The Application of Approximation and Complexity Theory Methods to the Solution of Computer Vision Problems

MICHAEL HATZITHEODOROU

Department of Computer Science
Columbia University
New York, NY 10027

CUCS - 335 - 88

March 1988

Abstract. We survey aspects of approximation and complexity theory and their application to the numerous computer vision problems that require an approximate solution because only partial information is available. We consider ill-posed computer vision problems and the methods that can be employed towards reformulating them as well-posed. We are particularly interested in the surface reconstruction problem that is encountered in the construction of the $2\frac{1}{2}$-D sketch, and which has been formulated and solved using different methods. We apply regularization theory, information-based complexity, and other methods to the solution of this problem. Finally, the ***shape from shadows*** problem is formulated and the optimal error algorithm is constructed and analyzed.

1. Introduction.

In this paper we address the application of various theoretical methods to computer vision. There are many problems in low- and middle-level computer vision that are currently solved by a variety of ad hoc methods. These methods are not guaranteed to work, nor do they provide any means of assessing the quality of the solution.

Many computer vision problems can be modelled as instances of the following general approximation problem. Assume we have a space of functions X and a function f from this space. Also assume that we are given partial information about the function f. We want to approximate the result of applying a known operator S to f.

There are many advantages to modelling computer vision problems as approximation problems. First, questions about the existence and uniqueness of the solution can often be answered. Second, upper and lower bounds on the error of the solution can be calculated. These bounds are often sharp. Finally, we can calculate complexity bounds.

Typeset by $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX

We will illustrate in this paper the applicability of mathematical modelling to low- and middle-level computer vision. Towards this end we will start by surveying approximation theory methods. We will show how information-based complexity provides a general unified approach to the solution of many problems studied by other means. The methodology proposed by this theory also provides a framework suitable to the modelling of new problems.

Computer vision problems that have been solved using information-based complexity will be reviewed. We will also look into other problems that have been formulated and solved using regularization theory methods, and we will contrast these two approaches.

Motivation for this survey has been given by the surface reconstruction problem and the various different methods used for its solution. Surface reconstruction is a very important step in the image recovery process, and has appeared extensively in a variety of different formulations. Because of its importance, the surface reconstruction problem will be used as the prime example throughout this paper. To further support the applicability of the proposed modelling we will show how the surface reconstruction problem can be modelled and solved under the completely new formulation of shape from shadows.

The rest of the paper will be organized as follows :

We will begin by describing the various theoretical frameworks that exist in approximation theory. In section 2 we will discuss some of the more widely known numerical analysis and approximation theory approaches to the solution of the interpolation and approximation problems.

In section 3 we will discuss the general framework of information-based complexity. This theory can be used to model many different problems using one general formulation. Optimal algorithms for the solution of the various modelled problems can often be derived. We will show a number of problems that can be solved using this approach (see sections 5 and 6.)

In section 4 we review the concept of an ill-posed problem. We will discuss many computer vision problems and show why most of them are ill-posed. Furthermore we will describe how these can be transformed into well-posed problems, and consequently how they can be solved by applying the methods proposed by regularization theory.

In section 5 we will elaborate on a particular class of problems that deal with surface reconstruction. We will review the recent work done on the subject and see how this is related to the work described in sections 3 and 4. We will compare the solutions obtained using regularization theory and information-based complexity principles.

In concluding this paper, we will describe, in section 6, the shape from shadows problem, which belongs in the class of surface reconstruction problems. This problem is different from the ones analyzed in section 5, because it uses the shadows cast on a surface, to extract data. The surface is recovered solely from the information contained in these shadows. The use of shadows is a new form of information that has not been used until now. An algorithm for the solution of this problem will be given, and its performance will be analyzed.

2. Classical approaches.

We will begin this section by describing methods that recover real functions of one real variable, $f\colon \mathbb{R} \longrightarrow \mathbb{R}$, and then proceed to discuss methods for recovering functions of two variables. There is a huge amount of work on the topic of approximating functions. It is not feasible to discuss all of them. Instead we will focus on methods that are widely known and that can be used to illustrate the progress towards the general theory that will be discussed in section 3.

2.1 The Bernstein polynomials.

We first state the *Weierstraß Approximation Theorem*. For an arbitrary continuous function f, we can obtain a polynomial approximation whose error is not larger than ϵ, for any given arbitrarily small ϵ [45, 48]. In particular, it holds.

THEOREM 2.1. *If a function f is continuous on a finite interval $[a,b]$ then $\forall \epsilon > 0\ \exists n = n(\epsilon)$ and a polynomial P_n of degree n such that,*

$$\left|f(x) - P_n(x)\right| < \epsilon, \qquad \forall x \in [a,b]. \tag{2-1}$$

The proof is due to Bernstein [6] and consists of constructing polynomials of the form,

$$B_\nu(x) = \sum_{k=0}^{\nu} \binom{\nu}{k} x^k (1-x)^{\nu-k} f(\tfrac{k}{\nu}), \qquad x \in [0,1]. \tag{2-2}$$

It can be shown that the sequence $B_\nu(x)$ converges uniformly to $f(x)$.

