A Branch and Bound Algorithm for Finding the Modes in Kernel Density Estimates

Oliver Wirjadi
Fraunhofer ITWM, Image Processing Group
67663 Kaiserslautern, Germany
wirjadi@itwm.fraunhofer.de

Thomas Breuel
University of Kaiserslautern, Department of Computer Science
67663 Kaiserslautern, Germany

Preprint of an article accepted for publication in
Int. J. Computational Intelligence and Applications

Abstract

Kernel density estimators are established tools in non-parametric statistics. Due to their flexibility and ease of use, these methods are popular in computer vision and pattern recognition for tasks such as object tracking in video or image segmentation. The most frequently used algorithm for finding the modes in such densities (the mean shift) is a gradient ascent rule, which can converge to local optima. We propose a novel, globally optimal branch and bound algorithm for finding the modes in kernel densities. We show in experiments on datasets up to dimension five that the branch and bound method is faster than local optimization and observe linear scaling of our method with sample size. Quantitative experiments on simulated data show that the new method gives statistically significantly more accurate solutions than the mean shift algorithm. The mode localization accuracy is about 5 times more precise than that of the mean shift for all tested parameters. Applications to color image segmentation on an established benchmark test set also show measurably improved results when using global optimization.

1 Introduction

This paper describes a globally optimal algorithm for finding the modes in kernel density estimates. These modes are of interest in many pattern recognition and computer vision applications, because they contain important information about the underlying random processes. Such problems include tracking of objects in color[12] or scale space[8], texture classification in feature spaces obtained from filter banks[15], and color image segmentation in joint coordinate and color space[9, 11].

1.1 Related Literature

A widely used algorithm for finding the local modes in kernel densities is the mean shift algorithm, which is a gradient-based approach: Using kernel densities, the mean shift is derived as the gradient of the smoothed density with respect to features. It is a theoretically well studied and commonly used method for the detection of maxima in kernel densities [7, 10, 11, 13]. Various modifications of this method have been proposed to avoid convergence in local optima. E.g., an annealing scheme to guide the mean shift to the global mode of a distribution [21] or the use of priors to force the mean shift path into a certain direction [19]. The use of the fast Gauss transform for mean shift was described by Yang et al.[24], who achieved linear scaling of their method when using this specialized algorithm.

The RAST (Recognition by Adaptive Subdivision of Transformation Space) algorithms solve the following problem: Given a set of model and observation points, find the optimal geometric transformation that best matches the two onto one another [6, 4, 5]. For solving this optimization problem, these algorithms employ a combination of interval arithmetic and branch and bound. The space of all possible transformations is subdivided into regions, and upper bounds for the quality of match are computed. By comparing the bounds of candidate regions in feature space, the method is guaranteed to converge to the global optimum. This strategy has been successfully applied to various computer vision problems[1, 17, 22].

An important tool for deriving and implementing RAST and the branch and bound algorithm presented in this paper is interval arithmetic[16]. As we will see later, it provides the means for computing bounds for the search algorithms. Interval arithmetic for global optimization is also discussed by Hansen and Walster[16], who deal with both constrained and unconstrained optimization.

Whereas the current paper transfers branch and bound from geometric matching problems to mode finding, Bandera et al. have recently proposed to use mean shift clustering for line finding in images[2].

1.2 Contribution of this Paper

This paper proposes the use of branch and bound algorithms for finding the modes of kernel densities. The basic idea and some preliminary results had previously been published[23]. The present paper contains a more thorough treatment of the theory and significantly extended experimental evaluations. Also, it is based on a completely new, more efficient C++ implementation, which is available from the authors (`http://www.iupr.org`).

2 Non-Parametric Density Estimation

All material in this paper addresses the following general question: Given $n > 0$ samples $\mathbf{x}_i \in \mathbb{R}^d$, $i = 1, \ldots, n$, from an unknown probability density function $p_{\mathbf{x}}$, how can we find the modes $\mathbf{x}_j^* \in \mathbb{R}^d$, $j = 1, \ldots, k$. Here, k denotes the number of modes of $p_{\mathbf{x}}$.

There are many ways for answering this question. For example, one could approximate $p_{\mathbf{x}}$ by a k-Gaussian mixture model. Then, the desired points $\mathbf{x}_j^*$ will be given by the mean vectors of the k model components. Instead of prescribing such a fixed model, non-parametric methods try to compute a density estimate $\hat{p}_{\mathbf{x}}$ directly from the data $\mathbf{x}_i$.

