# Parameter Tuning by Pairwise Preferences

Pavel Kisilev
pavel.kisilev@hp.com
Daniel Freedman
daniel.freedman@hp.com

Hewlett-Packard Laboratories
Haifa, Israel

**Abstract**

That most computer vision algorithms rely on parameters is a fact of life which cannot be avoided. For optimal algorithm performance, these parameters need to be tuned; generally speaking, this tuning is done manually or in some heuristic fashion. In this paper, we propose a new, general method for attacking the problem of parameter tuning, which is applicable to a wide variety of computer vision algorithms. Our method is semi-automatic: a user is given several pairs of outputs from a given vision algorithm, which have been generated by different parameter values; the user is then required to simply choose, for each pair, which output is preferred. Our method then finds the smoothest preference function which satisfies these user preferences. Using the theory of Reproducing Kernel Hilbert Spaces, we show how this problem can be reduced to a finite-dimensional convex optimization. We validate our parameter tuning scheme both on simulated data and on the problem of tuning the parameters of an image denoising algorithm.

## 1 Introduction

Most computer vision algorithms have parameters. This is a fact of life which is familiar to any researcher in the field. Unfortunately, for algorithms to work properly, the parameters have to be tuned. It is this general problem, of choosing parameters for optimal algorithm performance, that we address in this paper.

At first blush, a general approach to this problem seems difficult; indeed, most prior work seems to have focused on parameter tuning for specific algorithms. There are many examples of such tuning systems for specific algorithms; we give a few examples to illustrate the flavour of these approaches. Yitzhaky and Peli [13] choose the parameters for edge detection through a procedure of estimating the ground truth edge maps of images, and then using standard statistical quantities (area under the ROC curve, etc.) to measure the quality of a particular set of parameters. Lindeberg [7] focuses on the problem of choosing the scale in low-level feature detection, and shows how to choose scale automatically through the computation of local extrema of combinations of normalized derivatives of the image. Broggi *et al*. [1] tune the parameters of a stereo vision system which is intended for mobile robotic or automotive purposes. Here the ground truth is known through the placement of markers on the hood of the test vehicle. In the Design Galleries paradigm [8], Marks *et al*. present a GUI which allows users to visualize the results of many different parameter values, and thus to choose the best from amongst this gallery. Parameter tuning systems have also been explored within the machine learning literature, including work on SVM parameters [2] and a general technique [6] which combines cross-validation with a greedy search method.

In this paper, we propose a semi-automatic approach to parameter tuning, which is general-purpose and can be used for a wide variety of computer vision algorithms. The basic setup is as follows. The vision algorithm takes as input (i) an actual input (commonly an image) and (ii) parameter values. From the input and the parameter values, it produces an output (sometimes an image, sometimes another quantity). Thus, a single run of the vision algorithm may be characterized by the triple $(input, parameter, output)$.

Now, the vision algorithm is run several times, leading to several triples of the form $\{(input_i, parameter,_i output_i)\}_{i=1}^m$. From these runs, the user is given pairs of outputs, and asked to judge which output is preferred, in that it constitutes a higher quality output. For example, one such pair might be $(output_1, output_2)$, and the user might decide that $output_1$ is preferred to $output_2$. This implies that the set $parameter_1$ is preferable to the set $parameter_2$, in that it produces a higher quality output, according the user. We denote this situation as

$$parameter_1 \succ parameter_2$$

The user provides such a pairwise preference for several pairs of outputs. Based on these pairwise preferences, the goal is to find a function over the parameter space which respects the user's preferences, i.e. such that if $parameter_1 \succ parameter_2$ then the function is larger for $parameter_1$ than for $parameter_2$.

This approach is attractive since it puts minimal demands on the user: at any given time, the user is only required to judge between two outputs, and to indicate which of the outputs is better. This is generally much easier than requiring that the user supply a full ranking on a large number of outputs, which for many applications is quite challenging.

Given the user's pairwise preferences, the problem can be formulated as one of finding the smoothest function which satisfies the user's preference constraints. The smoothness energy can often be specified as a norm in a Reproducing Kernel Hilbert Space, leading to the reduction of the problem at hand to a finite-dimensional convex optimization. Once the preference function has been computed, the optimal parameters can be found as those which maximize the preference function.

