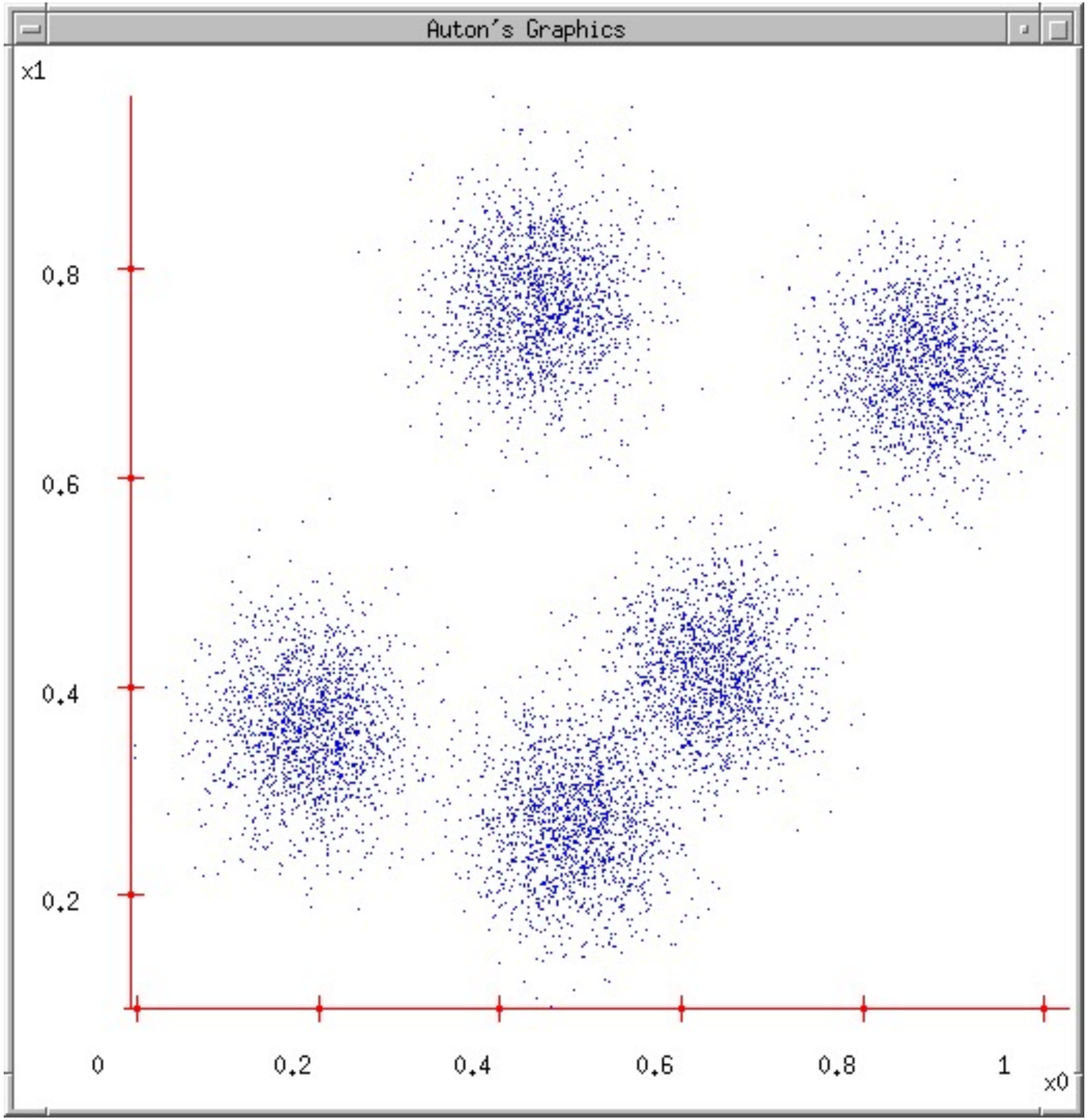
K-means

Note to other teachers and users of these slides. Andrew would be delighted if you found this source material useful in giving your own lectures. Feel free to use these slides verbatim, or to modify them to fit your own needs. PowerPoint originals are available. If you make use of a significant portion of these slides in your own lecture, please include this message, or the following link to the source repository of Andrew's tutorials: http://www.cs.cmu.edu/~awm/tutorials . Comments and corrections gratefully received.

Andrew W. Moore
Professor
School of Computer Science
Carnegie Mellon University
www.cs.cmu.edu/~awm
awm@cs.cmu.edu
412-268-7599

Nov 16th, 2001

Some Data

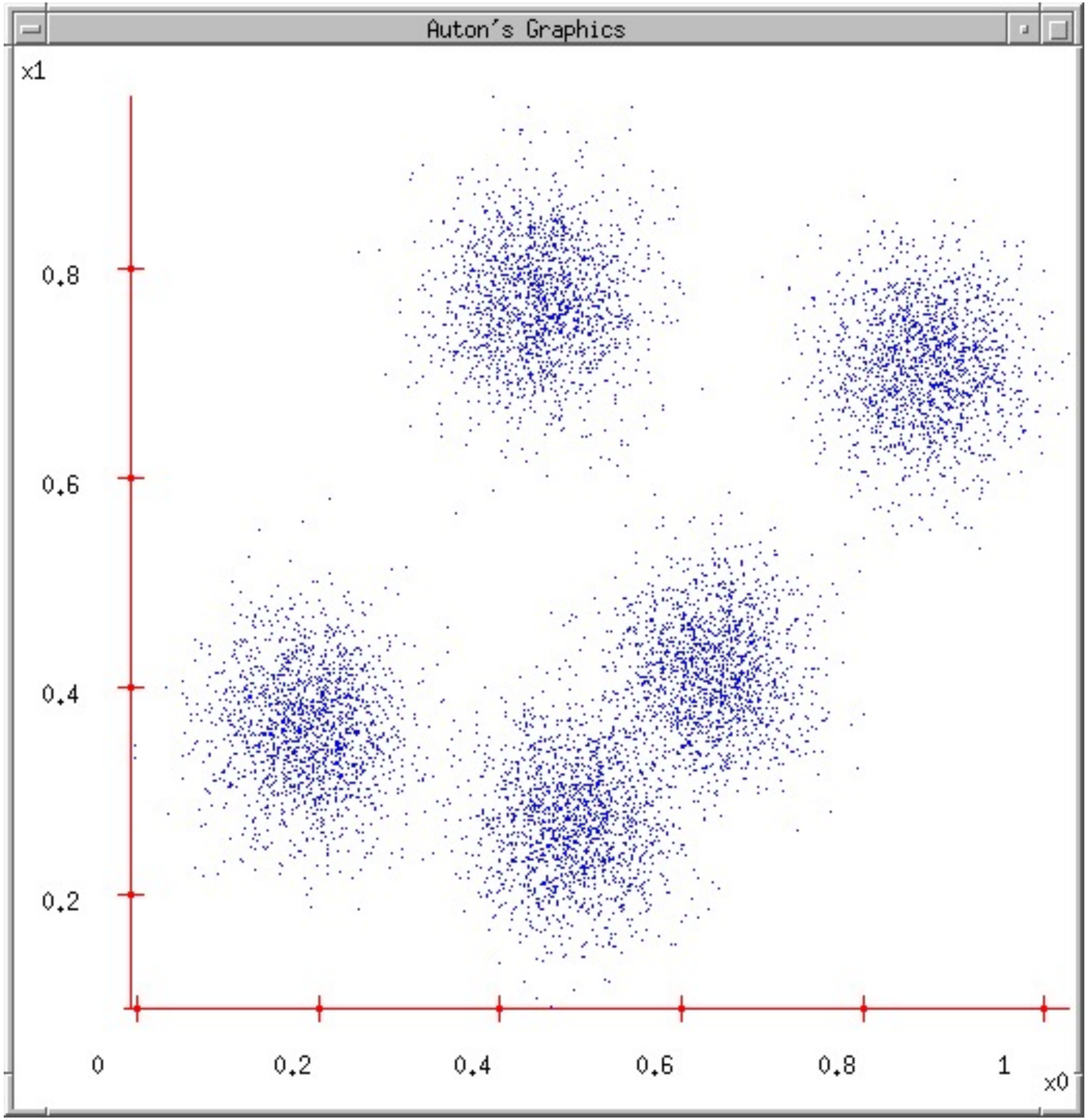


K-means clustering

- Input: K, set of points
- Place centroids c_1, c_2, …, c_k at random locations (alternatively add random lables to points)
- Repeat until convergences:
  - for each cluster j=1…K:
    - new centroid c_j = mean of all points P_i assigned to cluster j
  - for each point P_i:
    - find nearest centrod c_j
    - assing point P_i to cluster j
- Stop when none of the cluster assignments change