2.1 Kernel Densities

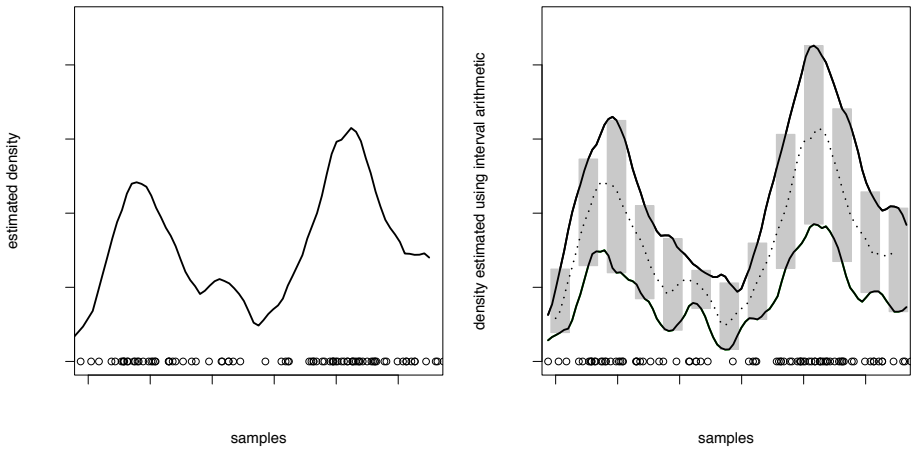
One such non-parametric method is *kernel density estimation*, where the density at point $\mathbf{x} \in \mathbb{R}^d$ is estimated as the outcome of a convolution of *kernel* K with the sample points.

$$\hat{p}(\mathbf{x}) = \frac{1}{nh^d} \sum_{i=1}^{n} K\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right), \tag{1}$$

where h denotes the bandwidth parameter of the kernel. This gives a non-parametric estimate $\hat{p}_{\mathbf{x}}$ of $p_{\mathbf{x}}$. Many kernel functions K_h may be used, e.g. Gaussian radial basis functions. Under the mean integrated square error (MISE) criterion, the optimal kernel is the *Epanechnikov Kernel* [20], given by

$$K_h\left(\mathbf{x} - \mathbf{x}_i\right) = \begin{cases} 0 & \|\mathbf{x} - \mathbf{x}_i\| < h \\ \frac{3}{4}\left(1 - \left(\frac{\|\mathbf{x}-\mathbf{x}_i\|}{h}\right)^2\right) & \text{else} \end{cases}. \tag{2}$$

This kernel function has finite support, i.e., it assumes a value of 0 for $\|\mathbf{x} - \mathbf{x}_i\| \geq h$. A one-dimensional example of a density estimated using this kernel function is shown in Fig. 1(a).



(a) A 1D kernel density with Epanechnikov kernel evaluated in conventional arithmetic.

(b) The same kernel density as in Fig. 1(a) with Epanechnikov kernel evaluated using interval arithmetic

Figure 1: Two ways of computing a kernel density: When using interval arithmetic, we can compute bounds for the value of $\hat{p}$ in each sample space interval (indicated by gray boxes).

2.2 Mean shift

The mean shift is a gradient ascent on the kernel density estimate given in (1). By taking the gradient of (1), setting it to zero and solving for $\mathbf{x}$, we obtain the iteration rule

$$\mathbf{x}^{t+1} = \frac{\sum_{i=1}^{n} \mathbf{x}_i K'_h(\mathbf{x}^t - \mathbf{x}_i)}{\sum_{i=1}^{n} K'_h(\mathbf{x}^t - \mathbf{x}_i)}, \tag{3}$$

in which K'_h denotes the derivative of K with bandwidth h. The Epanechnikov kernel in Eq. (2) is not differentiable at the boundary $\|\mathbf{x}-\mathbf{x}_i\| = h$. To show how it can be used in the mean shift iteration requires the introduction of "kernel profiles"[11], which is beyond the scope of this paper. The iteration in (3) is known to converge to the local stationary points of $\hat{p}$ [11].

2.3 Interval Arithmetic

Interval arithmetic (or interval analysis) was developed to allow for numerical computations with guaranteed results[16]. This paper uses interval arithmetic as a tool for optimization. We will denote intervals as variables with a "[]" in superscript, and we use half-open intervals in this paper. As an example, consider the addition of two intervals $a^{[]}$ and $b^{[]}$, which is given by

$$a^{[]} + b^{[]} = [\underline{a}, \bar{a}[+[\underline{b}, \bar{b}[= [\underline{a} + \underline{b}, \bar{a} + \bar{b}[. \tag{4}$$

Not all operations in interval arithmetic are as straight forward as addition (e.g., consider the multiplication of intervals that may contain the zero). Nevertheless, any operation $\circ$ can be extended to intervals by the min–max–rule,

$$a^{[]} \circ b^{[]} = [\min\left\{\underline{a} \circ \underline{b}, \bar{a} \circ \bar{b}, \underline{a} \circ \bar{b}, \bar{a} \circ \underline{b}\right\}, \max\left\{\underline{a} \circ \underline{b}, \bar{a} \circ \bar{b}, \underline{a} \circ \bar{b}, \bar{a} \circ \underline{b}\right\}[. \quad (5)$$

We now apply this mathematical tool to kernel density estimation: Consider the 1D example in Fig. 1(b). For each interval $\mathbf{x}^{[]}$, we can compute an upper and a lower bound for the value of the kernel density (indicated as gray boxes in this figure). For finding the modes of the kernel density, we would focus on intervals with large upper bounds. This idea leads to the branch and bound algorithm that will be described in Sec. 3.