The remainder of the paper is organized as follows. Section 2 shows how to formally pose and solve the problem of optimal parameter tuning by pairwise preferences. We show, after some mathematical development, that the solution of the problem can be converted into a quadratic program, which can be solved by any off the shelf solver. Section 3 presents experiments, both on simulated data and on parameter tuning for image denoising. Section 4 concludes.

## 2 Theory and Algorithm

In this section, we outline the problem of optimally choosing parameters based on a user's pairwise preferences. We begin by formalizing the problem. We then recast the problem as one of minimizing the norm of a function in an appropriate Reproducing Kernel Hilbert Space (RKHS), subject to the pairwise preference constraints. We then show how to use the Representer Theorem [5, 10] to convert the optimization from an infinite-dimensional one into a convex, finite-dimensional one. The problem is in fact a quadratic program, which can be solved with standard software.

Before diving into the mathematics, we should note that similar problems have been posed in the machine learning community, e.g. [4]. In the machine learning context, one is typically interested in the problem of ranking a discrete number of items, with applications in information retrieval (e.g. search engines); it is sometimes the case that the input there is in the form of a user's pairwise choices. While this type of problem is clearly related to our problem, the motivations are quite different. In our case, one wishes to learn an entire continuous function – the preference function – over parameter space, which one can then optimize to find the optimal choice of parameters for a particular vision algorithm. In the machine learning setting, one wishes to rank a discrete set of elements, and this ranking is generally the algorithm's end goal.

Nonetheless, the methods share some technical commonalities. We provide an alternative derivation for the solution to the relevant optimization problem than is provided in [4]. Our derivation is more closely related to the problem of continuous function approximation from pairwise inputs than is the method provided in [4], and so is more appropriate for the task at hand.

## 2.1 Formalizing the Problem

Let us recall the scenario that we have proposed for optimal parameter tuning. The vision algorithm takes an input both an actual input as well as its parameter values; from the input and the parameter values, it produces an output. A single run of the vision algorithm is thus characterized by the triple $(input, parameter, output)$. The vision algorithm is run several times, leading to several triples $\{(input_i, parameter,_i output_i)\}_{i=1}^m$. From these runs, the user is given pairs of outputs, and asked to judge which output is preferred, in that it constitutes a higher quality output. For a pair like $(output_1, output_2)$, the user might decide that $output_1$ is preferred to $output_2$, which implies that $parameter_1$ is preferable to $parameter_2$, which we denote by $parameter_1 \succ parameter_2$. The user provides such a pairwise preference for several pairs of outputs. Based on these pairwise preferences, the goal is to find a function over the parameter space which respects the user's preferences, i.e. such that if $parameter_1 \succ parameter_2$ then the function is larger for $parameter_1$ than for $parameter_2$.

There are obviously many such functions; which one should be chosen? A natural choice, in the absence of any specific prior, is the *smoothest* such function. We now formalize the problem of choosing the smoothest function subject to the user's pairwise choices.

Let a parameter be denoted by the vector $x$, and the parameter space by $\mathscr{X} = \mathbb{R}^d$. Let the user preference function be denoted by $f : \mathscr{X} \to \mathbb{R}$, and let the set of such functions be denoted $\mathscr{F}$. Let $S[f]$ denote the "smoothness energy" of $f$; when $f$ is very smooth, $S[f]$ will be small, and when $f$ is not smooth, and has many variations or wiggles in it, then $S[f]$ will be large. We assume that the smoothness energy functional is non-negative, i.e. $S : \mathscr{F} \to \mathbb{R}_+$.

The parameter pairs are denoted by $\{(x_{2i-1}, x_{2i})\}_{i=1}^n$, where they have been ordered so that $x_{2i-1}$ is preferred to $x_{2i}$. One might think that this preference constraint could be written as $f(x_{2i-1}) > f(x_{2i})$; however, it is clear that with such a constraint, one may take $f(x_{2i-1})$ arbitrarily close to $f(x_{2i})$, with the resulting smoothest function being essentially constant. Thus, we write the constraints instead as

$$f(x_{2i-1}) \geq f(x_{2i}) + 1$$

where the 1 above is arbitrary, and may be replaced by any other constant.[1]

[1]It is also possible to replace the constraint by $f(x_{2i-1}) \geq f(x_{2i}) + \delta_i$, where the value of $\delta_i$ varies with pair $i$.