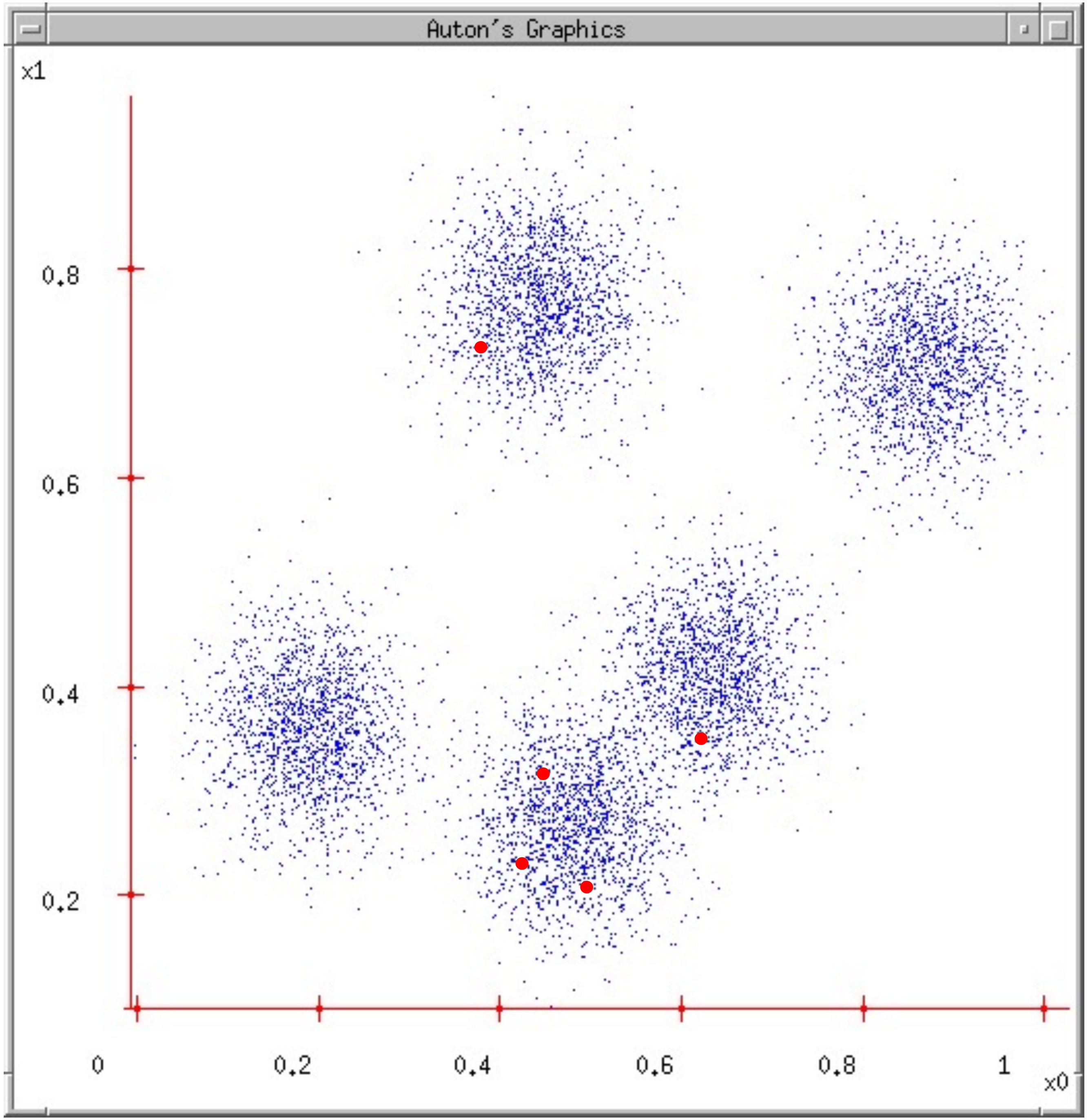
K-means

1. Ask user how many clusters they’d like. *(e.g. k=5)*



K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations

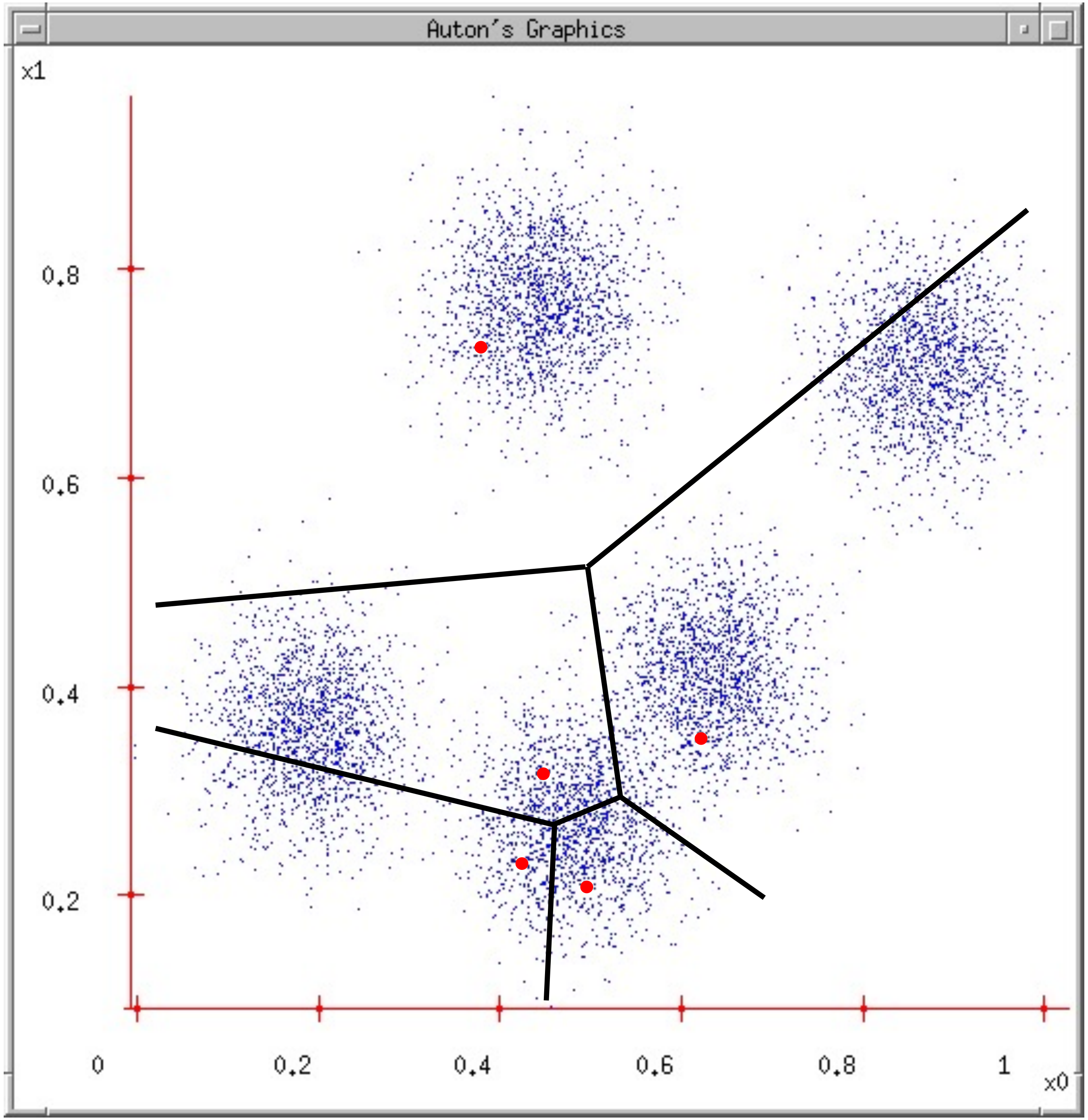


K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*

2. Randomly guess k cluster Center locations