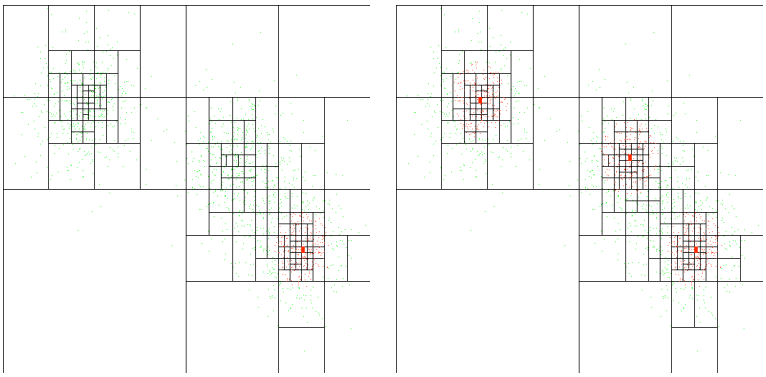
2.4 Explicit bounds for the Epanechnikov Kernel

For implementing an interval kernel density estimator, one could, in an object-oriented language, overload the usual arithmetic operators, obtaining an implementation of (1) with vectors replaced by d-dimensional boxes. Nevertheless, it is more efficient to explicitly compute and implement the computation of bounds, because it allows for some simplifications of the formulas.

Let $\mathbf{x}^{[]} = [\underline{\mathbf{x}}, \bar{\mathbf{x}}[$ be a d-dimensional real box, and the n samples $\mathbf{x}_i$ from above are wrapped into intervals $\mathbf{x}_i^{[]} := [\mathbf{x}_i, \mathbf{x}_i[$ with zero volume. The interval kernel density at $\mathbf{x}^{[]}$ using Epanechnikov kernel with positive width h, and dropping the constant factor $\frac{3}{4}$, can then be derived as follows.

$$\begin{aligned}
K_h^{[]}(\mathbf{x}^{[]} - \mathbf{x}_i^{[]}) &= \max\left\{[0,0[, [1,1[- \left(\frac{\|\mathbf{x}^{[]} - \mathbf{x}_i^{[]}\|}{h}\right)^2\right\} \\
&= \max\left\{[0,0[, [1,1[- \sum_{j=1}^{d}\left(\left[\frac{\underline{\mathbf{x}}^j - \mathbf{x}_i^j}{h}, \frac{\bar{\mathbf{x}}^j - \mathbf{x}_i^j}{h}\right[\right)^2\right\} \\
&= \left[\max\left\{0, 1 - \sum_{j=1}^{d}\max\left\{\left(\frac{\underline{\mathbf{x}}^j - \mathbf{x}_i^j}{h}\right)^2, \left(\frac{\bar{\mathbf{x}}^j - \mathbf{x}_i^j}{h}\right)^2\right\}\right\},\right. \\
&\qquad \left.\max\left\{0, 1 - \sum_{j=1}^{d}\min\left\{\left(\frac{\underline{\mathbf{x}}^j - \mathbf{x}_i^j}{h}\right)^2, \left(\frac{\bar{\mathbf{x}}^j - \mathbf{x}_i^j}{h}\right)^2\right\}\right\}\right[\quad (6)
\end{aligned}$$

The computational overhead of computing (6) instead of its conventional counterpart (2) consists of $2d+1$ comparisons and d subtractions, divisions and multiplications, each. In other words, evaluation of (1) is in $\mathcal{O}(dn)$ regardless of whether we use interval or vector arithmetic.



(a) The first mode (b) The top 3 modes

Figure 2: Examples of applying the adaptive subdivision to some 2D data. Data samples are show as green points. Red points indicate samples that had a non-zero contribution to the kernel density at the final intervals ("active samples").

3 Branch and Bound for Mode Finding

With the tools from the previous section at hand, we can now state the central algorithm of this paper. It is a branch and bound algorithm that uses the upper bounds of kernel densities for adaptively subdividing the sample space until arriving at a d-dimensional interval which is sufficiently small for being accepted as solution (measured by a parameter $\epsilon > 0$).

1. Let $\mathbf{x}_1^{[]}$ be the d–dimensional interval of all possible mode locations, i.e., $\forall i = 1, \ldots, n : \mathbf{x}_i \in \mathbf{x}_1^{[]}$. Enqueue it to a priority queue with priority $+\infty$
2. Remove the top ranking element $\mathbf{x}_t^{[]}$ from the queue. If $|\mathbf{x}_t^{[]}|_\infty < \epsilon$, accept $\mathbf{x}_t^{[]}$ as solution and finish.
3. Split $\mathbf{x}_t^{[]}$ into two new sub-intervals $\mathbf{x}_{2t}^{[]}$ and $\mathbf{x}_{2t+1}^{[]}$ by dividing $\mathbf{x}_t^{[]}$ in the dimension where it is widest. Enqueue these new intervals using the upper bounds of their kernel density values $\hat{p}(\mathbf{x}_{2t}^{[]})$ and $\hat{p}(\mathbf{x}_{2t+1}^{[]})$ (computed using Eq. (6)).
4. Continue at step (2).