Thus, the optimization problem we wish to solve is

$$\min_{f \in \mathscr{F}} S[f] \quad \text{subject to} \quad f(x_{2i-1}) \geq f(x_{2i}) + 1,\ i = 1, \dots, n \tag{1}$$

This is an infinite-dimensional optimization over a continuous function $f$. In what follows, we show how to reduce this to a more manageable convex, finite-dimensional optimization for a particular class of smoothness energy functionals.

## 2.2 The Smoothness Energy Functional as a Norm in an RKHS

We begin by defining a Reproducing Kernel Hilbert Space (RKHS), and the norm within the RKHS. Our goal will then be to express our smoothness energy functional as norm within an RKHS. The reason for doing so will be the ability to convert our infinite-dimensional optimization problem into a simpler, finite-dimensional optimization, but we will turn to that conversion in Section 2.3. For the moment, we will focus on explaining how interesting classes of smoothness energy functionals may be captured as norms within given RKHS's.

Given a set $\mathscr{X}$, a kernel $k$ on $\mathscr{X}$ is defined as a function $k : \mathscr{X}^2 \to \mathbb{R}$ such that for all finite subsets $\{x_i\}_{i=1}^n$ of $\mathscr{X}$, the matrix $K$ with elements given by $K_{ij} = k(x_i, x_j)$ is positive definite.[2] In an intuitive sense, the function $k$ measures the similarity between objects in the set. A Reproducing Kernel Hilbert Space (RKHS) may then be defined as follows [9].
**Definition:** Let $\mathscr{X}$ be a nonempty set and $\mathscr{H}$ a Hilbert space of functions $f : \mathscr{X} \to \mathbb{R}$. Then $\mathscr{H}$ is called a *Reproducing Kernel Hilbert Space* endowed with the dot product $\langle \cdot, \cdot \rangle$ if there exists a function $k : \mathscr{X}^2 \to \mathbb{R}$ with the following properties:

1. $k$ has the reproducing property $\langle f, k(x, \cdot) \rangle = f(x)$ for all $f \in \mathscr{H}$.
2. $k$ spans $\mathscr{H}$, i.e. $\mathscr{H} = \overline{\text{span}\{k(x, \cdot) : x \in \mathscr{X}\}}$, where $\overline{Z}$ denotes the completion of the set $Z$.

Given an RKHS $\mathscr{H}$, the inner product defines a norm in the standard way, i.e. we define

$$\|f\|_{\mathscr{H}} \equiv \sqrt{\langle f, f \rangle}$$

Let us return to the main thread of our argument: we would like to express the smoothness energy functional as norm in an RKHS. We proceed by way of illustrative examples. A standard smoothness energy functional penalizes the second derivative of the function, i.e.

$$S[f] = \sum_{k,\ell=1}^{d} \left( \frac{\partial^2 f}{\partial x_k \partial x_\ell} \right)^2$$

This functional is sometimes called the *thin-plate spline* functional. It turns out that this $S[f]$ may be expressed as the square of the norm $\|f\|_{\mathscr{H}}^2$ in an RKHS $\mathscr{H}$ with kernel $k$ given by

$$k(x, x') = \begin{cases} \|x - x'\|^d & \text{if } d \text{ is even} \\ \|x - x'\|^d \log \|x - x'\| & \text{if } d \text{ is odd} \end{cases}$$

This may be relevant in situations where the user is able, for example, to distinguish between the case where $x_{2i-1}$ is "better" than $x_{2i}$ and the case where $x_{2i-1}$ is "much better" than $x_{2i}$. The derivation below carries through for the case of varying $\delta_i$ with minor modifications.

[2]There is an issue with terminology here: sometimes a kernel is just a function $k : \mathscr{X}^2 \to \mathbb{R}$, and a kernel which satisfies the extra positive definiteness condition is a *positive* kernel. We will have no use for non-positive kernels here, so we will keep the notation as is.

See, for example, the paper by Duchon [3].