3. Each datapoint finds out which Center it's closest to. (Thus each Center "owns" a set of datapoints)



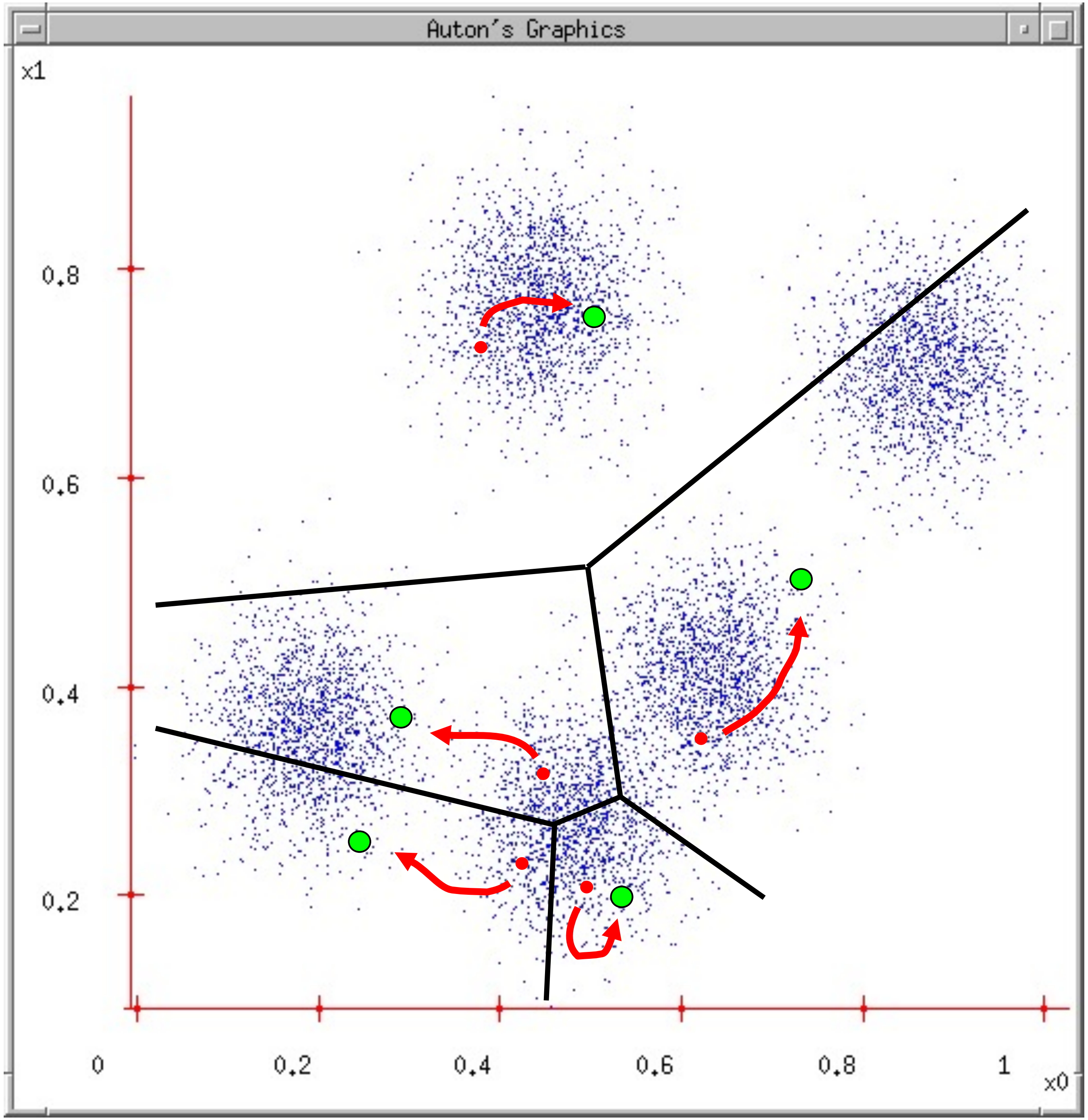
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*

2. Randomly guess k cluster Center locations

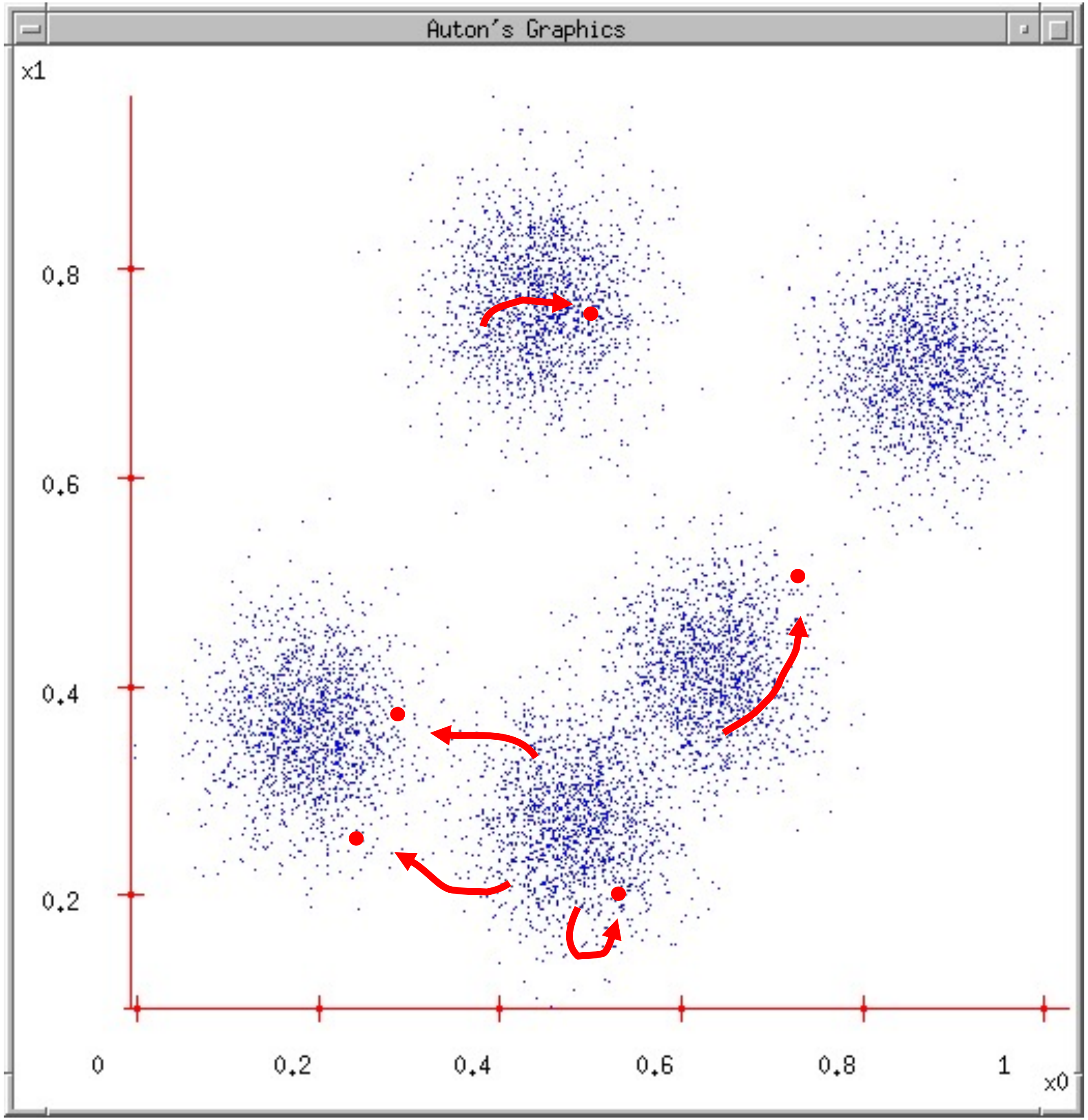
3. Each datapoint finds out which Center it's closest to.