One might be tempted to infer from theorem 2.1 that the problem of approximating a given function by a polynomial has been solved. Unfortunately, this is not the case for many reasons. The polynomials B_ν that are constructed in the proof given by Bernstein are usually of a much higher degree than the degree n of $P_n(x)$ [48]. Therefore, the cost of a solution that uses the polynomials B_ν is high. At the same time it is known that polynomials of a high degree oscillate. We would like to keep this oscillation as small as possible.

For these reasons, the practicality of the Bernstein polynomials is limited. This does not reduce the importance of theorem 2.1, that shows the existence of a polynomial approximation. Such approximations are often desirable, because polynomials are simple and easy to compute. Therefore, we will now attempt to find methods that yield better polynomial approximations.

2.2 Lagrange and piecewise Lagrange interpolation.

Lagrange and Newton interpolation are among the best known methods for interpolating a set of data. They both result in the same interpolating polynomial and yield the same error. The uniqueness of the interpolating polynomial is given by the following theorem [45].

THEOREM 2.2. *Let $\{x_i\}_{i=0,1,\ldots,n}$ be any set of $n+1$ distinct points in the interval $[a,b]$ and let $f\colon [a,b] \longrightarrow X \subseteq \mathbb{R}$ be an $(n+1)$ times continuously differentiable function. Then $\exists!$ polynomial $P \in \mathcal{P}_n$ that satisfies $P(x_i) = f(x_i)$, $\forall i = 0,1,\ldots,n$.*

The proof is based on constructing the polynomial,

$$P(x) = \sum_{k=0}^{n} f(x_k)\, l_k(x), \tag{2-3}$$

where

$$l_k(x) = \prod_{\substack{j=0\\ j\neq k}}^{n} \frac{(x - x_j)}{(x_k - x_j)}, \qquad x \in [a,b]. \tag{2-4}$$

The error of the unique interpolating polynomial is given by,

$$e(x) \;=\; |f(x) - P(x)| \;=\; \frac{f^{(n+1)}(\xi)}{(n+1)!} \prod_{j=0}^{n} (x - x_j), \qquad \xi \in \operatorname{span}\{x, x_1, \ldots, x_n\}. \tag{2-5}$$

Although Lagrange interpolation has been widely used, it has certain shortcomings. For a sample of $n+1$ points we require that $f^{(n+1)}(x)$ exists and is continuous $\forall x \in [a,b]$. We would like to relax this assumption so that we can approximate functions of a lower regularity. Additionally, the degree of the interpolating polynomial P will be high. Furthermore, P will oscillate wildly [50] which, as we already mentioned, is often undesirable. To rectify these problems we are led to use a *piecewise* polynomial approximation method. Piecewise Lagrange interpolating polynomials of some fixed degree can be easily constructed.

Piecewise methods have only one shortcoming. The resulting approximations tend to be discontinuous at the points where the polynomial pieces are joined together, and hence do not model the approximated functions very well.

2.3 Spline interpolation.

To achieve a given smoothness using piecewise polynomials of a given low degree we will use *spline* functions. Spline functions have been extensively dealt with in classical approximation theory [1, 9, 47, 49, 50]. More general definitions of splines in normed linear spaces can also be found in [3, 4, 5, 10, 11, 57, 58, 60, 61, 72, 73]. We will discuss the classical work in this section and the more recent work in section 3.

Given certain boundary conditions and the required degree of smoothness, the interpolating spline can be uniquely determined [1]. A general definition of the spline interpolant $S(x)$ is given in [45] and is,

$$S(x) = \sum_{i=1}^{n+k} f(x_i)\, B_i(x), \tag{2-6}$$

where $f(x_i)$ are the data, i.e. the evaluations of the approximated function at the points x_i, with $x_1 < x_2 < \cdots < x_{n+k}$, k is the degree of the interpolant, and $B_i(x)$ are the ***basis splines*** which form a basis of the space W_k of spline functions of degree k [9, 45, 47], and are given by,

$$B_i(x) = \sum_{j=i}^{i+k+1} \left(\prod_{\substack{l=i \\ l \neq j}}^{i+k+1} \frac{1}{x_l - x_j} \right) (x - x_j)_+^k, \tag{2-7}$$

where $(x - x_j)_+^k = (x - x_j)^k$ for $x > x_j$ and 0 otherwise. It can be seen that $B_i(x) = 0$ for $x \notin (x_i, x_{i+k+1})$.

In the above definition, and in all methods discussed until now, we assumed information consisting of function values. In DeBoor [9] the definition of spline functions is generalized to include the use of information consisting of evaluations of functionals.

THEOREM 2.3. *Let $\{B_j\}_{j=1,\ldots,n}$ be a basis of the space W_k of spline functions of degree k and let $\{\lambda_i\}_{i=1,\ldots,n}$ be a set of functionals $\lambda_i\colon X \longrightarrow \mathbb{R}$, where X is the space of functions we are approximating. The interpolating spline $S(x)$ can then be constructed and is given by,*

$$S(x) = \sum_{i=1}^{n} a_i\, B_i(x), \tag{2-8}$$

where $\{a_i\}_{i=1,\dots,n}$ *are the solutions of the system* $\mathbf{A}\ \vec{\alpha} = \vec{\beta}$ *with* $\mathbf{A} = \{\lambda_i(B_j)\}_{i,j=1}^{n}$ *and* $\vec{\beta} = [\lambda_1(f), \dots, \lambda_n(f)]^{\top}$.

