Logo Retrieval with Parallel Matching

Nikos Pitsianis Nikos Sismanis

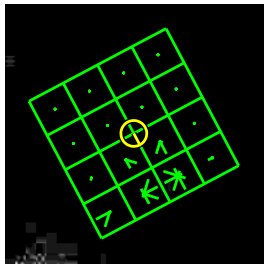
December 17, 2012

BELGA Images Logo Retrieval

- Content-based logos and trademarks retrieval in large natural image collections [1]
- Provided and copyrighted by BELGA press agency
- Composed of 10,000 images covering all aspects of life

[1] http://www-rocq.inria.fr/imedia/belga-logo.html

SIFT : Scale Invariant Feature Transform



1. Scale-space extrema detection using a difference-of-Gaussian function to identify potential interest points that are invariant to scale and orientation
2. Keypoint localization: a detailed model is fit to determine location and scale
3. Orientation assignment: to each keypoint location based on local image gradient directions
4. Keypoint descriptor: local image gradients are measured at the selected scale in the region around each keypoint [2]

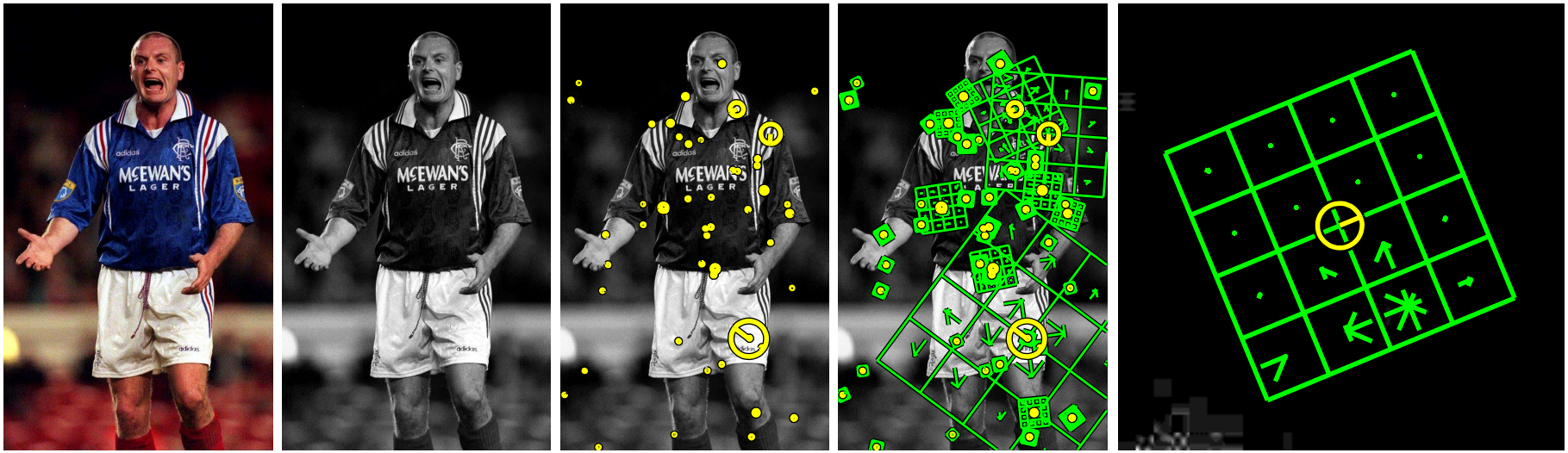
[2] David G. Lowe, Distinctive Image Features from Scale-Invariant Keypoints, IJCV, 2004

SIFT Vectors from Images

SIFT Vectors from Images

SIFT Vectors from Images

SIFT Vectors from Images



- SIFT vectors extract distinctive invariant features from images that can be used to perform reliable matching between different views of an object or scene
- Invariant to image scale and rotation
- Provide robust matching across a range of affine distortion, change in 3D viewpoint, noise, and change in illumination

Image Pattern Matching Algorithm

1. Extract SIFT vectors from known objects in a data base
2. Extract SIFT vectors from objects in query images
3. Find the nearest neighbors of each query point in the data base
4. Identify object based on the neighbors