This algorithm returns an interval containing the mode of the kernel density $\hat{p}(\mathbf{x})$, and is strongly related to a similar algorithm that has been introduced for geometric matching, see below. Fig. 2(a) shows a 2D example of the divisions generated by this algorithm and shows the interval $\mathbf{x}_*^{[]}$ (in red) to which it converged. We use the midpoint of the final interval as solution, i.e., $\mathbf{x}^* = \frac{1}{2}(\underline{\mathbf{x}}_* + \bar{\mathbf{x}}_*)$.

In order to efficiently implement this algorithm, each interval $\mathbf{x}_t^{[]}$ should keep track of the samples $\mathbf{x}_i$ that have a non–zero contribution to its kernel density. Call these the "active" samples of an interval. Because a child interval can not have any active samples that were inactive in its parent, this drastically reduces the number of required kernel evaluations.

3.1 Analysis of the Algorithm

Let $\mathbf{x}_*^{[]}$ be the final interval chosen in step (2). To guarantee that this interval indeed contains the optimal solution, we would need to check if there exists a point $\mathbf{x} \in \mathbf{x}_*^{[]}$ achieving the computed upper bound. This step is usually omitted for efficiency, resulting in what has been termed a weak geometric algorithm[5].

The algorithm given above constructs a d-dimensional kd-tree, not on the sample points $\mathbf{x}_i$, but rather by dividing the search space into boxes until arriving at a sufficiently tight interval. Let m be the number of nodes in the final tree. Construction of the tree structure can then be done in $\mathcal{O}(m \log m \cdot dn)$ time[14], where $\mathcal{O}(dn)$ is the time required for evaluating (1), cf. Sec. 2.4. Because at each level $l > 0$ the tree consists of at most 2^l nodes and because the width of the intervals is inversely proportional to $2^l/d$, the worst case depth of the search tree is in $\mathcal{O}\left(\log(d\epsilon^{-1})\right)$.

3.2 Relation to Geometric Matching

A slightly different variant of the above branch and bound algorithm is known as *RAST algorithm*, which has successfully been applied to various geometric matching problems [1, 4, 5, 17]. RAST algorithms have been developed for finding an optimal geometric transformation T from transformation space $\mathcal{T}$ under bounded error. Suppose we are given two sets $\{\mathbf{y}_j\}_{j=1,\ldots,p}$ and $\{\mathbf{z}_k\}_{k=1,\ldots,q}$ of points in $\mathbb{R}^d$. The optimal transformation T^* is the one that maps exactly one $\mathbf{y}$ onto one $\mathbf{z}$. Such a perfect match does usually not exist due to noise or because of some transformations that are not contained in the search space $\mathcal{T}$. To measure the quality of such matches, RAST algorithms use a quality measure Q,

$$Q(T) = \sum_{j=1}^{p} \max_k q\left(T \cdot \mathbf{y}_j - \mathbf{z}_k\right). \tag{7}$$

This means that these algorithms are searching for a transformation $T^* = \operatorname{argmax}_{T \in \mathcal{T}} Q(T)$ that locally minimizes the error of match. The algorithm from Sec. 3 can be used to find an interval containing T^* when instead of evaluating the kernel density in step (3), the upper bound of the match quality Q is computed. One quality measure q that has been used for geometric matching[1, 22] is

$$q\left(t\right) = \max\left\{0, 1 - (t/h)^2\right\}, \tag{8}$$

which, up to scaling, is equal to the Epanechnikov kernel from Eq. (2). This form of q is used for finding maximum likelihood solutions under a Gaussian error model[6]. It neglects outliers, i.e., points that are farther than h from a transformed point $T \cdot \mathbf{y}$, while points that are close to a model point $\mathbf{z}$ have a large contribution to the quality of match ($q(0) = 1$).

3.3 Greedy Mode Seeking

Sometimes, not only the global, but also local modes of a distribution are of interest. E.g., mean shift mode seeking is often used for image segmentation by clustering color values in color spaces. Under the assumption that image regions form tight clusters in color space, we are then interested in finding not one, but many cluster.

The above procedure for finding global modes in kernel densities is easily extended to finding the k first modes as follows: Assume we have found an interval $\mathbf{x}_*^{[]}$ which was accepted as a solution. Remove this element from the queue and also remove its active sample points from all elements that are currently in the queue. Update the kernel values of all intervals, re-sort the priority queue, and continue as above. Stop when k kernel modes have been found. Fig. 2(b) shows a 2D example result of this extended algorithm with $k = 3$.