The spline functions (2-8) have some very interesting properties.

(1) They are piecewise polynomials of a relatively low degree, and therefore do not oscillate wildly. Their degree now depends on k and not on the number of sample points n.

(2) They have given fixed regularity over their entire domain.

(3) They are simple to construct and they minimize certain norms or semi-norms.

These properties are very useful, and will be discussed again in the following sections.

2.4 Lagrange and piecewise Lagrange interpolation for functions in $\mathbb{R}^2$.

In this section we will discuss methods for the approximation of real functions of two variables, $f : [a, b] \times [c, d] \longrightarrow \mathbb{R}$. Straightforward generalization of sections 2.2 and 2.3 is not always possible. For data arbitrarily scattered on some planar region, the Lagrange interpolating polynomial does not always exist [46], and the result is not unique. Furthermore, it is possible in most cases to find a polynomial of lower degree solving the same interpolation problem. Additional problems are encountered if a piecewise method is used. The problems lie in the choice of the area of support/domain of each piece and the choice of the common boundaries between pieces [46, 47]. To overcome these problems we will assume rectangular grids.

A rectangular grid is defined as $\Pi = \{(x_i, y_j),\ 0 \leq i \leq n, 0 \leq j \leq m\}$. The Lagrange interpolating polynomial defined on Π is derived from the Lagrange interpolating polynomial in each direction. Its existence is given by the following theorem [47].

THEOREM 2.4. *Let* $f : [a, b] \times [c, d] \longrightarrow \mathbb{R}$ *and information consisting of function values* $\{f(x_i, y_j),\ \ 0 \leq i \leq n,\ \ 0 \leq j \leq m\}$. *Then,* $\exists!$ *polynomial* $P(x, y)$ *of degree* n, *in the x-direction, and degree* m, *in the y-direction, solving the interpolation problem* $P(x_i, y_j) = f(x_i, y_j)$.

The Lagrange interpolating polynomial is given by

$$P(x, y) = \sum_{\substack{0 \leq i \leq n \\ 0 \leq j \leq m}} f(x_i, y_j)\, l_{ij}(x, y). \tag{2-9}$$

$P(x, y)$ is the aproximating polynomial of degree $m+n$ that interpolates the given function values, and $l_{ij}(x, y) = l_i(x) l_j(y)$ where $l_i(x)$, $l_j(y)$ are as defined in (2-4).

The uniqueness of the interpolating polynomial has been obtained by imposing sampling on a rectangular grid. The following holds.

THEOREM 2.5. *Let $f \in C^{(n+1),(m+1)}[a,b] \times [c,d]$ be a real function of two real variables. Then, the error of the Lagrange polynomial (2-9) is given by,*

$$e(x,y) = \beta \ \max\{h_x, h_y\}, \tag{2-10}$$

where $h_x = \max_{i=0,\ldots,n}(x_{i+1} - x_i)$, $h_y = \max_{j=0,\ldots,m}(y_{j+1} - y_j)$, and β depends on the partial derivatives of f in each direction.

The shortcomings of Lagrange interpolation in two dimensions are similar to the shortcomings of the one-dimensional version. Some of them can be overcome by using a piecewise method, which is defined as follows.

DEFINITION 2.1. *Given a rectangular grid Π which can be divided into subgrids Π_{nm} each one containing $n \times m$ information points, the piecewise Lagrange interpolant is given by,*

$$P_{nm}(x,y) = \sum_{\substack{0 \leq i \leq n \\ 0 \leq j \leq m}} f(x_i, y_j)\, l_{ij}(x,y), \tag{2-11}$$

where $l_{ij}(x,y)$ are defined in (2-9).

If we cover the whole domain with pieces of the form (2-11), we obtain the required approximation to our function. The error of this approximation is the maximum of the errors of the individual pieces, $e_p(x,y) = \max\{e_{nm}(x,y)\}$.

Two-dimensional piecewise Lagrange interpolation suffers from the same problems as the one-dimensional version. The partial derivatives of the interpolant do not exist at the boundaries between the pieces.[1] A different approach must be used when some degree of smoothness is required.

2.5 Spline interpolation for functions in $\mathbb{R}^2$.

We will now define an interpolant that maintains a fixed degree of smoothness throughout the domain. This property is possessed by spline functions [1, 9, 47, 50] (see also sections 3, 4, and 5.)

The interpolants in two dimensions are defined as sums of products of one-dimensional basis splines. From [9] we obtain the following definitions.

[1]The interpolant can even be discontinuous at the boundaries between the pieces.