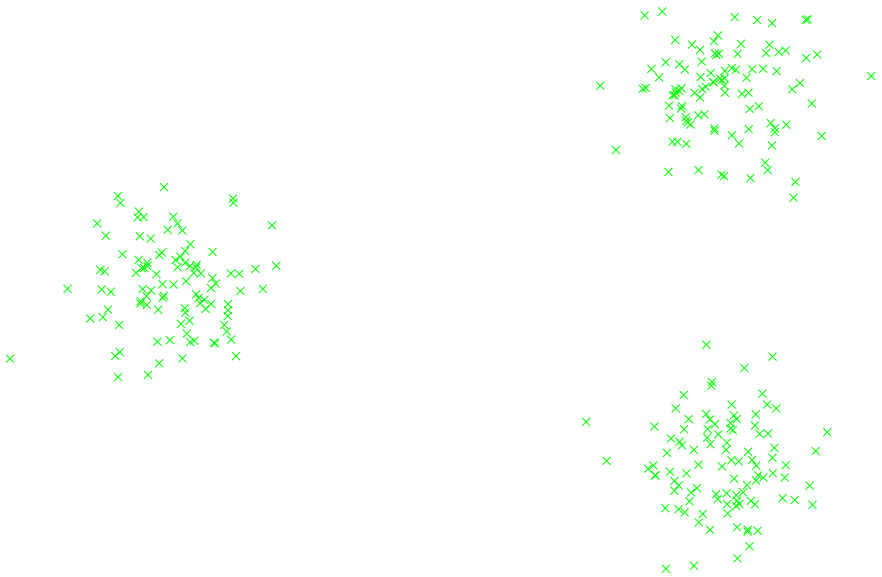
Two Basic Operations in Matching

- k-means clustering
- k-nearest neighbor search

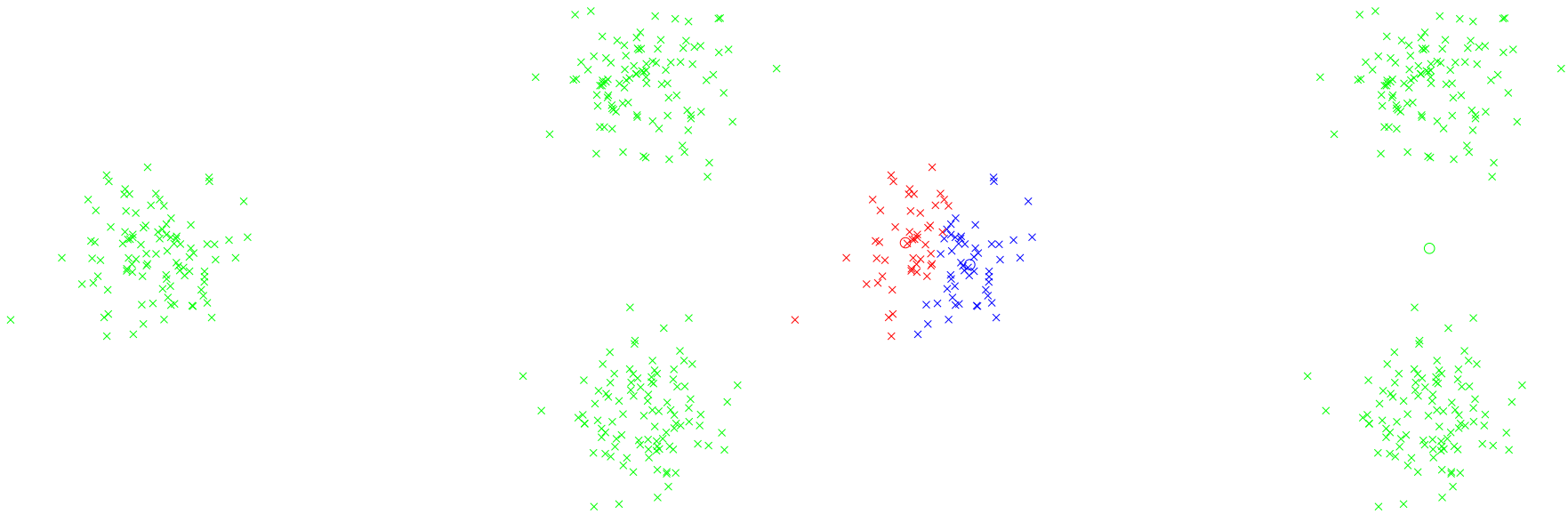
Image Pattern Matching Problem Sizes

- corpus data size n : 3 - 80 million vectors
- number of dimensions d : 128
- number of clusters k : 5 - 1000
- number of neighbors k : 2-256
- number of queries q : 100 - 500 vectors per logo
- 1000 - 8000 SIFT vectors per image

k-Means Clustering



k-Means Clustering



- Choose of the initial centroids
- Assign each element to the closest centroid and
- Calculate new centroids
- Minimize sum of square errors

$$SSE = \sum_{i=1}^{k} \sum_{j=1}^{n_i} \|x_{ij} - c_i\|^2$$

Distance Calculation Using BLAS

$$D_{ij} = \|X_i - Y_j\|^2$$

$$D = X^2 e_d e_n^T - 2XY^T + e_m e_d^T Y^{2T}$$

- Store the corpus data column-wise and the cluster centers row-wise
- All threads maintain coalesced memory accesses
- Centroids are stored in shared memory

Calculation of Centroids

- Divide the corpus data into S sub-blocks
- Assign each sub-block to a block of threads
- Sum the members of each cluster independently
- Cluster summations via reduction

k-Nearest Neighbor Search

- Find the k smallest elements and their positions from an unordered list of n numbers
- Exploit multiple independent queries
- Performance sensitive to algorithm choice, parameter size and implementation
- Explore multiple algorithm implementations and choose best combination

k-NN Selection Search

for $i = 1 \rightarrow k$ **do**
  $x(i) \leftarrow a(1)$
  $t \leftarrow 1$
  for $j = 2 \rightarrow n$ **do**
    if $(x(i) > a(j))$ **then**
      $x(i) \leftarrow a(j)$
      $t \leftarrow j$
    end if
  end for
  $a(t) \leftarrow \infty$
end for

k-NN Heap Search

$qk = \log_2(k) + 1$
for $i = 1 \rightarrow 2^{qk} - 1$ **do**
    $h(i) \leftarrow \infty$
end for
for $i = 1 \rightarrow n$ **do**
    if $a(i) < h(1)$ **then**
        $c \leftarrow 1$
        $h(c) \leftarrow a(i)$
        for $j = 1 \rightarrow qk - 1$ **do**
            $l \leftarrow 2 \cdot c$
            $r \leftarrow l + 1$
            if $h(c) \geq h(l) \cup h(c) \geq h(r)$ **then**
                BREAK
            else if $h(l) \leq h(r)$ **then**
                if $h(c) \leq h(r)$ **then**
                    $exchange(h(c), h(r))$
                    $c \leftarrow r$
                end if
            else
                if $h(c) \leq h(l)$ **then**
                    $exchange(h(c), h(l))$
                    $c \leftarrow l$
                end if
            end if
        end for
    end if
end for