Another smoothness energy functional may be expressed more easily in the Fourier Domain. Let

$$S[f] = \int \frac{\|\tilde{f}(\omega)\|^2}{\tilde{G}(\omega)} d\omega$$

where $\tilde{f}$ denotes the Fourier Transform of $f$. The function $\tilde{G}(\omega)$ should be small when $\omega$ is large; this will effectively penalize high frequency oscillations in $f(x)$. This $S[f]$ may be expressed as the square of the norm $\|f\|^2_{\mathscr{H}}$ in an RKHS $\mathscr{H}$ with kernel $k$ given by $k(x,x') = G(x-x')$, where $G$ is the Inverse Fourier Transform of $\tilde{G}$. For example, we may choose $\tilde{G}(\omega) = |\sigma| e^{-\sigma^2 \|\omega\|^2/2}$, in which case $K(x,x') = e^{-\|x-x'\|^2/\sigma^2}$, the familiar Gaussian kernel.

## 2.3 The Representer Theorem

We have shown that the smoothness energy functional can be expressed as the norm (or squared norm) in an RKHS.[3] In this case, we may rewrite the optimization problem (1) as

$$\min_{f \in \mathscr{H}} \|f\|_{\mathscr{H}} \quad \text{subject to} \quad f(x_{2i-1}) \geq f(x_{2i}) + 1,\ i = 1, \ldots, n \tag{2}$$

Let us further rewrite the constraints as follows.

$$C(f(x_1), \ldots, f(x_{2n})) = \begin{cases} 0 & \text{if } f(x_{2i-1}) \geq f(x_{2i}) + 1 \text{ for } i = 1, \ldots n \\ \infty & \text{otherwise} \end{cases}$$

Then we may rewrite our optimization as

$$\min_{f \in \mathscr{H}} C(f(x_1), \ldots, f(x_{2n})) + \|f\|_{\mathscr{H}}$$

To solve this problem, we draw on the famous Representer Theorem [5, 10], which has played an important role in machine learning.

**Theorem 1 (Representer Theorem)** *Let $\Theta : \mathbb{R}_+ \to \mathbb{R}$ be a strictly monotonically increasing function, and let $C : \mathbb{R}^{2n} \to \mathbb{R} \cup \{\infty\}$ be an arbitrary function. Then any minimizer of the problem*

$$\min_{f \in \mathscr{H}} C(f(x_1), \ldots, f(x_{2n})) + \Theta(\|f\|_{\mathscr{H}})$$

*admits a representation of the form*

$$f(x) = \sum_{i=1}^{2n} \alpha_i k(x, x_i) \tag{3}$$

The importance of this theorem is that it reduces the infinite-dimensional optimization where one must search over an entire space of functions, to a finite dimensional optimization in which one must simply find the best $2n$ scalars $\{\alpha_i\}_{i=1}^{2n}$. We now show how to perform this reduction explicitly.

[3] Note that it does not matter whether we optimize the norm or squared norm, as the resulting minimizing argument will be the same in both cases. We will thus move freely between these two objective functions.

## 2.4 Solving the Problem as a Quadratic Program

According to the Representer Theorem, the optimal solution for $f$ can be written as $f(x) = \sum_{i=1}^{2n} \alpha_i k(x, x_i)$. In this case, the squared norm of $f$ in the RKHS may be simplified as follows:

$$\begin{aligned}
\|f\|_{\mathscr{H}}^2 &= \langle f, f \rangle \\
&= \left\langle \sum_{i=1}^{2n} \alpha_i k(\cdot, x_i), \sum_{j=1}^{2n} \alpha_j k(\cdot, x_j) \right\rangle \\
&= \sum_{i=1}^{2n} \sum_{j=1}^{2n} \alpha_i \alpha_j \langle k(\cdot, x_i), k(\cdot, x_j) \rangle \\
&= \sum_{i=1}^{2n} \sum_{j=1}^{2n} \alpha_i \alpha_j k(x_i, x_j) \\
&= \alpha^T K \alpha
\end{aligned}$$

where in the fourth line, we have made use of the reproducing property of the kernel $k$, and in the final line we have defined $K$ as a $2n \times 2n$ matrix, and $\alpha$ as a $2n \times 1$ column vector, in the natural ways.