DEFINITION 2.2. *Let $\tilde{U}$, V be linear spaces of functions $f : K \subseteq \mathbb{R} \longrightarrow \mathbb{R}$. Then, $\forall u \in U$, $\forall v \in V$, and for $(x, y) \in K \times K$ the product $w(x, y) = u(x)v(y)$ is called the tensor product of u and v and is symbolized by $u \otimes v$.*

DEFINITION 2.3. *The tensor product of the linear spaces U and V is defined as,*

$$U \otimes V = \{w \mid w = \sum_{i=1}^{n} a_i(u_i \otimes v_i), \quad a_i \in \mathbb{R},\ u_i \in U,\ v_i \in V,\ n \in \mathbb{N}\}. \tag{2-12}$$

Assume that U is the space generated by the basis splines $\{B_i\}_{i=1,\dots,n}$ that solve the interpolation problem in the x-direction, and V is the space generated by $\{\bar{B}_j\}_{j=1,\dots,m}$ solving the same problem in the y-direction. Then we have [9],

THEOREM 2.6. *Let $U = span\{B_i\}_{i=1,\dots,n}$, and $V = span\{\bar{B}_j\}_{j=1,\dots,m}$ be the spaces of functions solving the problems of interpolating $f \in X$, given $\{\lambda_i(f)\}_{i=1,\dots,n}$, and $g \in Y$, given $\{\mu_j(g)\}_{j=1,\dots,m}$, respectively. Then $\exists S \in U \otimes V$ solving the interpolation problem $h = f \otimes g \in X \otimes Y$ given $\nu_{ij}(h) = \lambda_i(f)\,\mu_j(g)$, $\forall i, j$. S is given by,*

$$S(x, y) = \sum_{i,j} \nu_{ij}(h)\,(B_i \otimes \bar{B}_j)(x, y). \tag{2-13}$$

Other versions of theorem 2.6 can be found in [1, 47]. This work on splines has been extended to more general spaces in [4, 5, 10, 11, 37, 38, 39, 40, 57, 58, 72, 73].

2.6 Spline interpolation for functions in $\mathbb{R}^2$ and noisy data.

The problem of interpolating a set of noisy data has been addressed by Wahba in [60, 61]. The interpolating function minimizes the weighted sum,

$$\theta(f) = \left(\sum_{i=1}^{n} \frac{(y_i - f(t_i, s_i))^2}{n}\right) + \lambda\, \|f\|, \tag{2-14}$$

for some (semi)-norm $\|\cdot\|$. The following theorem holds [61].

THEOREM 2.7. *Let W be a semi-Hilbert space of real valued functions f defined on $\mathbb{R}^2$, and equipped with a semi-norm $\|\cdot\|_m$ of the form,*

$$\|f\|_m = \int_{\mathbb{R}^2} \sum_{k+l=m} \left(\frac{\partial^m f}{\partial x^k \partial y^l}\right)^2 dx dy. \tag{2-15}$$

Then the function minimizing the expression (2-14) is,

$$S(x,y) = \sum_{i=1}^{n} c_i K_{t_i,s_i}(x,y) + \sum_{j=1}^{2m(m+1)} d_j \phi_j(x,y). \tag{2-16}$$

K_{t_i,s_i} is the reproducing kernel of the space W and ϕ_j the linearly independent polynomials that produce the null-space of $\|\cdot\|_m$. The reproducing kernel is given by,

$$K_{t_i,s_i} = \theta_m \left[(x-t_i)^2 + (y-s_i)^2\right]^{m-1} \log\left[(x-t_i)^2 + (y-s_i)^2\right], \tag{2-17}$$

where $\theta_m = \frac{1}{2^{m-1}\pi[(m-1)!]^2}$. The coefficients c_i and d_j are obtained by solving a system of linear equations.

The above is similar to the work of Atteia, Duchon, and Meinguet on reproducing kernel Hilbert spaces [3, 4, 5, 10, 11, 37, 38]. Reproducing kernel Hilbert spaces, and their applications, will be discussed in section 5.

The parameter λ determines how close our approximation should be to the data. Its choice is not obvious and has been determined heuristically in many applications [52, 53]. We will discuss some of those applications in sections 4 and 5. One of the methods that is used for calculating λ is the method of *cross-validation* [32, 64, 71].

2.7 Other approaches.

There are many more methods that can be used for interpolating a set of data points on the plane. These methods have been developed for use in various disciplines, and can also be applied to computer vision.

In particular, we have not discussed *local* spline methods. Local methods construct a different interpolant for every small subset of the data. The approximation to one set of data is completely independent from the approximation to the neighboring set. In this respect local spline methods are similar to the piecewise Lagrange methods, but they do usually guarantee some degree of smoothness. The methods developed by Akima and Shepard are local methods. The interested reader can find an excellent survey of those and other methods in [14].

3. Information-based complexity.

In this section we will describe a framework under which we can solve a large number of problems. We will provide tools for the calculation of upper and lower bounds on the computational complexity of certain approximately solved problems as well as upper and lower bounds on the error of the algorithms used in obtaining solutions. The main core of work we will be describing will be from [57, 58, 69, 72, 73]; but see also the work of Michelli, Rivlin, and others [39, 40].

Information-based complexity obtains solutions to problems that have to be solved approximately, because only *partial*, or *contaminated* information about them is known.

We will discuss in this paper the *worst case* setting. It is defined as follows.

i. Uncertainty is measured by a norm.
ii. Error and cost are measured for the worst possible case, i.e. for the elements in our spaces that maximize them.

The worst case setting is not the only setting used in information-based complexity. The *average* and *probabilistic* settings are also studied. We will only briefly mention these areas in section 3.5.

3.1 Problem formulation and information.

Let F be a subset of a linear space F_1, and F_2 a linear space equipped with a norm $\|\cdot\|$. Let S be a mapping $S : F \longrightarrow F_2$. S is called the *solution operator*.