k-NN Unordered Search

```
for i = 1 → k do
    x(i) ← a(i)
end for
maxval ← x(1)
maxid ← x(1)
for i = 2 → k do
    if maxval < x(i) then
        maxval ← x(i)
        maxid ← i
    end if
end for
for i = k + 1 → n do
    if maxval > a(i) then
        maxval ← x(1)
        maxid ← 1
        for j = 2 → k do
            if maxval < x(j) then
                maxval ← x(j)
                maxid ← j
            end if
        end for
    end if
end for
```

k-NN Folded Search

$P \leftarrow \frac{n}{k}$
$maxOfmins \leftarrow -\infty$
$r \leftarrow 1$
for $i = 1 \rightarrow k$ **do**
    $min \leftarrow \infty$
    for $j = 0 \rightarrow P - 1$ **do**
        if $min > a(r + j)$ **then**
            $min \leftarrow a(r + j)$
        end if
    end for
    if $maxOfmins < min$ **then**
        $maxOfmins \leftarrow min$
    end if
    $r \leftarrow r + P$
end for
$j \leftarrow 1$
for $i = 1 \leftarrow n$ **do**
    if $a(i) \leq maxOfmins$ **then**
        $a(j) \leftarrow a(i)$
        $j \leftarrow j + 1$
    end if
end for

k-NN Bitonic Search

Require: kn
  for $k = 2 \rightarrow kn$ **do**
    for $j = \frac{k}{2} \rightarrow 1$ **do**
      for $i = 0 \rightarrow N$ **do**
        $ii \leftarrow i \otimes j$
        if $ii > i$ **then**
          if $(i \oplus k) == 0 \cup a(i) > a(ii)$ **then**
            $exchange(a(i), a(ii))$
          end if
          if $(i \oplus k)! = 0 \cup a(i) < a(ii)$ **then**
            $exchange(a(i), a(ii))$
          end if
        end if
      end for
      $j \leftarrow \frac{j}{2}$
    end for
    $k \leftarrow 2 \cdot k$
  end for

Hardware Specs

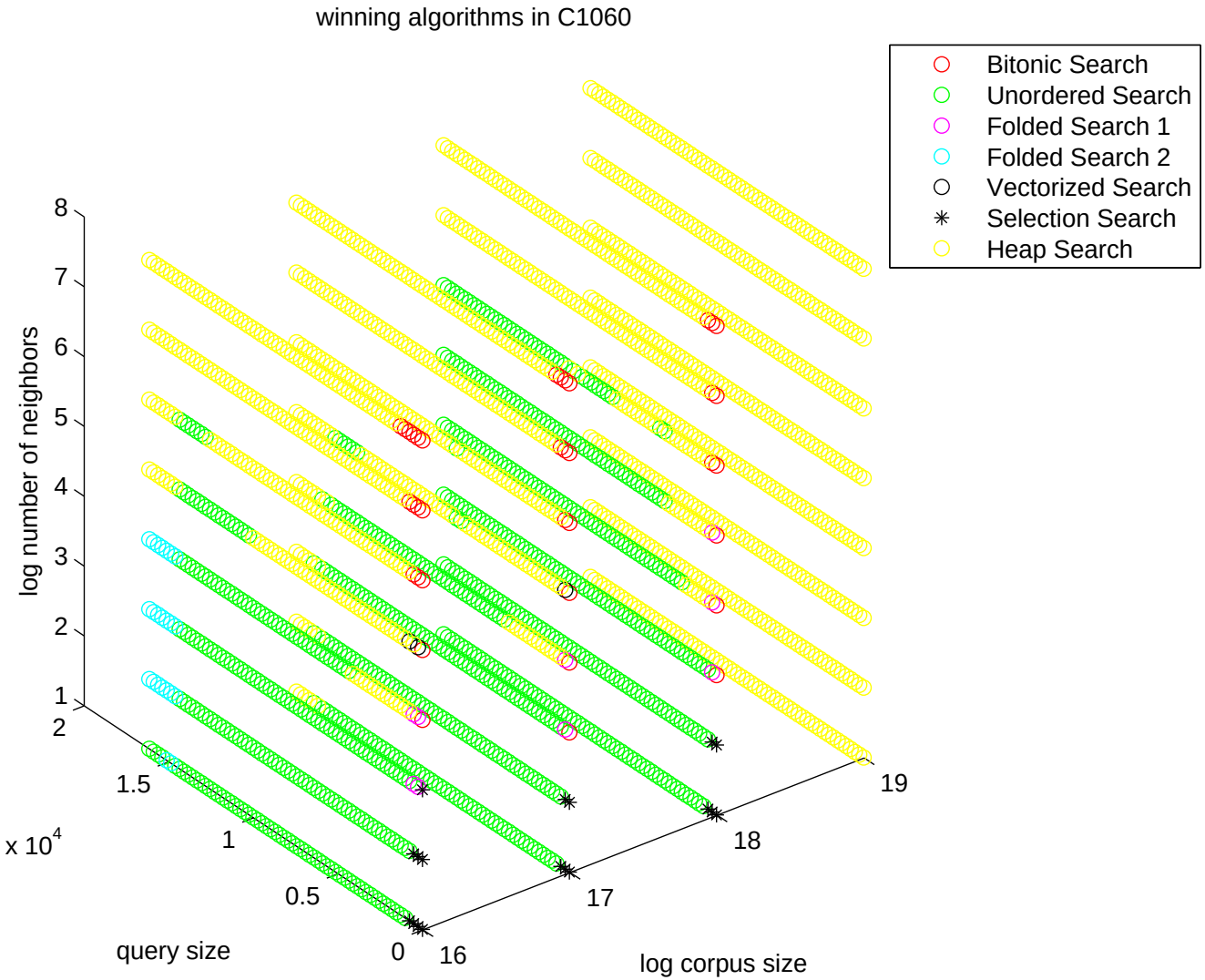
Specification	C1060	GTX-480
GPU architecture	Tesla	Fermi
CUDA cores	240	480
Multiprocessors	30	15
Shared Memory per multiprocessor	16	48
Max number of threads per block	512	1024
Core clock (GHz)	1.296	1.401
Memory size (GB)	4	1.5
Memory clock (GHz)	0.8	1.848

