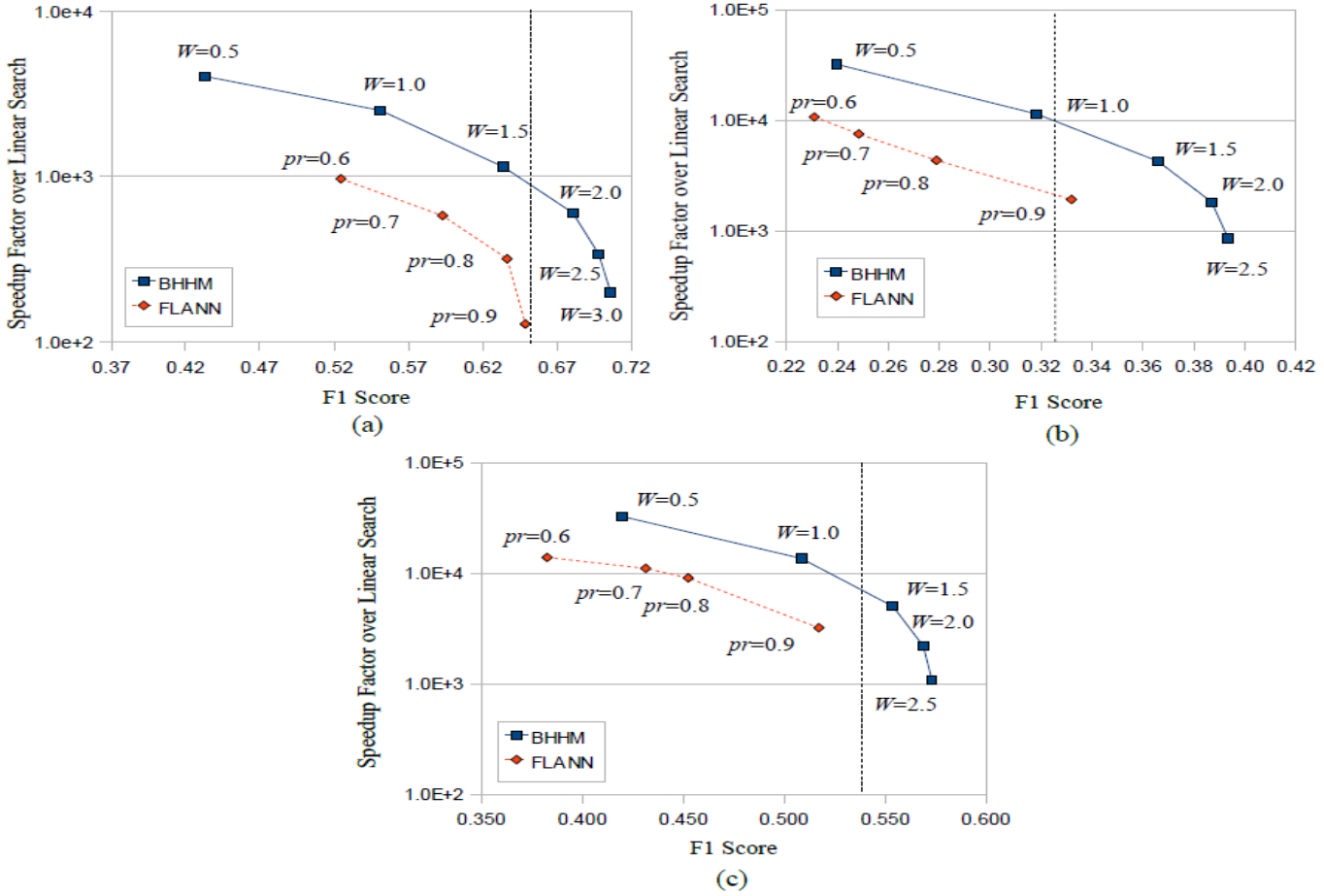


Bin retrieval arrays:

[3, 1, 1]
[3, 1, 2]
[4, 1, 1]
[4, 1, 2]

3.4 PCA Binning: Experimental Results

- Comparison against FLANN using SIFT databases of 100,000, 1 million and 2 million vectors



3.4 PCA Binning: Experimental Results (cont'd)

- Search time required for 1000 queries (typical image) at 90% of linear-search F1 score:

Algorithm	Avg. Search Time for 1000 Query Features (s)		
	SIFT100K	SIFT1M	SIFT2M
Linear Search	123	1233.7	2549.9
FLANN	0.21	0.27	0.42
SIFT-BHHM	0.06	0.06	0.13

- Database build times:

Algorithm	Average SIFT100K Build Time (s)	Average SIFT1M Build Time (s)	Average SIFT2M Build Time (s)
FLANN	55.4	883.0	1116.4
SIFT-BHHM	3.0	70.7	90.2

4. Conclusions and Future Work

- **Areas for future work:**
  - Apply PCA-Binning to other descriptors (i.e. SURF)
  - Combine with k-means?
  - Distributed processing

5. Questions?

6. References

- **Main SIFT journal article:**

  D. G. Lowe. Distinctive image features from scale-invariant keypoints. Int. Journal of Computer Vision, 60(2): pp. 91--110, 2004.

- **PCA-SIFT:**

  A. Y. Ke and R. Sukthankar. PCA-SIFT: A more distinctive representation for local image descriptors. In Proc. CVPR, pages 506--513, June 2004.

- **SURF:**

  H. Bay, T. Tuytelaars, and L. J. V. Gool. SURF: Speeded up robust features. In Proc. Europearn Conf. Computer Vision, pages 404-417, 2006.

- **FLANN:**

  M. Muja, D. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. International Conference on Computer Vision Theory and Applications (VISAPP'09), 2009.

- **SIFT-HHM (previous paper):**

  G. Treen and A. Whitehead. Efficient SIFT Matching from Keypoint Descriptor Properties. IEEE Workshop on Applications of Computer Vision, pp. 216-222, Snowbird, Utah, December 2009.