Given $\epsilon > 0$, we want to compute an approximation $x_\epsilon = x(\epsilon, f) \in F_2$ to the image $S(f)$ of an element $f \in F$ under the mapping S. We want, x_ϵ to be as *close* to the element $S(f)$ as possible and definitely no more than ϵ away.

Formally, x_ϵ should satisfy

$$\|S(f) - x_\epsilon\| \leq \epsilon. \tag{3-1}$$

Formulation of a problem in the above notation consists of the definition of the spaces F, F_2, the norm $\|\cdot\|$, and the operator S.

As mentioned before, the element $f \in F$ and its image $S(f) \in F_2$ are not known. To obtain an approximation, some information about the element f must be known, else it is easy to show that we cannot solve the problem. Therefore, assume that we are allowed to compute $L(f)$, for some functionals $L : F_1 \longrightarrow \mathbb{R}$. A natural choice for L are functionals of the form $L_x : F_1 \longrightarrow \mathbb{R}$ which $\forall f \in F_1$ return the value $f(x)$ of the function f at the point x. The functionals L_x are widely used in practice.

Assume now that we have a set of functionals $\{L_i\}_{i=1,\ldots,n}$. Define the operator $N : F_1 \longrightarrow F_3 \subseteq \mathbb{R}^n$ that produces $\forall f \in F$ the vector,

$$N(f) = [L_1(f), L_2(f), \ldots, L_n(f)]^{\mathsf{T}}. \tag{3-2}$$

$N(f)$ is all we know about the function f, except the a priori knowledge of the space F. $N(f)$ is the *information* about f and is named the *information operator*. The number n is called the *cardinality* of information.

The information $N(f)$ is, in general, not sufficient for the exact recovery of the function f. This happens because the operator N is not 1-1. Intuitively, the higher the cardinality of the information n, the better approximation we can obtain.[2] On the other hand, the evaluation of $L_i(f)$ may be costly. Hence we would like to keep the cardinality n as small as possible when computing an ϵ-approximation.

We can define two types of information operators; N^{non} and N^{a}. The first is called *non-adaptive* information and is of the form mentioned above,

$$N^{\mathrm{non}}(f) = [L_1(f), L_2(f), \ldots, L_n(f)]^{\mathsf{T}}, \tag{3-3}$$

where $L_i(f)$ are computed independently from each other. This means that the computation of the vector $\vec{y} = N(f)$ can be performed in parallel, a very useful and desirable property.

The second type of information is called ***adaptive***. Here, the computation of the value of $L_i(f)$ depends on the values of $L_j(f)$, $\forall j < i$. The adaptive information operator is of the form,

$$N^{\mathrm{a}}(f) = [L_1(f), L_2(f, y_1), \ldots, L_n(f, y_1, \ldots, y_{n-1})]^{\mathsf{T}}, \tag{3-4}$$

where $y_i = L_i(f, y_1, \ldots, y_{i-1})$. The operator N^{a} has a richer structure than N^{non}. This richer structure can be useful in many problems [57, 73]. On the other hand, for some problems and under certain assumptions, it can be shown that N^{a} is not stronger than N^{non} [57, 58, 68, 72].

3.2 Radius of information.

We have defined the problem and the information available for its solution. We define next the concept of the ***radius of information***. The radius of information measures the intrinsic uncertainty of the problem given information $\vec{y} = N(f)$.

Assume that the computed information vector is $\vec{y}$. The inverse image of $\vec{y}$ under N^{-1} is the set $V(f) = \{\bar{f} \in F \,:\, N(\bar{f}) = N(f)\}$, the set of all the problem elements that are indistinguishable under the information $\vec{y}$. The image of $V(f)$ under the mapping S is $S(V(f)) = U(f)$, (see Fig. 1.)

[2] Although there are cases when this does not hold (see [57].)

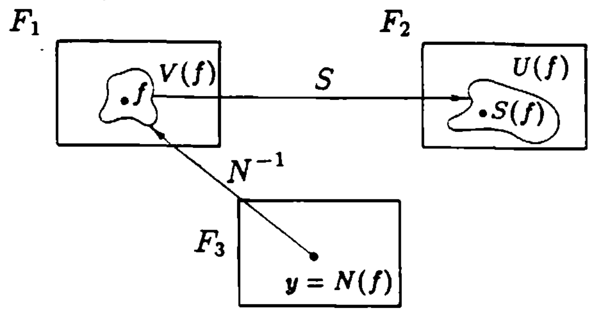

Figure 1.

Clearly, $S(f) \in U(f)$. Since the operator N is usually not 1-1, the set $V(f)$ and consequently the set $U(f)$ are not singletons.

We want the approximation x_ϵ to be as close to $S(f)$ as possible. Since $S(f)$ is not known and is only known to be in $U(f)$, it can be seen that as $U(f)$ gets smaller, the maximum distance between x_ϵ and $S(f)$ gets smaller.

DEFINITION 3.1. *[58] Let $f \in F$, information $N(f)$ and the set $U(f)$ be as defined above. Then, the radius of information $r(N, f)$ is,*

$$r(N, f) = \sup_{\tilde{y} \in N(f)} rad(U(f)), \tag{3-5}$$

where $rad(A)$ is the radius of the smallest ball containing A.