CPU : 2x AMD Opteron 6168 1.9GHz with 24-cores total, 12MB L3, 32GB GDDR3

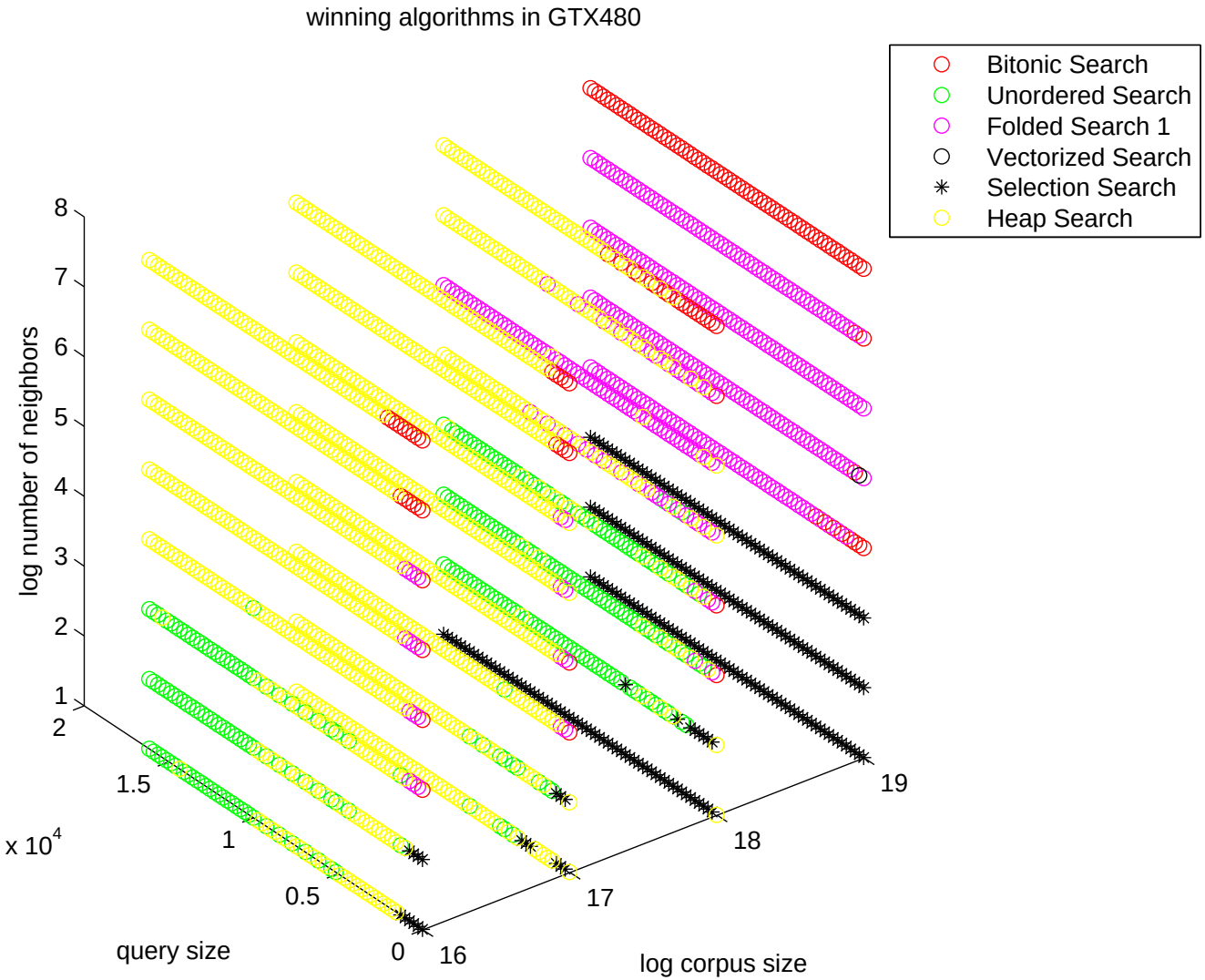
Implementations and Data Sets

- Software
  - Rodinia Benchmark Suite V 2.0.1 `lava.cs.virginia.edu/Rodinia`
  - VLFeat V 0.9.14 `www.vlfeat.org`
  - Fast K Nearest Neighbors on GPU (release 01/27/2010) `www.i3s.unice.fr/čreative/KNN`
- Data Sets
  - 1999 KDD-Cup `kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`
  - Caltech-101 Image Library `www.vision.caltech.edu`
  - BELGA images `www.belga.be`