4. Each Center finds the centroid of the points it owns

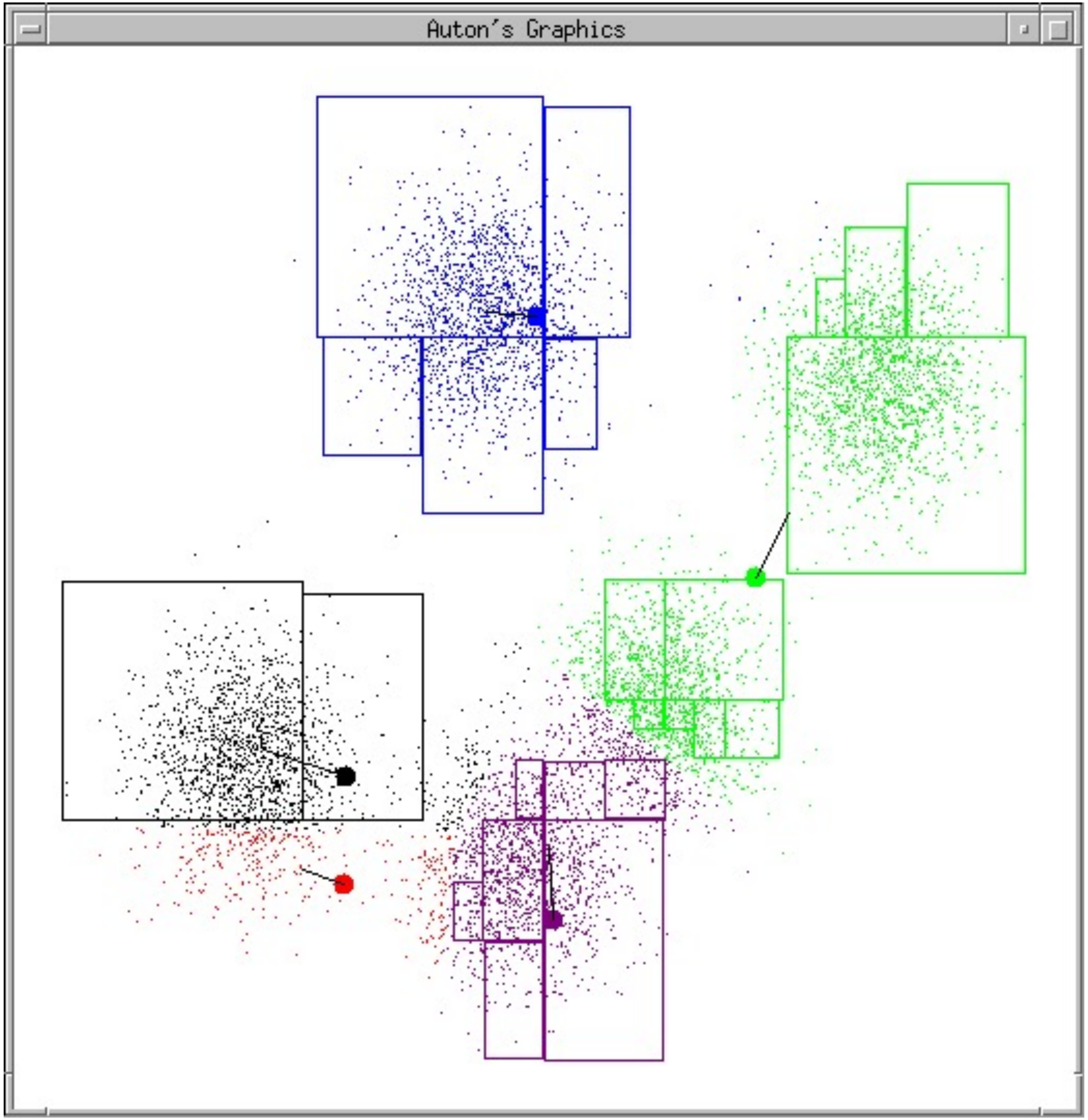


K-means

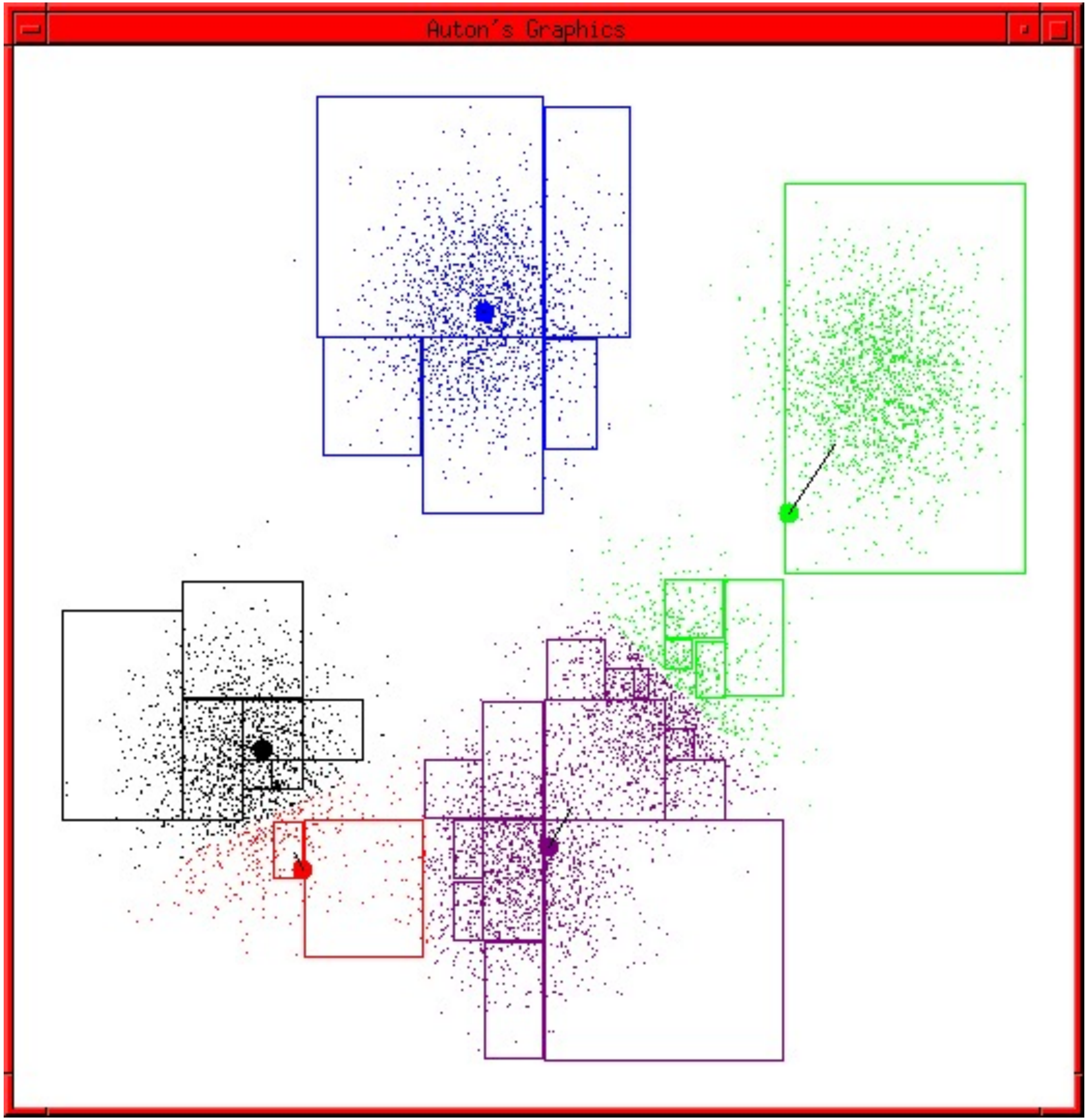
1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center finds the centroid of the points it owns…
5. …and jumps there
6. …Repeat until terminated!