The constraint $f(x_{2i-1}) \geq f(x_{2i}) + 1$ may be rewritten as

$$\sum_{j=1}^{2n} \alpha_i k(x_{2i-1}, x_j) - \sum_{j=1}^{2n} \alpha_i k(x_{2i}, x_j) \geq 1 \tag{4}$$

Denote by $K_{odd}$ the $n \times 2n$ submatrix of $K$ with the odd rows selected, and $K_{even}$ the analogous submatrix with the even rows selected. Then we may rewrite the set of $n$ constraints in (4) as

$$B\alpha \geq e$$

where $B = K_{odd} - K_{even}$ and $e$ is the $n$-dimensional vectors whose entries are all 1's.

Thus, the optimization in (2) is reduced to

$$\min_{\alpha \in \mathbb{R}^{2n}} \alpha^T K \alpha \quad \text{subject to} \quad B\alpha \geq e \tag{5}$$

which is a quadratic program, and can be solved by any standard solver.

## 2.5 Summary

We now summarize the algorithm. We solve the quadratic program in (5) to get the optimal $\{\alpha_i^*\}_{i=1}^{2n}$. We then plug these into Equation (3) to find the smoothest preference function $f^*(x)$ satisfying the user constraints. Given this preference function $f^*(x)$, we may now find the optimal parameters by maximizing $f^*$, i.e.

$$x^* = \arg\max_{x \in \mathscr{X}} f^*(x)$$

This latter optimization can be performed exhaustively if the dimension of $\mathscr{X}$ is sufficiently small, i.e. 2 or 3, as in the applications described in Section 3. Alternatively, if the dimension of the parameter space is high, we may approximate the optimal parameter by the best of the parameter values that have been used for learning, i.e.

$$\tilde{x} = \arg\max_{x \in \{x_i\}_{i=1}^{2n}} f^*(x)$$

## 3 Results

In this section we present results of several experiments on simulated and real world data which show the effectiveness of the proposed method. In all cases, we used a Gaussian kernel; the bandwidth was chosen by a heuristic based on the density of the sampling which space constraints preclude us from describing at greater length. Very similar results come from using an inverse multiquadric kernel.

### 3.1 Simulated Data

The first set of experiments involve simulated data: we generate a true preference function $f_{true}(x)$, but our algorithm is only given access to pairwise preferences. That is, the algorithm is given pairs of the form $(x_a, x_b)$ where $f_{true}(x_a) > f_{true}(x_b)$; however, the algorithm is *only* given those pairs, and is given no access at all to the function $f_{true}$. In particular, it is important to note that the algorithm is not even given function values for the pairs $(x_a, x_b)$, but only their order (i.e. $x_a$ is preferred to $x_b$). In this way, we effectively simulate the problem of parameter tuning, but we have a ground truth preference function $f_{true}$ which we can compare to our approximated preference function.

We begin with 1D parameters, and generate $f_{true}$ as a 1D function from a mixture of three Gaussians. To generate the pairs of parameters, we randomly sample 16 values leading to 8 pairs of parameters; as explained above, the preference constraints within each pair are determined according to the function $f_{true}$. The left side of Figure 1 shows the true function $f_{true}$ in blue and and the estimated function $f$ in red. Even from the very limited data of 8 pairs we are able to predict the maxima quite accurately, and even reconstruct the overall shape of the unknown function. It is critical to emphasize that this is a blind reconstruction, in that we do not use the function values, but only the pairwise orders.