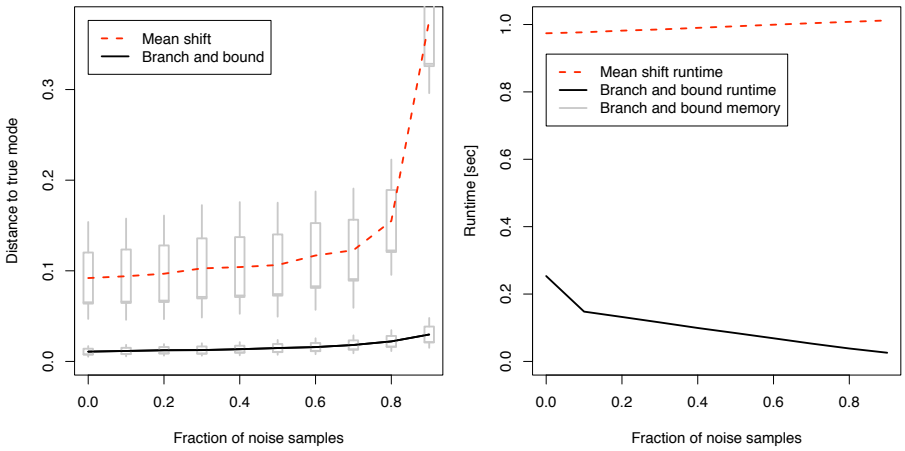
Note that the k modes of the kernel density $\hat{p}$ alone do not give a clustering, but just the locations of these modes in feature space. For the examples that we will show in Sec. 5, we have used the nearest neighbor rule to assign feature points to clusters.

4 Experimental Evaluation

We have claimed that branch and bound mode finding would be more accurate and more efficient than mean shift. This section presents numerical simulations to verify these claims. The experimental setup is similar to the one used by Georgescu et al. for evaluating the performance of their adaptive mean shift implementation[15].

First, to check the global mode finding performance, an *unimodal distribution* was used. The test data consisted of samples from a d-dimensional isotropic normal distribution with fixed standard deviation σ and uniformly distributed noise points covering the same interval as the true samples. A noise fraction quantifies the noise level: Say we set $n = 100$ and simulate a noise fraction of 0.5. By this we mean that we draw $0.5 \cdot 100 = 50$ samples from the normal and add another 50 noise points, thus completing the total of 100 points. A new uniformly distributed mean vector in $[-1, 1]^d$ was drawn in each trial.

Precision and runtime of our and the mean shift methods were evaluated with respect to each of these three parameters (dimension, sample size, noise) independently. The distance between a resulting vector and the true mode measures the precision.



(a) Branch and bound mode finding is less sensitive to noise than mean shift.

(b) Runtime of our method is significantly lower.

Figure 3: Experiments with samples from a 3-dimensional isotropic Gaussian ($\sigma = 0.1$) at different noise levels. The total number of points was fixed to 5000.

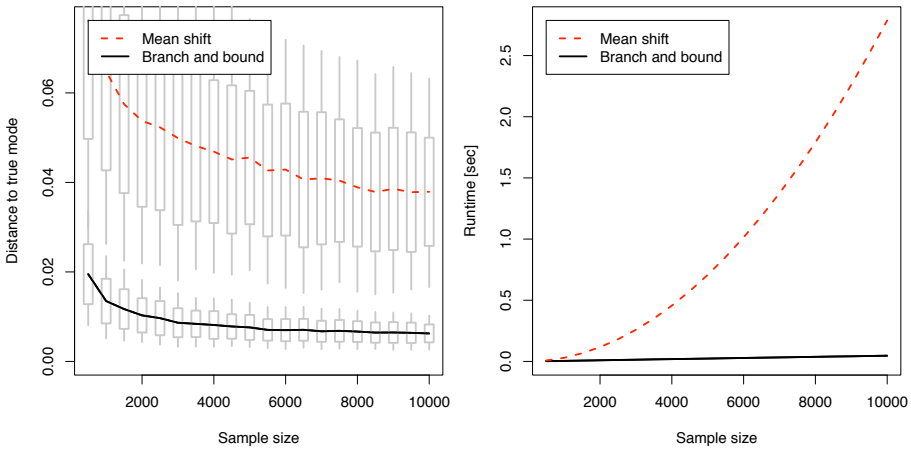
Second, for evaluating the performance of the greedy mode finding scheme for k modes, a *multimodal distribution* in uniform noise was used. The model for this k mode finding experiment was a k-Gaussian mixture with equal priors $1/k$ and identical standard deviation σ for each mixture component.

The performance measure here is the receiver-operator-characteristic (ROC): We counted true and false matches in terms of a distance threshold θ specifying the acceptable offset of a solution to a true mode. The Gaussian mixture EM algorithm approximates the maximum likelihood solution for this test data[3], and is therefore suitable as an additional control method.

In order to show the differences due to the algorithms themselves – rather than due to parameter tuning – our and the mean shift method always used equal parameters. That is, the bandwidth of the kernel always was $h = \sigma$ and the same ϵ determined termination of our algorithm and of the mean shift gradient ascent, which stopped when the change in one iteration fell below ϵ. All results in this section are averages over 1000 trials.

4.1 Samples from an unimodal distribution

Fig. 3, 4 and 5 show the results of the above described experiment on a unimodal Gaussian distribution. The branch and bound mode finding algorithm shows statistically significant improvements in mode localization accuracy over the mean shift method, with respect to all three evaluated parameters. Additionally, our algorithm is faster than the mean shift for feature space dimension up to four. Linear scaling of the new branch and bound algorithm with sample size is shown in Fig. 4(b).