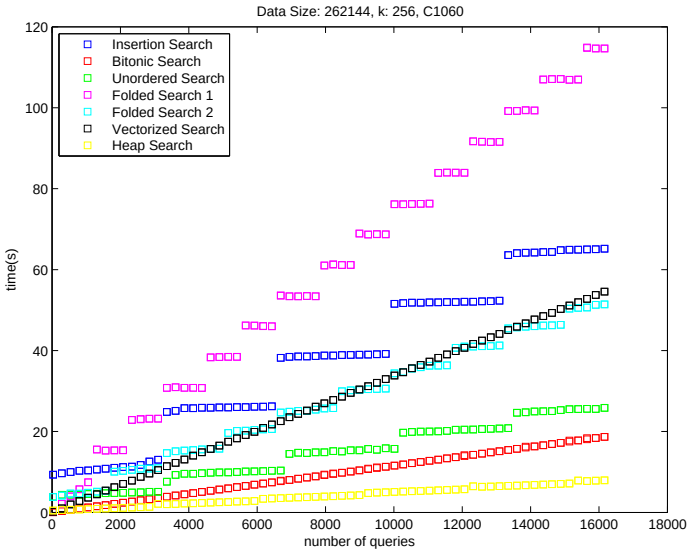
C1060 k-NNS Algorithmic Selection

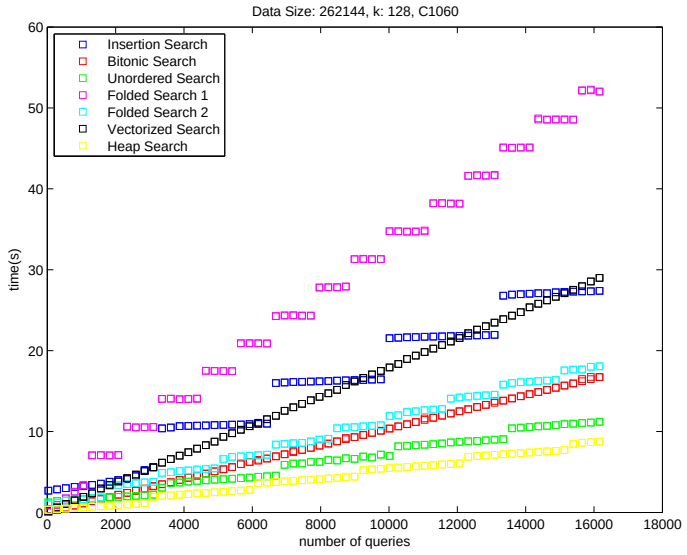


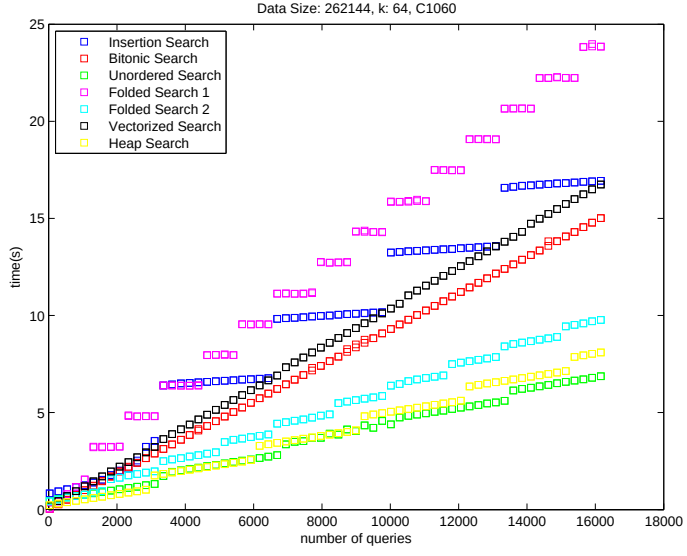
GTX-480 k-NNS Algorithmic Selection

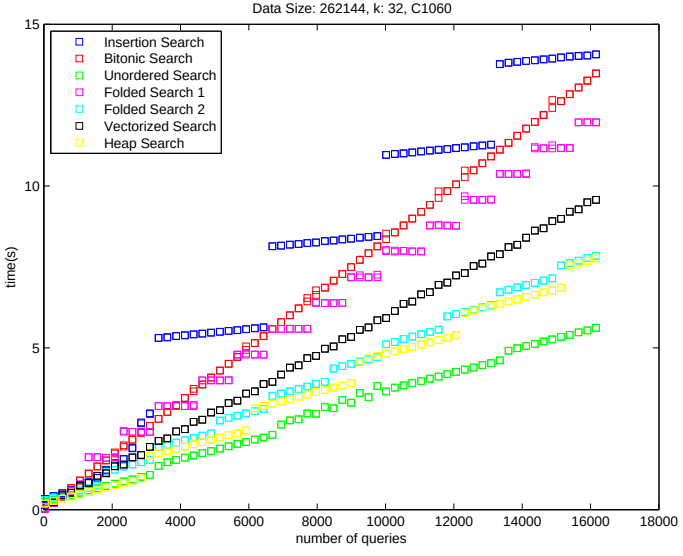