The radius of information can be used as a measure of the best possible approximation. In particular, we can find an ϵ-approximation x_ϵ to every $f \in F_1$ if and only if $r(N, f) \leq \epsilon$. Intuitively, if $r(N, f) \leq \epsilon$, we can choose x_ϵ to be the center of the ball, in which case we are guaranteed to be no more than ϵ away from $S(f)$. This is not true if $r(N, f) > \epsilon$. A straightforward proof can be found in [58, 73].

3.3 Algorithms.

We now discuss algorithms for obtaining an ϵ-approximation. An *algorithm* is a mapping $\varphi : F_3 \longrightarrow F_2$ that takes the information $N(f)$ and computes an approximation $x_\epsilon = \varphi(N(f))$ to $S(f)$.[3]

[3]This is a very general definition of an algorithm and does not assume a particular model of computation.

The error of the algorithm φ given information $N(f)$ is the distance between the value of $\varphi(N(f))$ and the unknown value $S(f)$. Formally,

$$e(\varphi, N) = \sup_{f \in F} \|S(f) - \varphi(N(f))\|, \tag{3-6}$$

for some information operator N.

The radius of information is a sharp lower bound on the error of any algorithm that solves the problem using the given information $N(f)$ [58, 73]. Namely, it holds :

$$r(N, f) = \inf_{\varphi} e(\varphi, N). \tag{3-7}$$

The algorithm φ^* that satisfies $e(\varphi^*, N) = r(N)$ is called ***optimal error algorithm.***

The error of an algorithm is not the only issue we are facing when solving a problem. Another issue is the cost of the proposed solution. Expensive algorithms may be abandoned in favor of cheaper ones.[4] This can be seen in the following example.

Assume that an algorithm φ_1 costs 3 times less than another algorithm φ_2 and makes twice the error, using the same information. Then, φ_1 using a sample that is twice as large, can have the same error as φ_2 and still cost less. This difference between the costs of the two algorithms may permit the use by φ_1 of information of higher cardinality and thus result in a better approximation.

3.4 Spline algorithms for linear problems.

We will now discuss a problem formulation where cheap optimal error algorithms [39, 57, 58, 69, 72] can always be obtained. Assume that we have a linear mapping $T : F_1 \longrightarrow F_4$. Define $F_0 \subset F_1$ to be, $F_0 = \{f \in F_1 : \|Tf\| \leq 1\}$. The operator T that is used to restrict the space of elements is called the ***restriction operator.***

DEFINITION 3.2. *Let F_1, F_2, and F_4 be linear spaces, S and T linear mappings, and information N consisting of linear functionals. The problem defined this way is called a linear problem.*

It is not feasible to cover all the known results about linear problems in this survey. We will mention only two interesting points. The reader is referred to [39, 40, 57, 58, 72] for further discussion of linear problems.

[4]The cost is calculated assuming that information of fixed cardinality is used.

The first issue deals with adaptive versus non-adaptive information. It can be shown [57, 58, 68] that for linear problems,

$$r(N^{\text{non}}) \leq 2\, r(N^{\text{a}}). \tag{3-8}$$

The inequality (3-8) means that adaption does not help, modulo a constant factor of 2, in the solution of linear problems. Furthermore, the simpler structure of N^{non} allows the evaluation of the functionals $L_i(f)$ to be done in parallel and reduces the total cost of the approximation.

The second issue deals with the existence of a linear optimal error algorithm. It can be shown, see [69], that in this formulation of the problem the spline algorithm is an optimal error algorithm. A spline is defined as follows [2, 3, 25].

DEFINITION 3.3. *Let spaces F_1, F_2 and F_4 and operators T and N be defined as before, and information vector $\vec{y} = N(f)$. Then, a spline $\sigma = \sigma(\vec{y})$ is an element of F_1 such that,*

i. $N(\sigma) = \vec{y}$.
ii. $\|T\sigma\| = min_{f\in F_1}\{\|Tf\|,\ N(f) = \vec{y}\}$.

Thus, the spline σ interpolates the information $N(f)$, and minimizes the norm $\|\cdot\|$. This property will be very useful in sections 4 and 5. In contrast to the traditional definitions of splines given in section 2, definition 3.3 is non-constructive. Nevertheless, it can be seen that the splines defined in (2-6), (2-13) and (2-16) satisfy conditions *i* and *ii* for appropriately chosen spaces and operators. The following definition of a spline algorithm is taken from [58].

DEFINITION 3.4. *An algorithm φ^s is called a spline algorithm if it is of the form $\varphi^s(N(f)) = S\sigma(N(f))$.*

The spline algorithm may take many different forms according to the problem definition. If $T(F_1)$ is a Hilbert space, and $T(Ker\ N)$ is closed, then φ^s is a linear algorithm and takes the form,

$$\varphi^s(N(f)) = \sum_{i=1}^{n} L_i(f)\, S(g_i), \tag{3-9}$$

where L_i are the information functionals that make $N(f) = [L_1(f), L_2(f), \ldots, L_n(f)]$ and g_i are the splines interpolating the unit vector $[0, \ldots, 1, \ldots, 0]^{\mathrm{T}}$, for $i = 1, \ldots, n$. Spline algorithms in Hilbert space settings have been studied in [4, 5, 10, 11, 37, 38, 39, 41, 58, 69]. The following optimality result holds [39, 58].