(a) Localization of our method is significantly better and converges faster than the one of mean shift.

(b) In contrast to mean shift, branch and bound mode finding scales linearly with samples size.

Figure 4: Experiments with samples from a 3-dimensional isotropic Gaussian ($\sigma = 0.1$) at 25% noise level. The total number of points was increased from 500 to 10000.

The peak size in memory of the branch and bound priority queue during search was 700kB for the 10000 3-dimensional samples and 20MB for 5000 5-dimensional points.

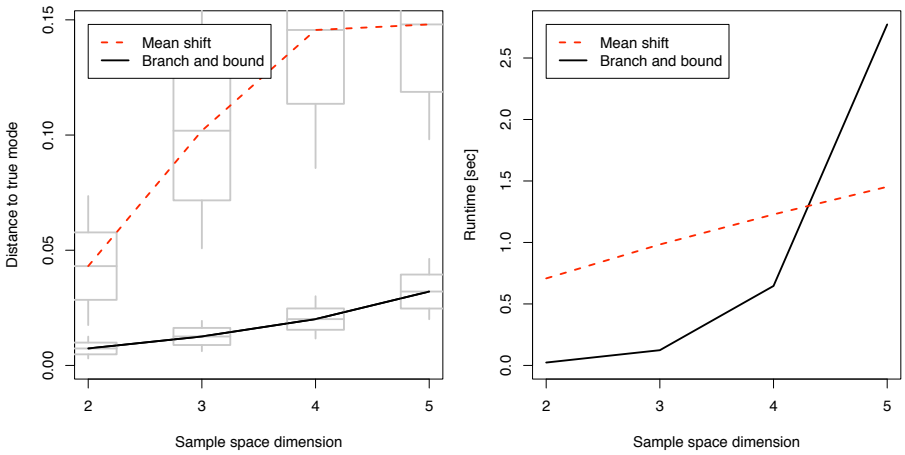
From the results of these experiments, we conclude that branch and bound mode finding is significantly more accurate than the mean shift algorithm, more robust to noise, and scales linearly with the number of samples. Linear scaling for the mean shift method when using a specialized implementation of a Gaussian kernel has been reported by other authors[24].

The unfavorable behavior of the runtimes our method for increasing data dimension compared to the mean shift can be explained by the growth of the search tree for high dimensional data. This observation gives hints for further optimization: Breadth-first search may sometimes be preferable to best-first search, also for lowering the memory requirements in high dimensional applications.

4.2 Samples from a multimodal distribution

Fig. 6 presents the ROC results for experiments in detecting $k = 25$ modes in a two and a three dimensional problem. Our method is the best out of these three in both cases, but tends to miss some modes. The accuracy gain of the proposed algorithm over the Gaussian mixture EM was smaller in the 3 than in 2-dimensional simulations.

This evaluation on samples from a multimodal density demonstrates the



(a) Our method outperforms mean shift for mode finding in all tested dimensions.

(b) For dimensions smaller 5, mean shift is slower than the branch and bound algorithm.

Figure 5: Experiments with samples from an isotropic Gaussian and 25% noise points in different dimensions. The total number of points was fixed to 5000.

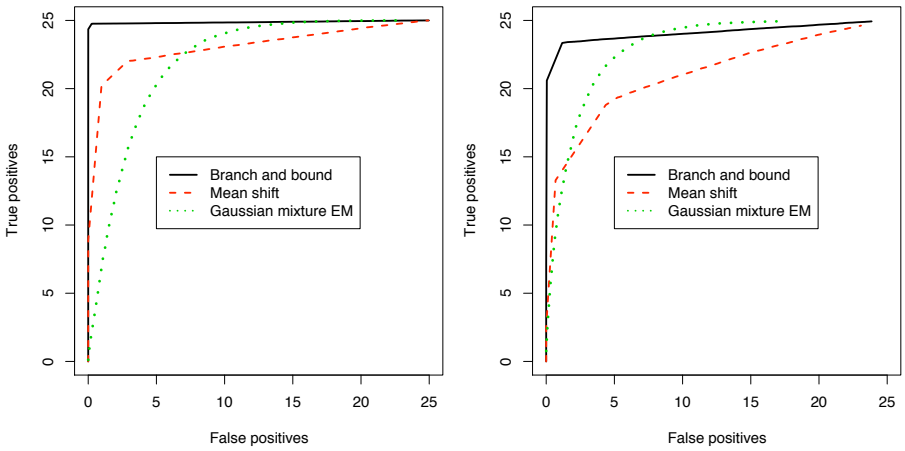
effectiveness of the greedy mode finding scheme outlined above. By removing active samples of detected modes from the remaining search steps, our greedy search will by design not find any modes closer than $h/2$ to one another. In the simulations presented here, the mixture components were allowed to become arbitrarily close, an unusual situation for real data. This explains the lower true positive rates of our method compared to the Gaussian mixture EM in Fig. 6(b).