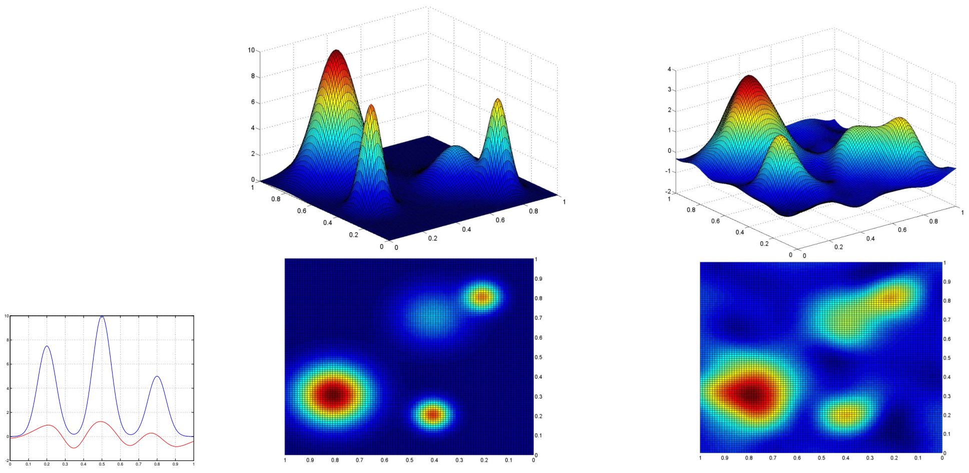
Figure 1: Simulated data examples of 'blind' reconstruction of functions. Left: 1D example - true function (blue) and reconstructed function (red). Center: original 2D function and its projection. Right: reconstructed 2D function and its projection.

For the case of 2D parameters, we generated a 2D function $f_{true}$ using a mixture of four 2D Gaussians, see Figure 1, center. The corresponding projection of the function in the lower

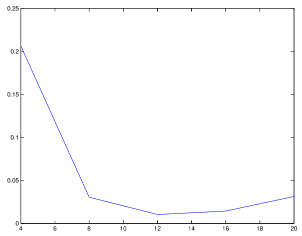

Figure 2: MSE of the maximum position of the 2D simulated example for varying number of pairs.

Figure 3: An image denoised using 16 different settings of bandwidth parameters of the Bilateral Filter.

part shows more clearly the position, that is the coordinates, of the function maxima. As in the 1D case, we generate pairs randomly. The right side of Figure 1 shows the reconstructed function with 16 pairs. As in the 1D case, we are able to predict correctly the position of maxima, and the shape of the function.

We then evaluate the algorithm performance for different numbers of pairs. For each of 4, 8, 12, 16, and 20 pairs, we run 100 independent experiments (i.e. randomly sample the pairs 100 times) and calculate the mean squared error (MSE) between the coordinates of the maximum of $f_{true}$ and the maximum of our reconstructed function $f$. The results are shown in Figure 2. Notice that the MSE is the lowest for 12 pairs, and is slightly larger for 16 and 20. This seems surprising, as we would expect the MSE to decrease for larger samples; the reason for the increase is that the heuristic for choosing bandwidth is not perfect, and selects a bandwidth which is larger than necessary for the latter two sample sizes.

## 3.2 Image Denoising

Next, we tested our method on image denoising applications. Our image denoiser is a Bilateral Filter (BF) [11] with two parameters, the range bandwidth and the spatial bandwidth; our goal is to determine their optimal values using our algorithm.

In our first experiment, we added Gaussian noise to a single image, and ran the BF with 16 combinations of bandwidth settings (4 different settings for each bandwidth parameter).

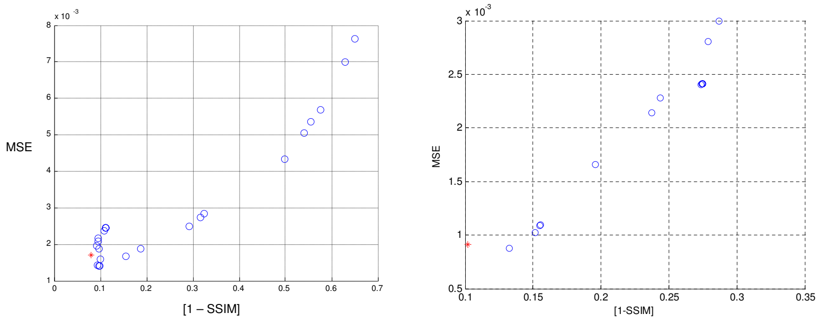

Figure 4: Scatter plot of MSE versus 1-SSIM of the denoised images. Left: single image denoising. Right: average over "test set" of 30 different images (see details in the text). The red star corresponds to the denoising with the estimated optimal parameter set.