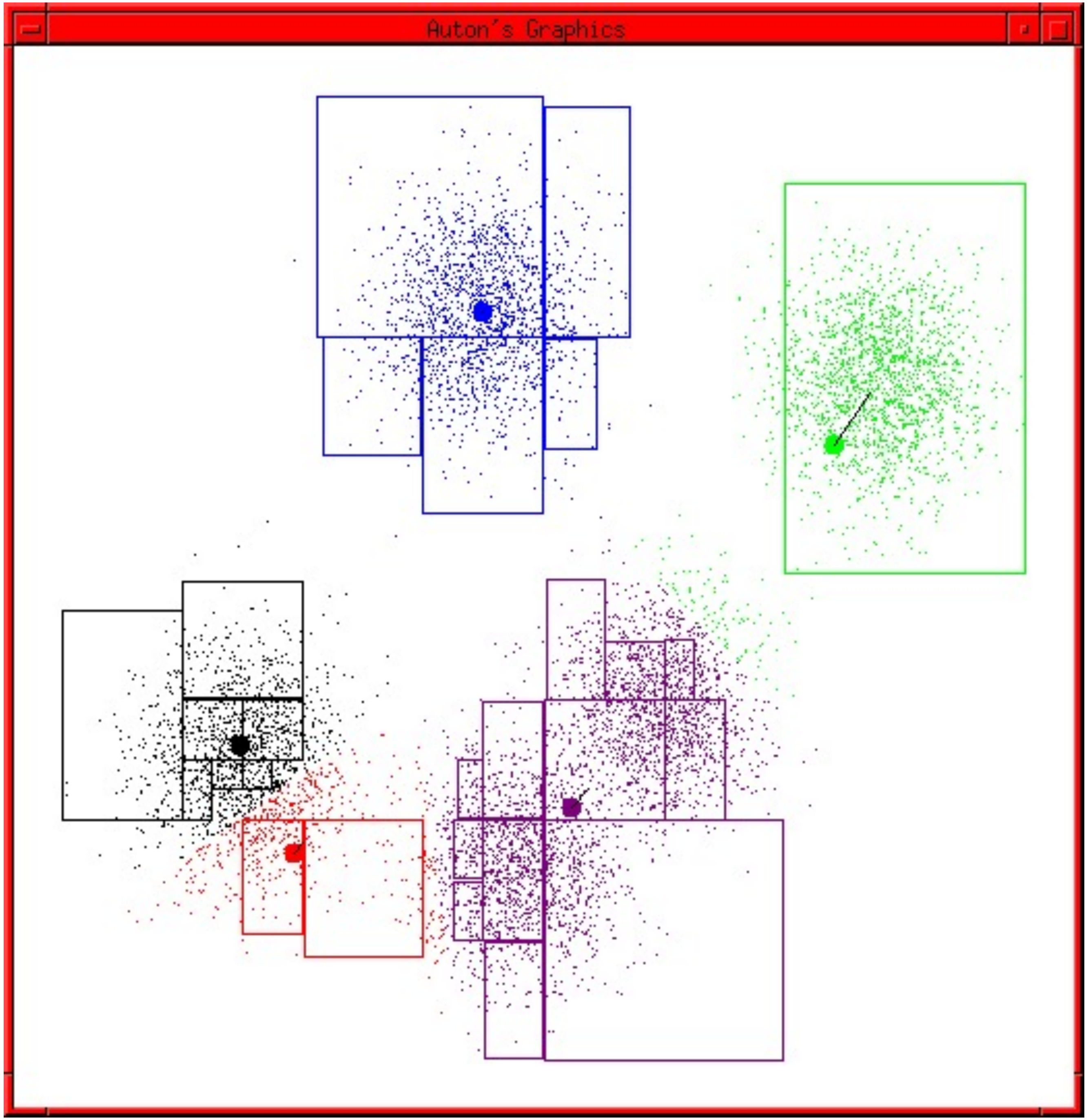
K-means continues …



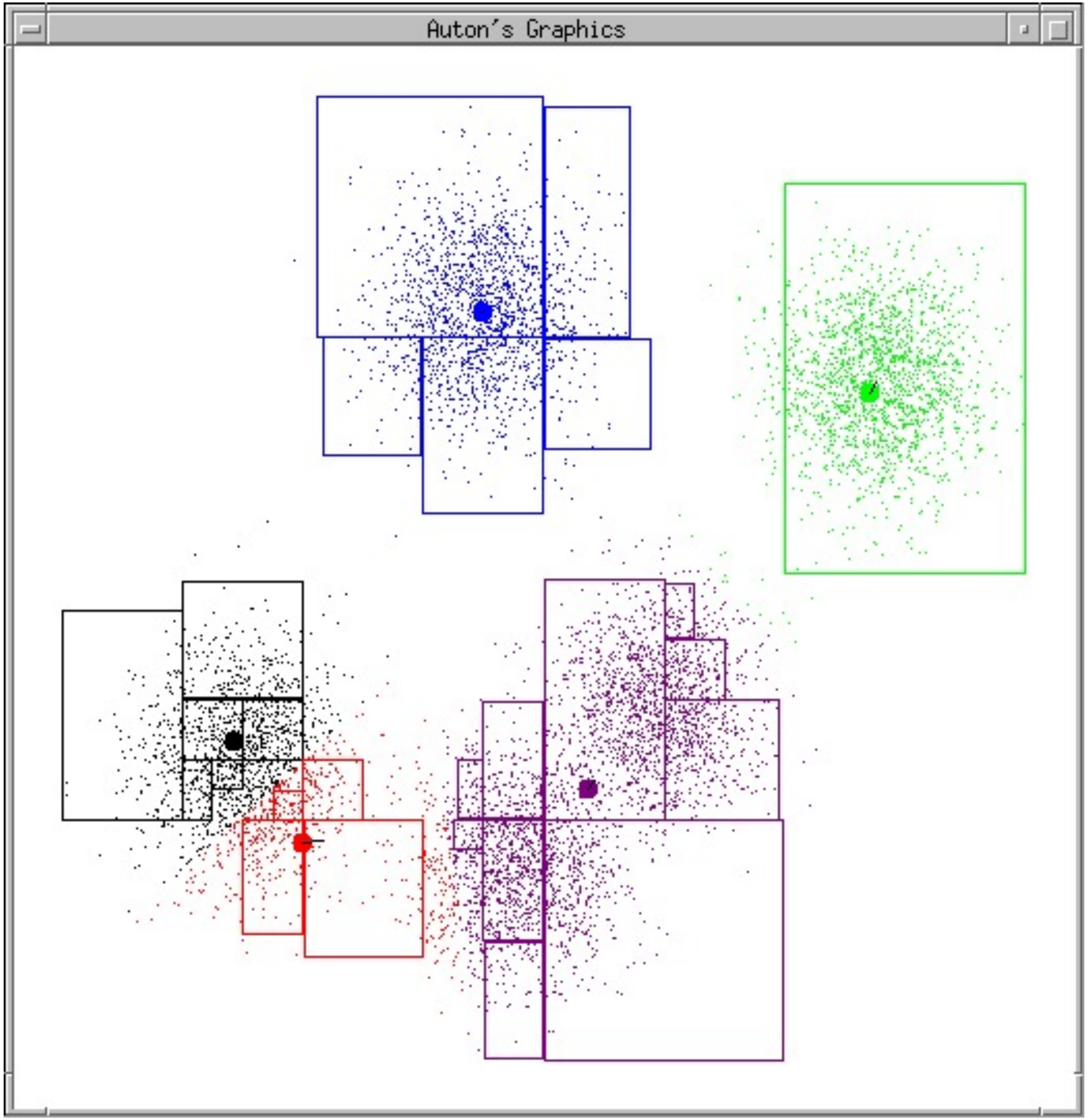
K-means continues …



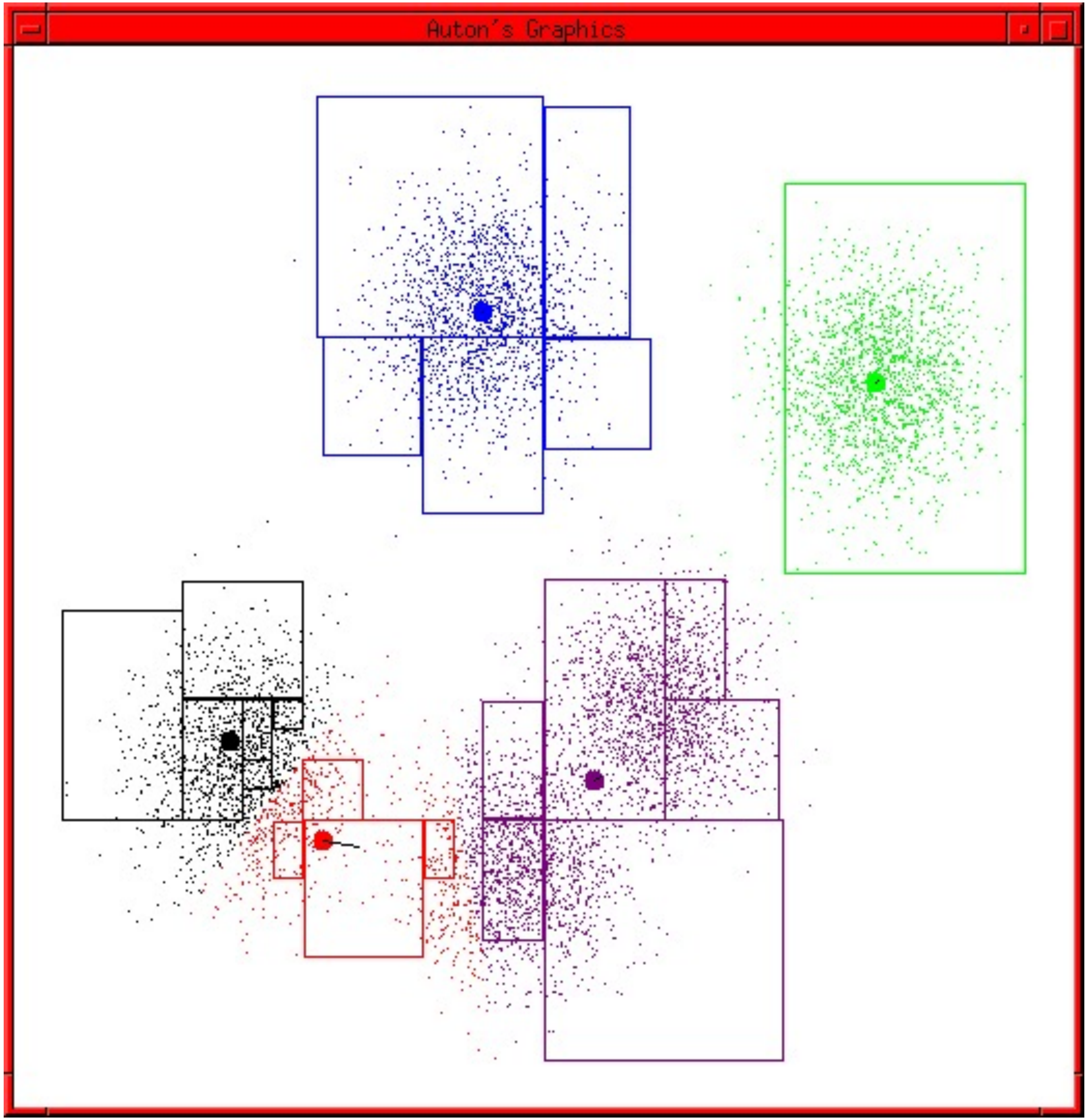
K-means continues …



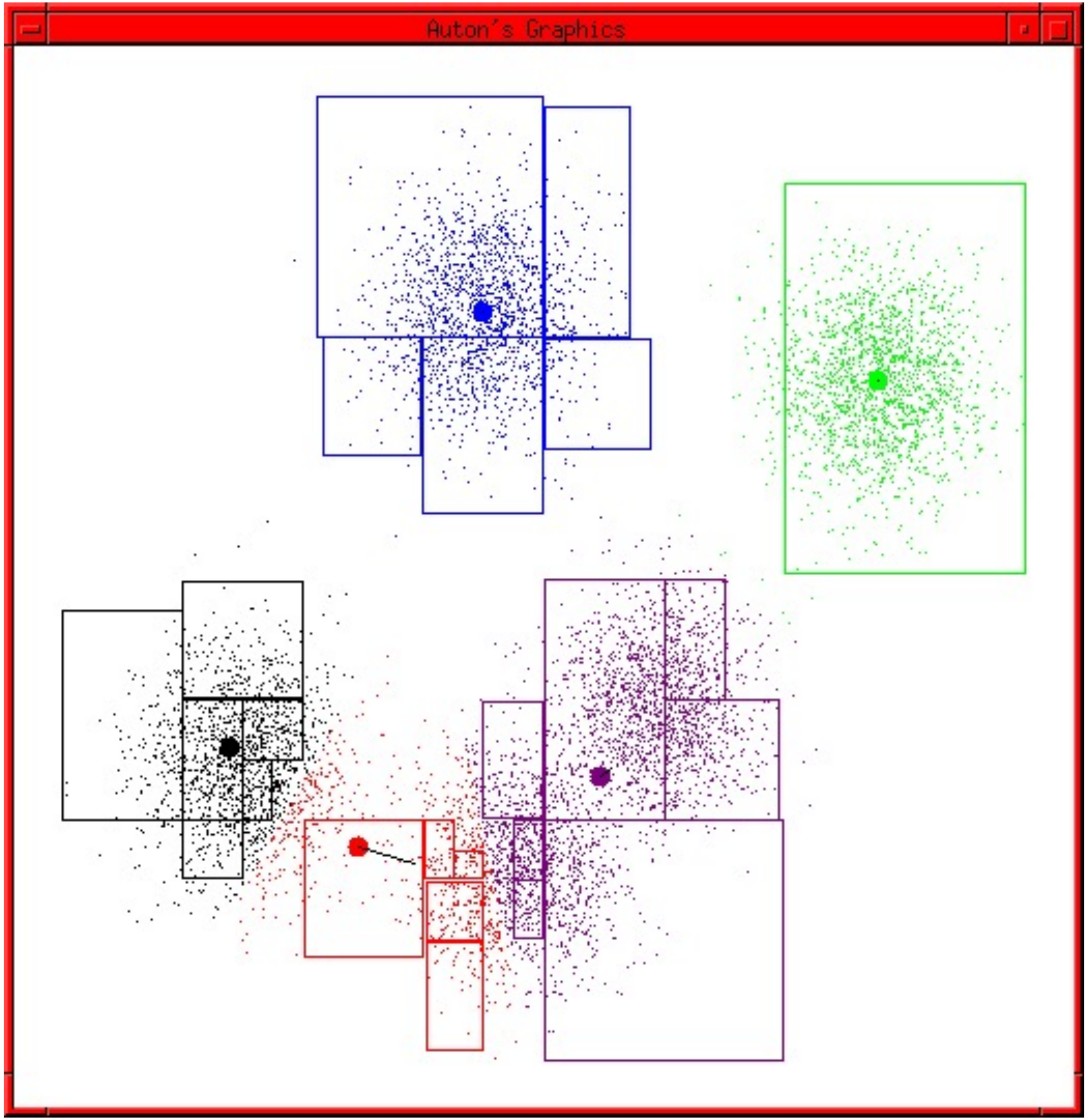
K-means continues …



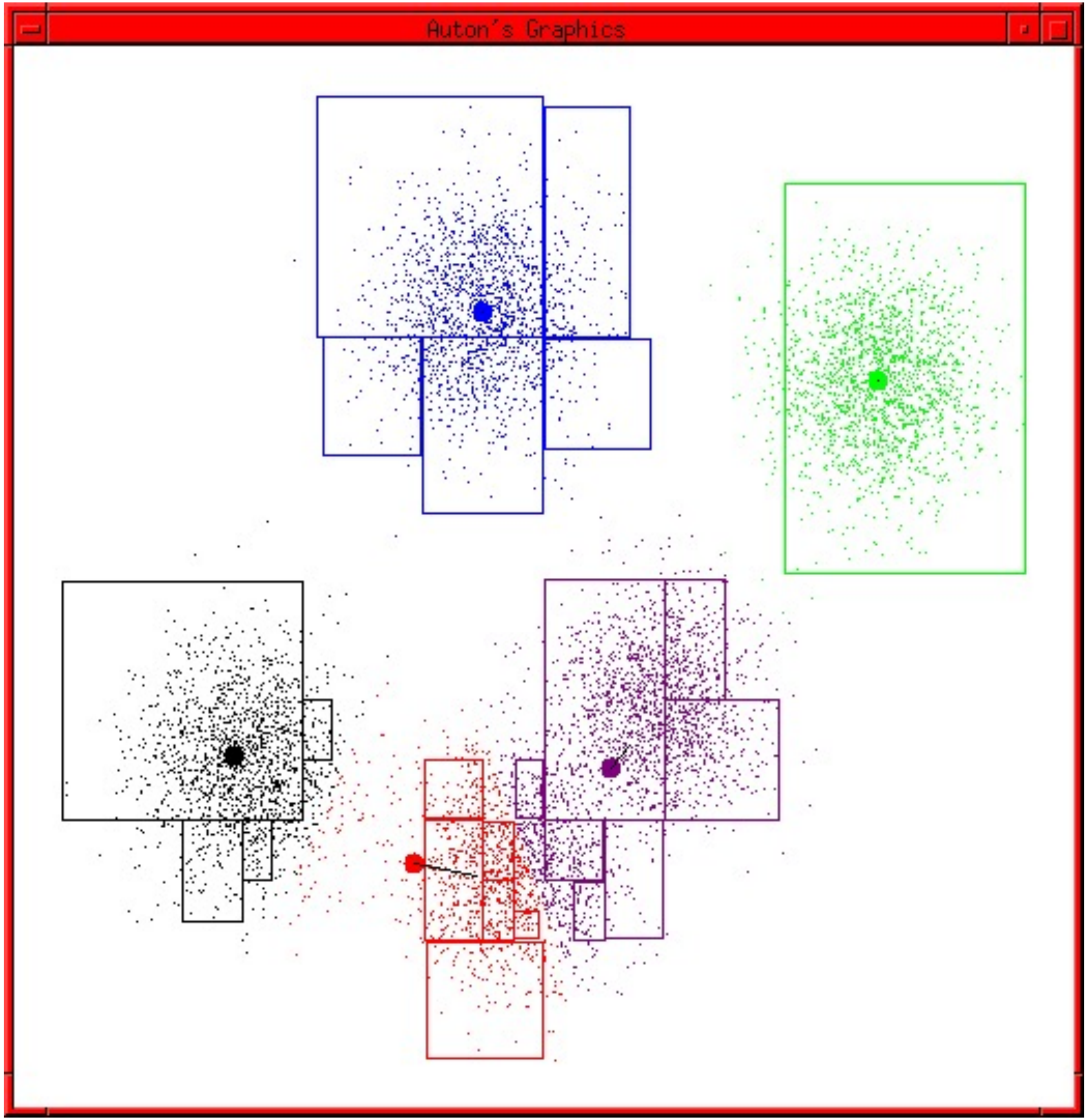
K-means continues …



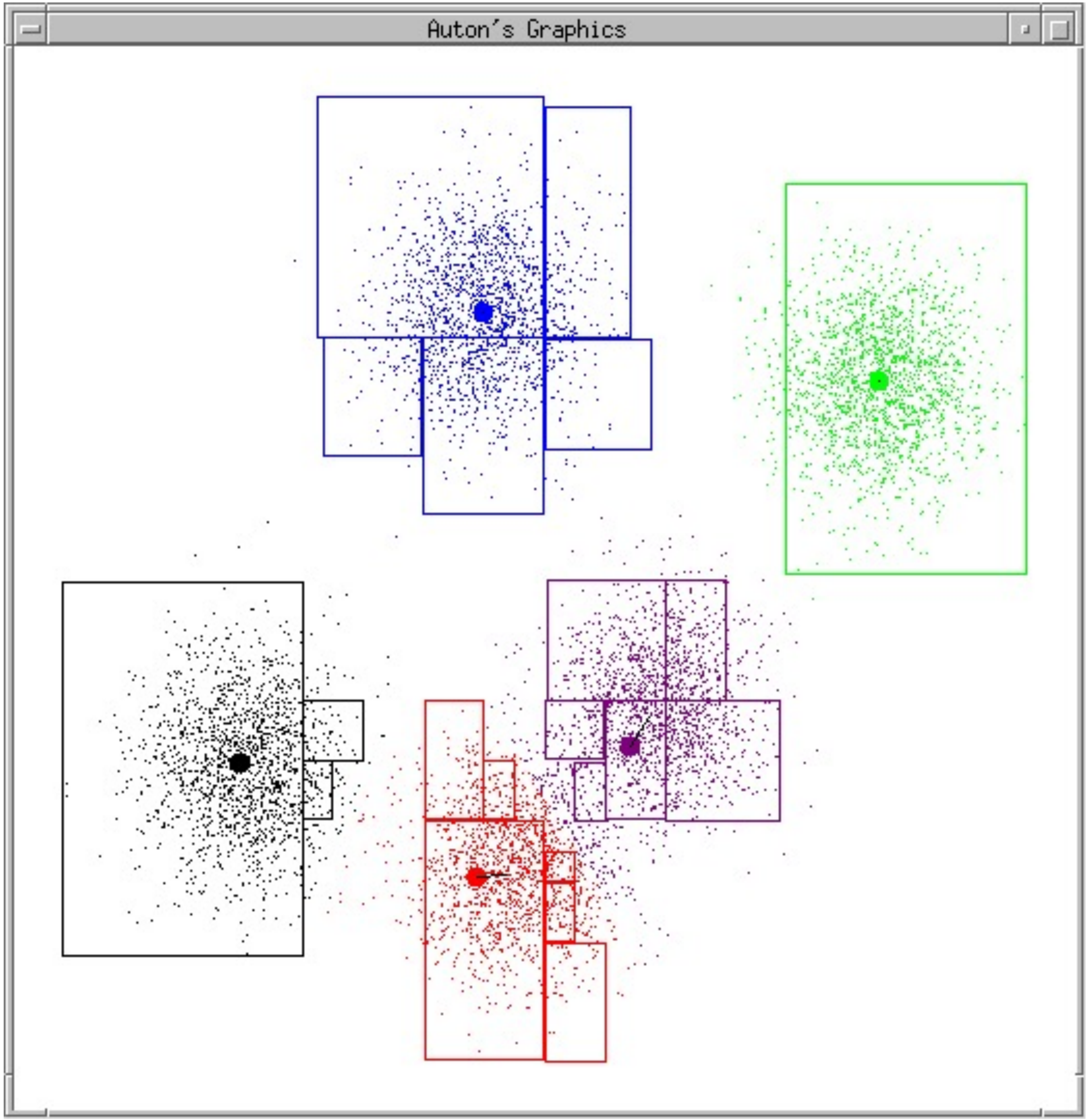
K-means continues …