5 Color Image Segmentation

An algorithm for segmenting images in joint coordinate and color space using the modes of kernel densities was proposed by Comaniciu and Meer[11]. There, the kernel definition in (2) was slightly modified by using two bandwidth parameters, h_c and h_s for the spatial and color component of the 5D feature space, respectively. Image color triples are transformed to the CIELUV color space, which was designed in such a way that measuring distances between the triples in this space was more closely related to human perception than in other color spaces.

This algorithm was implemented using either our branch and bound or the mean shift algorithm for mode detection. We chose the Berkeley Segmentation Dataset and Benchmark (`http://www.eecs.berkeley.edu/Research/Projects/CS/vision/`) for testing these methods with respect to one another. This dataset is an established benchmark set with human ground truth segmentations and an associated software for measuring the segmentation error[18].

Example images from this database have been used by Comaniciu to demon-



(a) Finding 25 modes in 2D (b) Finding 25 modes in 3D

Figure 6: Experiments with samples from a 25-Gaussian mixture ($\sigma = 0.25$) and 50% noise fraction. The total number of points was fixed to 5000. To measure these ROCs, the distance at which a true mode was counted as matching an estimated one was varied.

	Score (higher values are better)
Human[18]	0.79
Branch and bound	0.48
Mean shift	0.43

Figure 7: Performance on The Berkeley Segmentation Database and Benchmark, measured using the benchmarking software provided by D. Martin et al.

strate the performance of a modified version of the algorithm we use here[9]. We simply fix the bandwidth parameters to $h_c = 20$ and $h_s = 4$, as was reported in that paper. Then, we apply both algorithms to the Berkeley test set and evaluate the errors. The mean results of all 100 test images are given in the table in Fig. 7.

Note that the branch and bound results tend to oversegment the images. This is preferable to missing objects both according to the scoring system[18] and in practice.

Theses results demonstrate a performance gain when using the identical segmentation algorithm and kernel parameters, which is achieved by switching from the local mean shift search to global branch and bound mode finding. State-of-the-art results using both gray and color image segmentation are published on the The Berkeley Segmentation Database and Benchmark website.

Examples of segmentations from this database, along with the scores, are shown for both our branch and bound and the mean shift methods in Fig. 8

and 9.

6 Conclusions

This paper introduced a globally optimal algorithm for finding the modes in kernel density estimates, which is more accurate than the mean shift method in terms of mode localization. When using a conventional implementation of the mean shift, it is also more efficient and, in contrast to mean shift, scales linearly with sample size.

Numerical experiments on simulated data and on The Berkeley Segmentation Database and Benchmark have revealed the merits of using global rather than local optimization: Increased accuracy, especially when few samples are available or when there exist large distortions due to noise.

We have also shown the strong relations of our algorithm to the RAST class of algorithms known from geometric matching. This comparison revealed that a quality function used there for robust least squares matching under a Gaussian error model also had a rigorous interpretation in the kernel estimation framework: It is the MISE-optimal Epanechnikov kernel.

The proposed algorithm may be used as an out-of-the-box solution for mode finding when only some samples are available. Only two parameters are required: A bandwidth h, related to the scale of the given problem, and a accuracy ϵ governing the precision of the final solution.

Acknowledgments

We would like to thank Christoph Lampert for providing an open source implementation of RAST for geometric matching, which was the basis for our implementation.

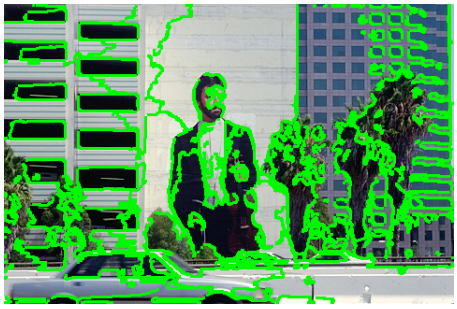
References

[1] H. Ali, C.H. Lampert, and T.M. Breuel. Satellite tracks removal in astronomical images. In *Proc. 11th Iberoamerican Congress on Pattern Recognition*, 2006.

[2] A. Bandera, J.M. Pérez-Lorenzo, J.P. Bandera, and F. Sandoval. Mean shift based clustering of hough domain for fast line segment detection. *Pattern Recognition Letters*, 27(6):578–586, Apr 2006.

[3] C.M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[4] T.M. Breuel. Fast recognition using adaptive subdivisions of transformation space. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 445–51, 1992.



(a) Image No. 119082 segmented using branch and bound; score 0.64.

(b) Image No. 119082 segmented using mean shift; score 0.62.

(c) Image No. 145086 segmented using branch and bound; score 0.67.

(d) Image No. 145086 segmented using mean shift; score 0.56.

(e) Image No. 147091 segmented using branch and bound; score 0.66.

(f) Image No. 147091 segmented using mean shift; score 0.26.