The corresponding denoised images are shown in Fig. 3. Out of the 16 images, 8 pairs were randomly generated, and a user then judged each pair, selecting the output with better denoising. From these pairwise preferences, the optimal bandwidths were chosen.

To quantify the effectiveness of the algorithm, the following procedure was used. For each of the 16 images, we calculated two well known image quality measures, the mean squared error (MSE) and the structural similarity measure (SSIM) [12], using our knowledge of the noise-free image. The result of these computations is presented as a scatter plot in Figure 4, left. Using the bandwidths selected by our algorithm we again run the BF, and the corresponding MSE versus (1-SSIM) point is marked as a red star in the same plot. Note that our optimal parameters are close to the MSE optimum, and are better than the best SSIM result of the 16 settings in the test. This is a remarkable result as the SSIM index is known to reflect well the visual similarities of images, and our input data are based on purely on a visual pairwise comparison test.

Figure 5: The "learning set" of denoised images, using 16 different settings of bandwidth parameters of the Bilateral Filter.

Next, we run a similar experiment, but with Gaussian noise added to 16 *different* images (the "learning set"), with varying resolutions, illumination conditions, and so on (Figure 5). We used 16 settings of the bandwidth parameters; each image gets its own setting, and the rest of the optimal parameter determination is precisely as in the previous experiment, except that now the pairs consist of two *different* images. We evaluate on a separate "test set," consisting of 30 new images. For the optimal bandwidth parameter setting, as well as the 16 settings used above, we compute the average MSE and SSIM – where the average is taken over the 30 images of the test set. The scatter plot with the average MSE and SSIM values, for the 17 parameter settings, is shown in Figure 4, right. As in the case of a single image, our method yields the best SSIM index (red star), and is quite close to the best MSE result. This further proves the effectiveness of our method.

## 4 Conclusions and Future Directions

We have presented a new method for parameter tuning which relies on a user to specify pairwise preferences. The algorithm then computes the smoothest preference function over parameters consistent with this user input, and computes the parameters which maximize this preference function. We have shown the method's promise on simulated data and an image denoising application. In the future, we expect to use the method on a variety of vision algorithms, including segmentation and edge detection.

## References

[1] A. Broggi, M. Bertozzi, and A. Fascioli. Self-calibration of a stereo vision system for automotive applications. In *IEEE International Conference on Robotics and Automation, 2001. Proceedings 2001 ICRA*, volume 4, 2001.

[2] V. Cherkassky and Y. Ma. Practical selection of SVM parameters and noise estimation for SVM regression. *Neural Networks*, 17(1):113–126, 2004.

[3] J. Duchon. Splines minimizing rotation-invariant semi-norms in Sobolev spaces. *Constructive theory of functions of several variables*, pages 85–100, 1977.

[4] R. Herbrich, T. Graepel, and K. Obermayer. Large margin rank boundaries for ordinal regression. In *Advances in Large Margin Classifiers*. MIT Press, Pages, 2000.

[5] G. Kimeldorf and G. Wahba. Some results on Tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications*, 33(1):82–95, 1971.

[6] R. Kohavi and G.H. John. Automatic parameter selection by minimizing estimated error. In *Proceedings of the International Conference on Machine Learning*, pages 304–312, 1995.

[7] T. Lindeberg. Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30(2):79–116, 1998.

[8] J. Marks, B. Andalman, PA Beardsley, W. Freeman, S. Gibson, J. Hodgins, T. Kang, B. Mirtich, H. Pfister, W. Ruml, et al. Design galleries: A general approach to setting parameters for computer graphics and animation. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, page 400. ACM Press/Addison-Wesley Publishing Co., 1997.

[9] B. Scholkopf and A.J. Smola. *Learning with kernels: Support vector machines, regularization, optimization, and beyond.* MIT Press, 2002.

[10] B. Scholkopf, H. Herbrich, and A.J. Smola. A Generalized Representer Theorem. In *14th Annual Conference on Computational Learning Theory, COLT 2001*, pages 416–426. Springer, 2001.

[11] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Computer Vision, 1998. Sixth International Conference on*, pages 839–846, 1998.

[12] Z. Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[13] Y. Yitzhaky and E. Peli. A method for objective edge detection evaluation and detector parameter selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(8):1027–1033, 2003.