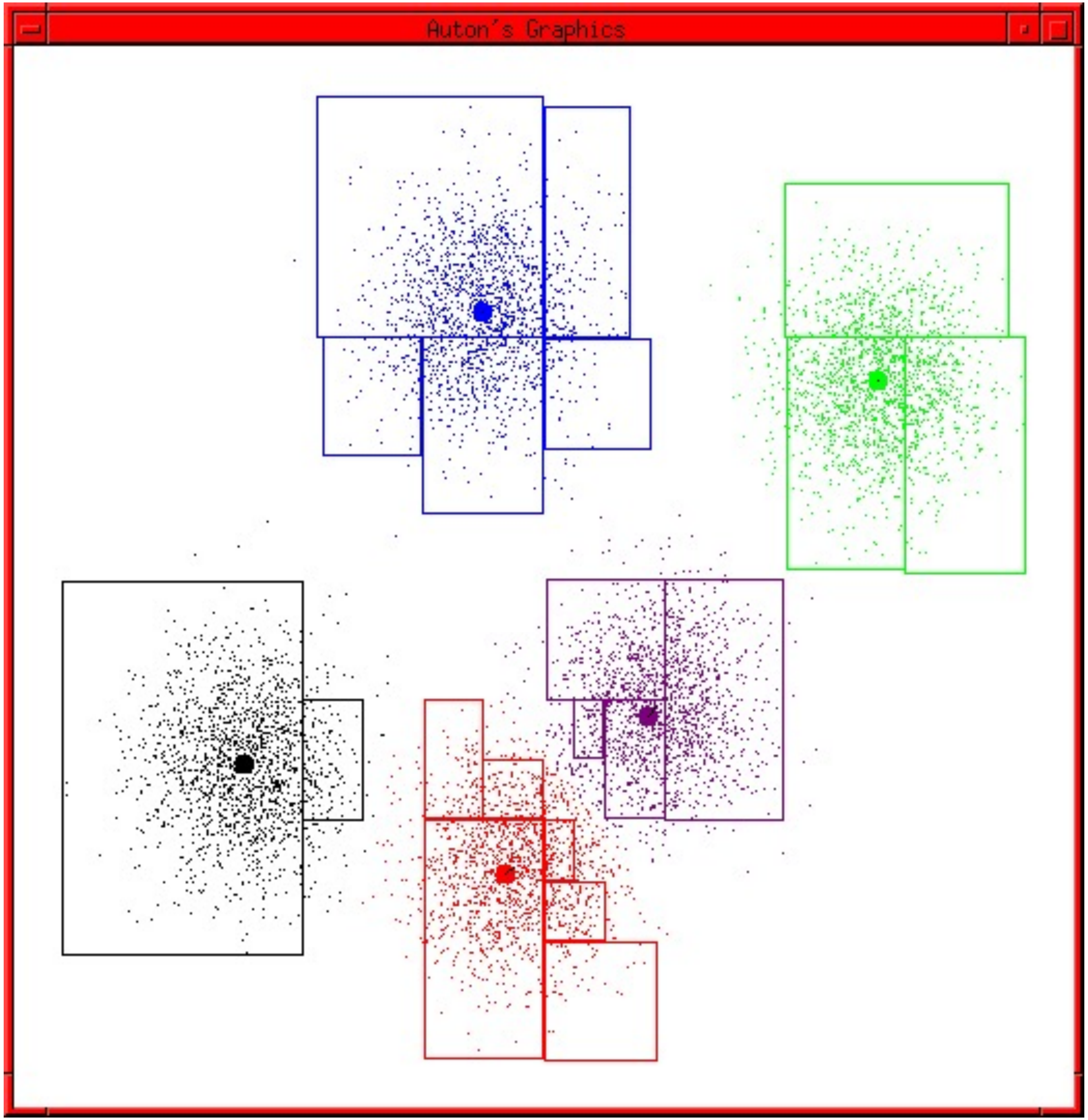
K-means continues …



K-means continues …

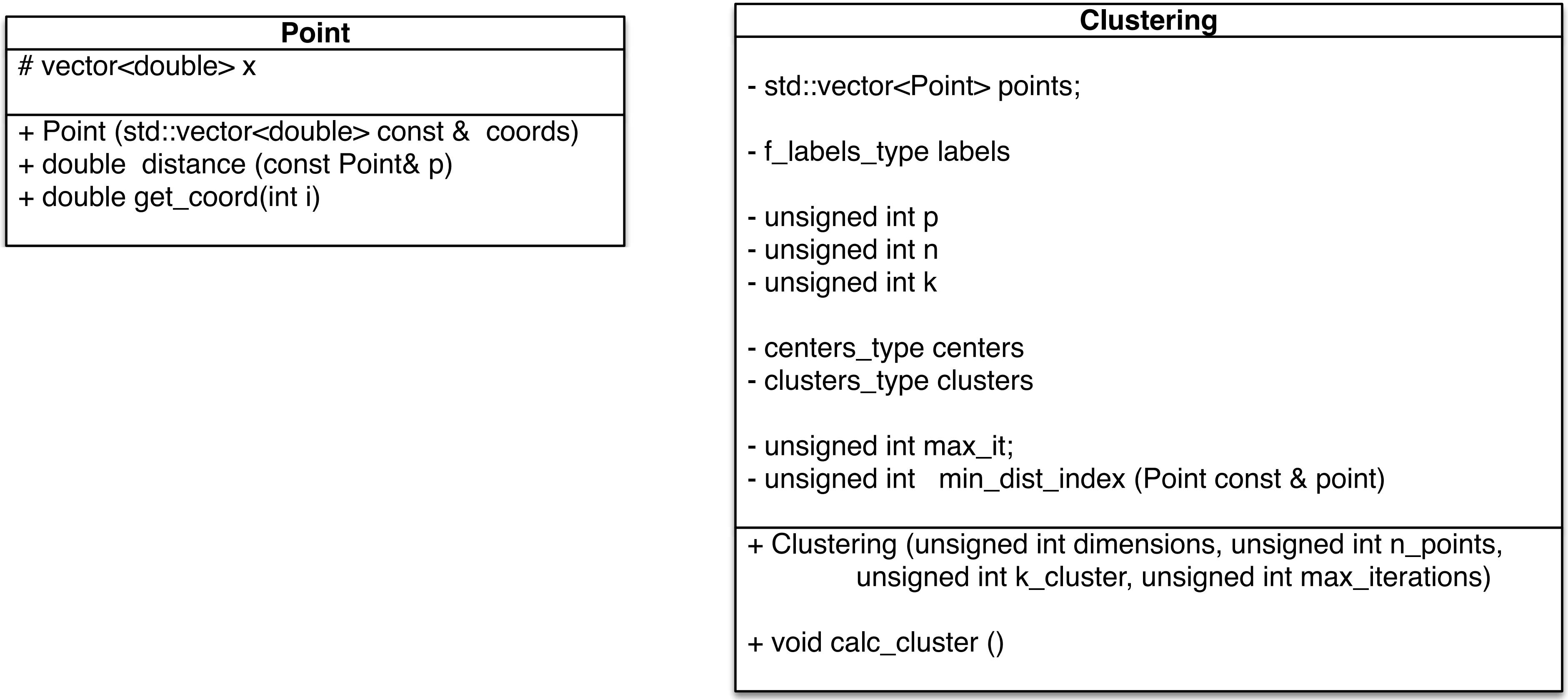


K-means terminates



K-means clustering

- Input: K, set of points
- Place centroids c_1, c_2, …, c_k at random locations (alternatively add random lables to points)
- Repeat until convergences:
  - for each cluster j=1…K:
    - new centroid c_j = mean of all points P_i assigned to cluster j
  - for each point P_i:
    - find nearest centrod c_j
    - assing point P_i to cluster j
- Stop when none of the cluster assignments change



Point

```
# vector<double> x

+ Point (std::vector<double> const &  coords)
+ double  distance (const Point& p)
+ double get_coord(int i)
```

Clustering

```
- std::vector<Point> points;

- f_labels_type labels

- unsigned int p
- unsigned int n
- unsigned int k

- centers_type centers
- clusters_type clusters

- unsigned int max_it;
- unsigned int   min_dist_index (Point const & point)

+ Clustering (unsigned int dimensions, unsigned int n_points,
            unsigned int k_cluster, unsigned int max_iterations)

+ void calc_cluster ()
```