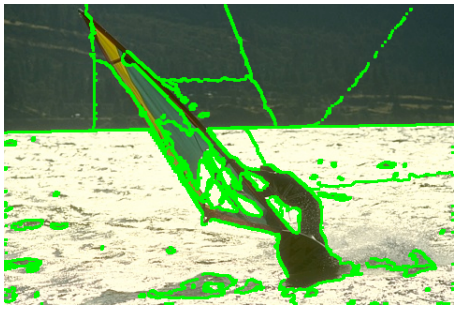
Figure 8: Some examples of segmented images from The Berkeley Segmentation Database and Benchmark. We show the results of branch and bound against mean shift segmentation both graphically and quantitatively. For details on algorithms and parameters, refer to the text.

(a) Image No. 216081 segmented using branch and bound; score 0.63.

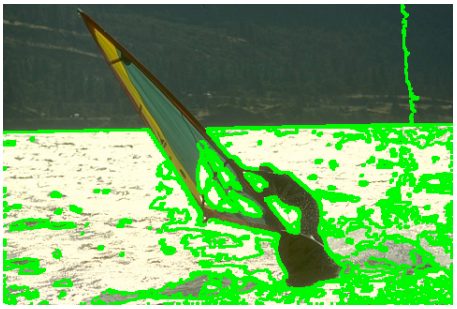
(b) Image No. 216081 segmented using mean shift; score 0.59.

(c) Image No. 42049 segmented using branch and bound; score 0.75.

(d) Image No. 42049 segmented using mean shift; score 0.62.



(e) Image No. 62096 segmented using branch and bound; score 0.70.



(f) Image No. 62096 segmented using mean shift; score 0.42.

Figure 9: Further examples, a for description see Fig. 8.

[5] T.M. Breuel. Finding lines under bounded error. *Pattern Recognition*, 29(1):167–178, 1996.

[6] T.M. Breuel. On the use of interval arithmetic in geometric branch-and-bound algorithms. *Pattern Recognition Letters*, 24(9–10):1375–1384, 2003.

[7] M.A. Carreira-Perpinan. Gaussian mean-shift is an EM algorithm. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 29(5):767–776, May 2007.

[8] R.T. Collins. Mean-shift blob tracking through scale space. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, volume 2, pages 234–240, 2003.

[9] D. Comaniciu. Image segmentation using clustering with saddle point detection. In *Proc. Int. Conf. Image Processing*, volume 3, pages 297–300, 2002.

[10] D. Comaniciu and P. Meer. Mean shift analysis and applications. In *Proc. 7th Int. Conference Computer Vision*, volume 2, pages 1197–1203, 1999.

[11] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 25(5):603–619, May 2002.

[12] D. Comaniciu and V. Ramesh. Mean shift and optimal prediction for efficient object tracking. In *Proc. IEEE Int. Conf. Image Processing*, volume 3, pages 70–73, 2000.

[13] D. Comaniciu, V. Ramesh, and P. Meer. The variable bandwidth mean shift and data–driven scale selection. In *Proc. 8th Int. Conf. Computer Vision*, volume 1, pages 438–445, 2001.

[14] M. de Berg, M. van Kreveld, M. Overmas, and O. Schwarzkopf. *Computational Geometry*. Springer, 2nd edition, 2000.

[15] B. Georgescu, I. Shimshoni, and P. Meer. Mean shift based clustering in high dimensions: A texture classification example. In *Ninth IEEE Int. Conf. Computer Vision*, volume 1, pages 456–463, 2003.

[16] E. Hansen and G.W. Walster. *Global Optimization Using Interval Analysis*. Pure and Applied Mathematics. CRC Press, 2nd edition, 2003.

[17] D. Keysers, T. Deselaers, and T.M. Breuel. Optimal geometric matching for patch-based object detection. *Electronic Letters on Computer Vision and Image Analysis*, 2007. accepted for publication.

[18] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. Computer Vision*, volume 2, pages 416–423, July 2001.

[19] K. Okada, M. Singh, and V. Ramesh. Prior-constrained scale-space mean shift. In *Proc. 17th British Machine Vision Conference*, volume 2, pages 829–838, 2006.

[20] D.W. Scott. *Multivariate density estimation: Theory, practice, and visualization*. Wiley, 1992.

[21] C. Shen, M.J. Brooks, and A. van den Hengel. Fast global kernel density mode seeking with application to localization and tracking. In *Proc. 10th Int. Conf. Computer Vision*, volume 2, pages 1516–1523, 2005.

[22] A. Ulges, C. Lampert, D. Keysers, and T.M. Breuel. Optimal dominant motion estimation using adaptive search of transformation space. In *Proc. DAGM Symposium*, pages 204–213, 2007.

[23] O. Wirjadi and T.M. Breuel. Global modes in kernel density estimation: Rast clustering. In *Proc. Hybrid Intelligent Systems*, pages 314–319, 2007.

[24] C. Yang, R. Duraiswami, N.A. Gumerov, and L. Davis. Improved fast gauss transform and efficient kernel density estimation. In *Proc. Ninth IEEE International Conference on Computer Vision (ICCV'03)*, volume 1, pages 464–471, 2003.