THEOREM 3.1. *Let $T(\tilde{F}_1)$ be a Hilbert space, and let $T(Ker\ N)$ be closed. Then the spline algorithm (3-9) always exists, is unique and its error is equal to the radius of information.*

Spline algorithms are well behaved in terms of their error properties even in a non-Hilbert space setting, or when the problem is not linear. In that case, the error of the algorithm is larger than the radius of information, but it can be proven [57, 72] that,

$$e(\varphi^s, N) \leq 2\ r(N, f). \tag{3-10}$$

There are many papers dealing with the theoretical and practical aspects of splines. The optimality of spline algorithms was initially realized by Schönberg in [49]. Optimal spline algorithms have been developed for many different problems and formulations [1, 9, 15, 16, 17, 18, 41, 69].

3.5 The average case model.

The worst case model has been extensively discussed in this section, since it will be used for the applications that will be presented in the following sections. Average and probabilistic models have also been studied [65, 66, 67, 68, 70, 72, 73]. They differ from the worst case in the way errors and costs are measured. We will briefly discuss the average case model to illustrate the important differences in the definitions.

In the average case, the error and the cost of an algorithm is measured for the *average* function f and not for the worst possible one. This is a plausible setting if we are interested in the expected behavior of the algorithm, or if we can assume that the worst possible function will be encountered with a very low probability. The space F_1 will now be equipped with a probability measure μ which will measure the frequency of occurrence of any elements from F_1.

The choice of the measure is not obvious and requires extensive consideration and study of the properties of the problem. This choice is as important in the attempt to correctly model the problem, as is the choice of the norm in the worst case.

The *average error* of the algorithm is given by,

$$e^{\text{avg}}(\varphi, N) = \int_{F_1} \|S(f) - \varphi(N(f))\|\ \mu(df). \tag{3-11}$$

The average radius of information can now be defined accordingly. Many of the results that hold in the worst case setting have been shown to hold in the average (see [57, 65, 66, 67, 70].) Also, there are relations between the average case setting and Bayesian statistics [32, 62, 63].

3.6 Advantages of information-based complexity.

Information-based complexity provides methods for the derivation of optimal algorithms, optimal information, upper and lower bounds on the error of the solution of a problem, and upper and lower bounds on its complexity. These issues are of great interest to the study of problems requiring approximate solutions. Problems have to be solved approximately if the available information is partial. Even when complete information is available we may choose to solve approximately if the cost of an exact solution is prohibitively high.

There are many computer vision problems that have to be solved approximately. These include surface interpolation from sparse depth and orientation measurements, motion recovery, edge detection etc. We will review some of these problems in the following sections and discuss methods that can be used for their solution. Information-based complexity is one of these methods.

All of the methods that will be discussed in sections 4, 5 and 6 investigate the existence and uniqueness of the solution, but not all of them address issues of optimality and complexity. When these issues are important to the analysis of the problem, it is suggested that information-based complexity is employed.

4. Modelling ill-posed problems.

Consider the following optics problem. Given a 3-dimensional object, construct a 2-dimensional image of this object. This problem is easily solved if we assume that we know certain parameters of the imaging system. In this case the solution always exists and is unique.

Computer vision deals with the *inverse* problem of recovering a 3-dimensional object from its 2-dimensional image. It can be immediately seen that this is a more difficult problem, because during the imaging process the information contained in the third dimension is lost. In other words, the imaging process is in general not one to one, which means that there are many objects that could create the same image. Also note that the image is not continuous but a discrete sample of scene intensities. Thus the recovery process becomes even more difficult.

4.1 What is an ill-posed problem?

Hadamard [21] gave the following definition of a well-posed problem (see also [55].)

DEFINITION 4.1. *A problem is well-posed iff its solution,*

i. *Always exists.*
ii. *Is unique.*
iii. *Depends continuously on the input data.*

DEFINITION 4.2. *A problem is ill-posed iff at least one of the conditions in definition 4.1 does not hold.*

It can be seen from the discussion at the beginning of this section, that computer vision problems are usually ill-posed. This is, most of the time, due to the existence of many solutions. There are also certain problems that are ill-posed because their solution does not always exist, or does not depend continuously on the data. Some examples follow.

1. Edge detection.

In edge detection the boundaries of the imaged objects are reconstructed. One way of attacking this problem is to attempt to detect changes in the intensity of the image; this process is similar to differentiation. Therefore, the procedure that detects those changes can be seen as numerical differentiation. The problem is ill-posed because the solution does not depend continuously on the input data [44, 56].

2. Motion.

Assume that we want to recover the velocity of a point belonging to the contour of an object. The velocity vector has two components, one perpendicular to the contour and one parallel to it. Local measurements of the motion of points on the contour yield only the perpendicular component of the velocity field [23, 24]. The problem is ill-posed because there are many solutions that interpolate the data. The set of solutions contains all velocity vectors that have the perpendicular component equal to the sampled one and the one parallel to the contour equal to some arbitrary value.

3. Surface reconstruction.

The shape of a surface is recovered from different types of data, usually depth and orientation measurements. The problem is ill-posed because there are many surfaces that interpolate the given set of data. On the other hand, if we combine data given by more than one process, and if the data are perturbed, then the problem may be overconstrained and not have a solution [53].