C1060 *k*-NNS Performance Detail

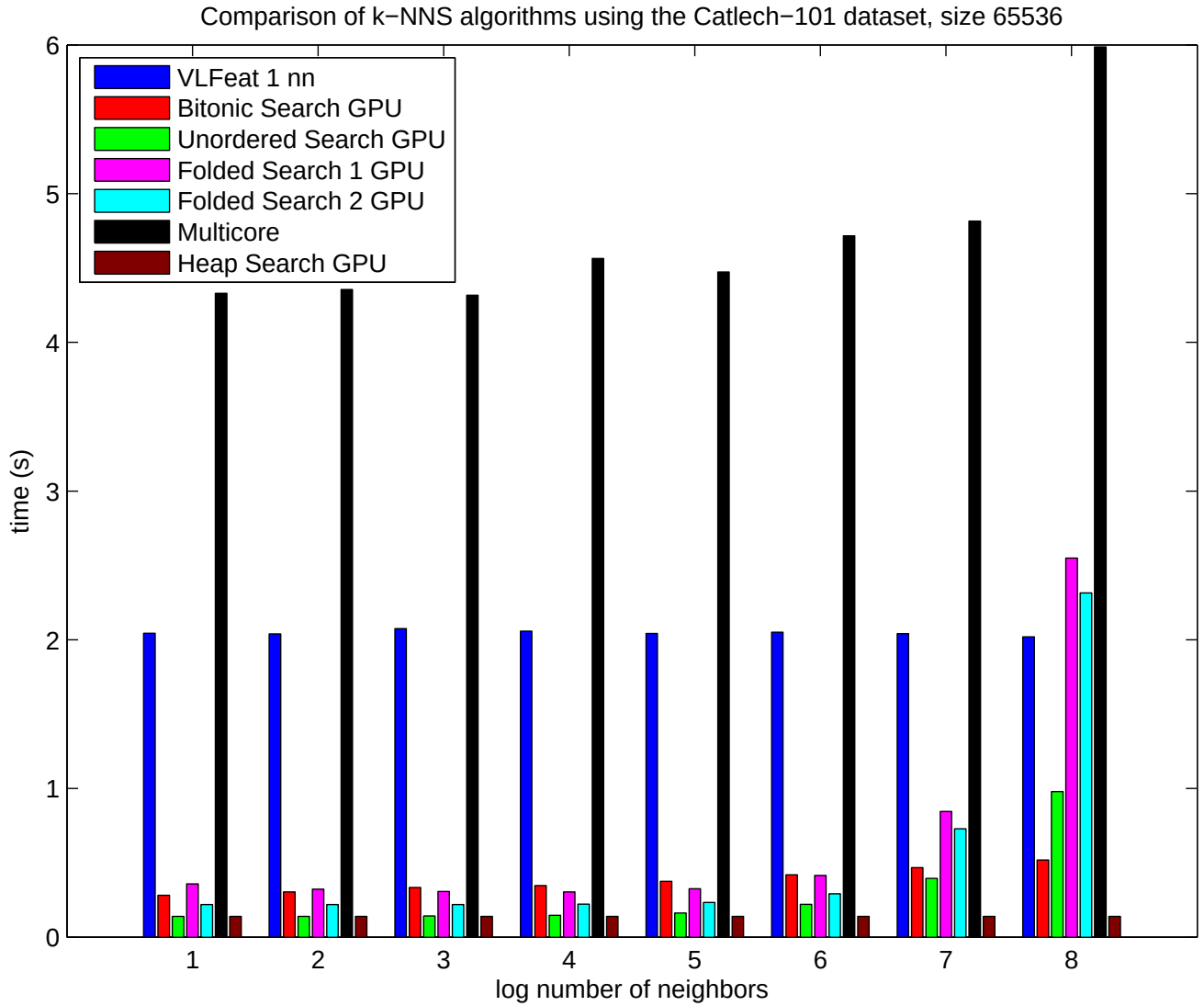




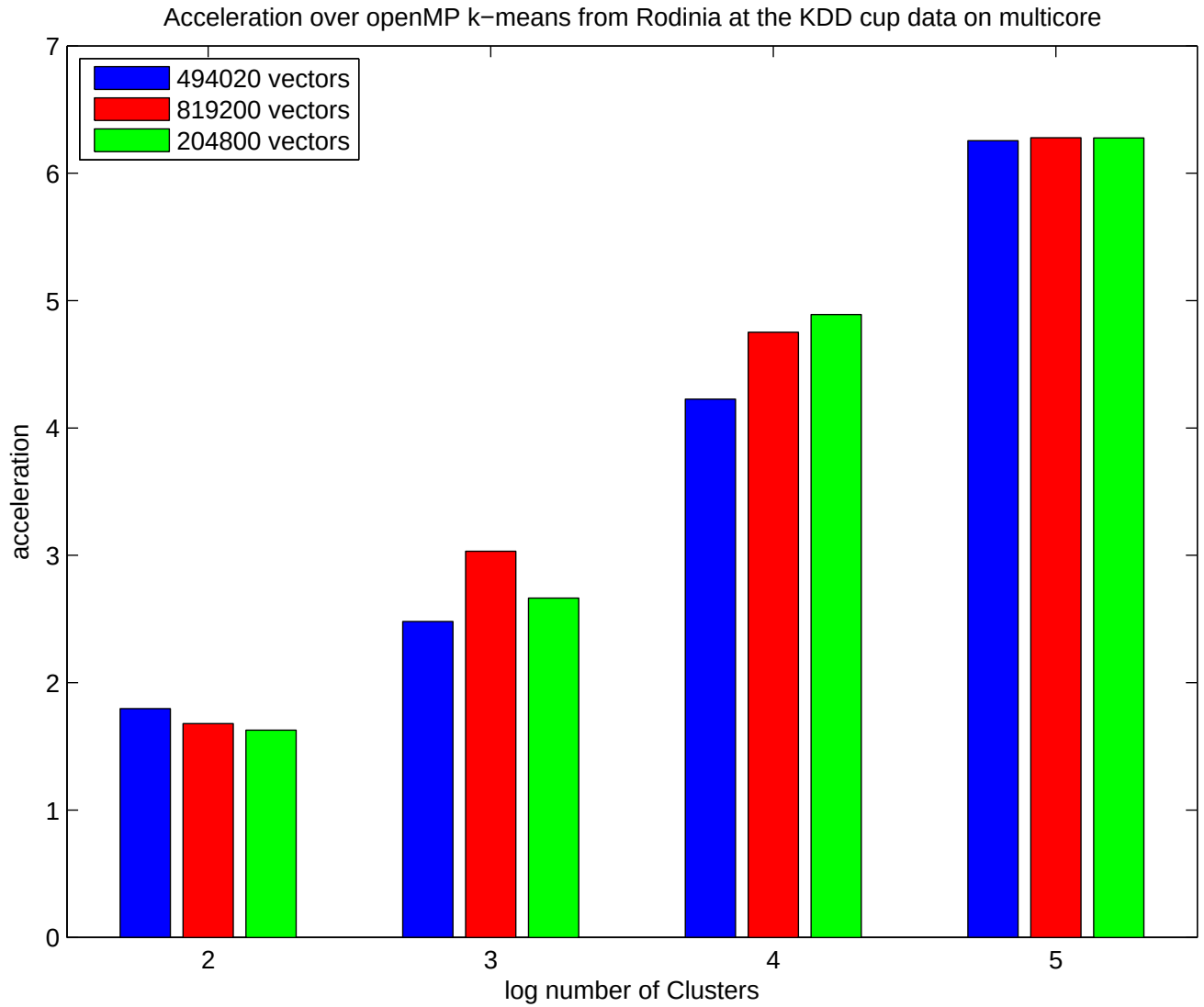




k-NNS CPU vs GPU on Caltech-101

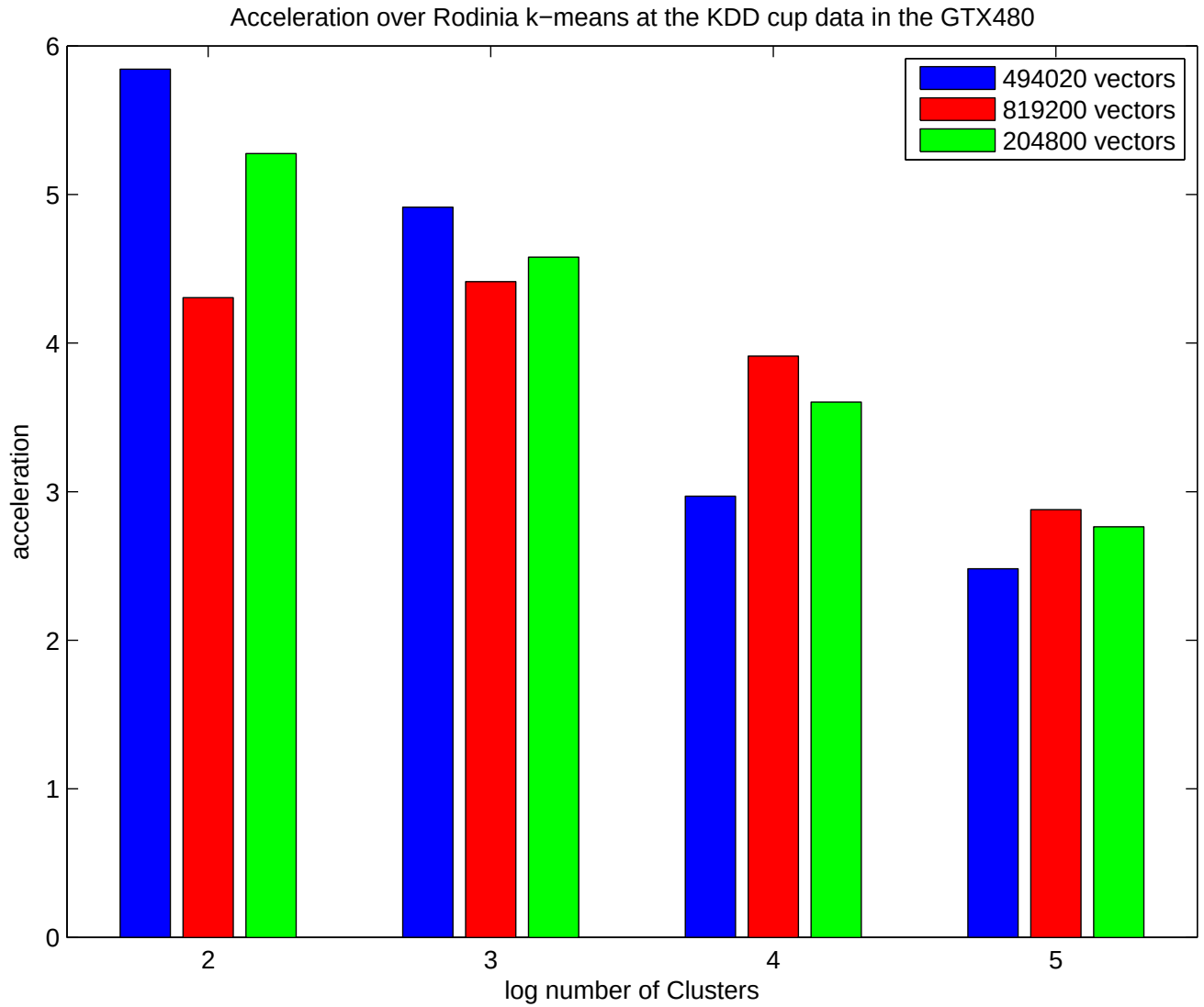


k-MC CPU (Rodinia) vs GPU on KDD-Cup



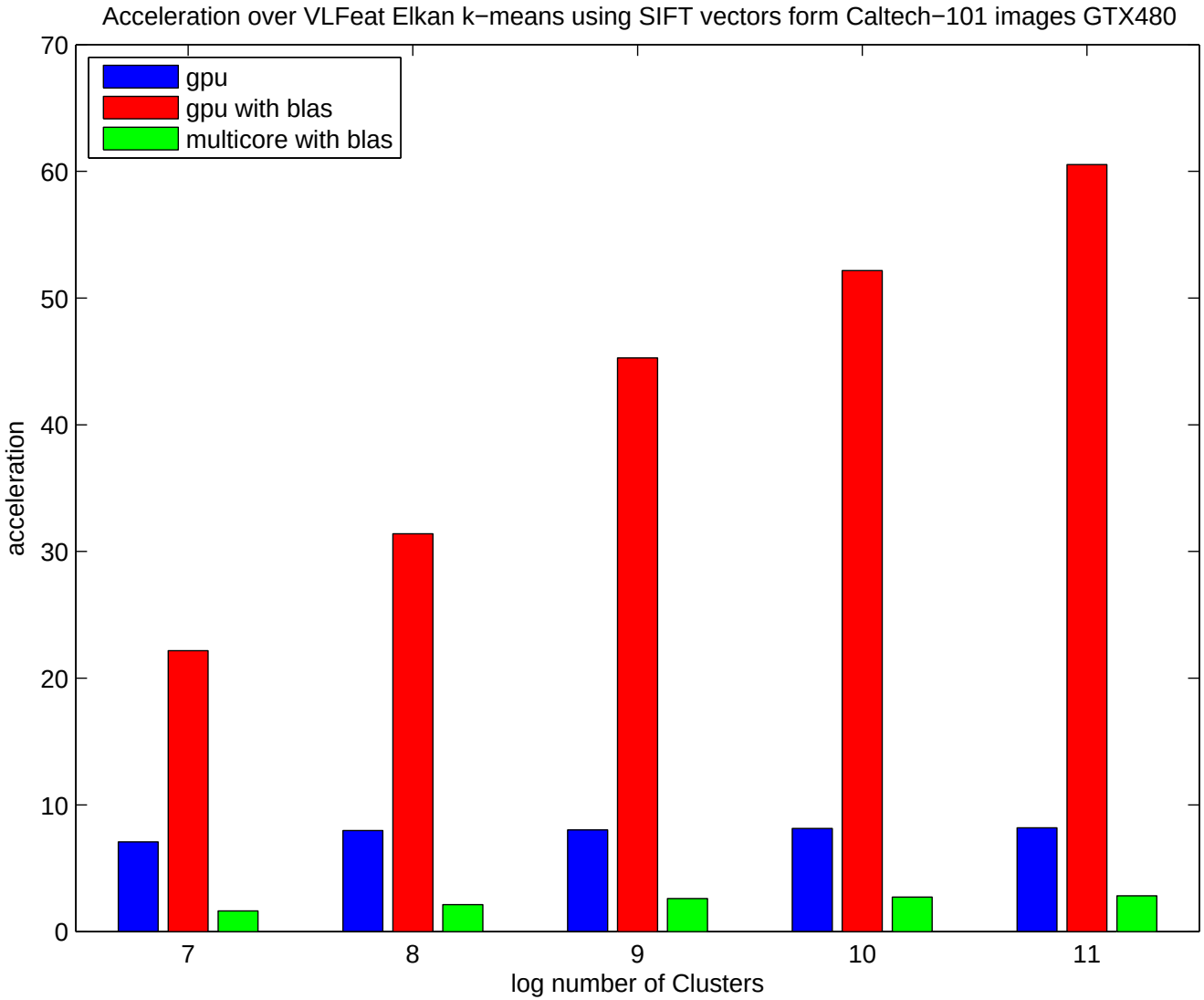
$d = 34$

k-MC Rodinia KDD-Cup



$d = 34$

VLFeat *k*-MC on Caltech-101



Acknowledgements

- The AUTh AutoGPU Project is supported by a Marie Curie International Reintegration Grant
- Duke University and NSF supported the stay of Nikos Sismanis at Duke in July 2011
- Profs G. Petridis and A. Delopoulos provided motivation for this work

- Codes and data available from `http://autogpu.ee.auth.gr`