There are many more examples of ill-posed vision problems. We briefly mention some :

(1) Shape from shading [27, 35].
(2) Optical flow [26, 33, 34].
(3) Structure from motion [59].
(4) Shape from texture [28].

A review can be found in [42, 43].

4.2 Restoring well-posedness – Regularization.

Regularization theory [54, 55] deals with the solution of ill-posed problems. The problems are restated, so that they become well-posed, and then solved. This requires the definition of the solution space, as well as certain constraints that the solution must satisfy. These constraints are usually functionals that have to be minimized.

Choosing the space of functions in which the solution and the function that created the data must lie, and choosing the functional to be minimized are difficult tasks. In particular, while we want our model to be mathematically correct, the physical plausibility of the solution is also important. Hence, the choice must be based on the physical properties of the problem, which must be modelled as closely as possible.

The functionals that provide the constraints are (semi-)norms of various kinds, or generalizations of (semi-)norms. For example :

(1) Norms or semi-norms on Hilbert spaces [10, 11, 36, 37, 38].
(2) The L_p norm over some operator T [58], i.e.

$$\|T(f)\|_{L_p} = \left[\int_{\mathcal{X}} |Tf(x)|^p \, dx\right]^{1/p} . \tag{4-1}$$

(3) The *Tikhonov stabilizers* [53, 55] given by,

$$\|u\|_p^2 = \sum_{m=0}^{p} \left[\int_{\mathbb{R}^d} \lambda_m(x) \sum_{j_1+\cdots+j_d=m} \frac{m!}{j_1!\ldots j_d!} \left(\frac{\partial^m u(x)}{\partial x_1^{j_1} \ldots \partial x_d^{j_d}} \right)^2 dx \right]. \tag{4-2}$$

Assume that the space of functions has been defined, and the functional has been selected. Let $\vec{y}$ be the data, obtained from a problem $S\vec{x}$, where S is a known operator. Also assume, that we do not require interpolation of the data, i.e. we permit $\vec{y} \neq S\vec{x}$. We want to retrieve $\vec{x}$. Then, the solution to the problem is obtained using one of the following three minimization methods, sometimes called *regularization methods* [43].

(1) Find $\vec{x}$ such that $\|T\vec{x}\| \leq c$ holds for some fixed c and such that $\|S\vec{x} - \vec{y}\|^2$ is minimized.

(2) Find $\vec{x}$ such that $\|S\vec{x} - \vec{y}\| \leq c$ holds for some fixed c and such that $\|T\vec{x}\|^2$ is minimized.

(3) Find $\vec{x}$ such that $\|S\vec{x} - \vec{y}\|^2 + \lambda\ \|T\vec{x}\|^2$ is minimized, for a given $\lambda \geq 0$.

What is the meaning of these three methods? Using (1) we obtain as close an approximation to the data as possible while maintaining a fixed "regularity". In (2) we get a solution that is as smooth as possible while maintaining a fixed distance from the data, and (3) is a compromise between (1) and (2). The value of λ determines the weight between the choices (1) and (2). For λ close to zero the method is similar to (1), while for large λ the method is close to (2). The appropriate choice of λ is not an easy problem. In section 5, we will discuss work where a choice of λ had do be done.

4.3 Solution of some ill-posed problems.

We now outline a method for the solution of some of the problems of section 4.1 (see [43].)

1. Edge detection.

As we said in the statement of the problem, we want to approximate the derivative f' of the function f. We choose $\|T(\cdot)\|$ to be : $\|Tf\| = \|f''\|_{L_2}$. We want to minimize,

$$\theta(f) = \sum_{i=1}^{n} \left(y_i - f'(x_i)\right)^2 + \lambda\ \|f''\|_{L_2}^2, \tag{4-3}$$

where $y_i = f'(x_i) + \epsilon_i$ are the given data, which we assume are perturbed by ϵ_i. The functional (4-3) corresponds to the third regularization method.

2. Motion.

We want to retrieve the velocity of a point, lying on the contour of the image of a moving object, only from the perpendicular component of the velocity vector. The parallel velocity component cannot be measured (See Fig. 2.)

To turn the problem into a well-posed one, we choose $\|T(\cdot)\|$ to be : $\|Tv\| = \|\frac{\partial v}{\partial s}\|_{L_2}$, where s is the direction parallel to the contour. We want to minimize $\|Tv\|^2$. This is a functional

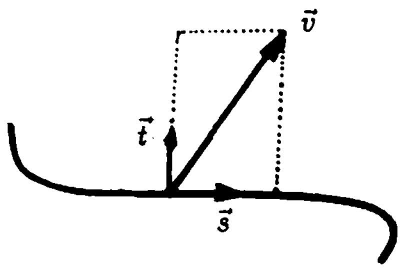

Figure 2.

measuring the smoothness of the velocity field in the s-direction. The minimization of $||Tv||^2$ corresponds to the second regularization method.

If the data are noisy, we can choose to use the third regularization scheme. We must then minimize $\theta(u) = \sum \left(\frac{\partial u}{\partial t}(s_i, t_i) - u_i\right)^2 + \lambda \, ||Tv||^2$.

3. Surface reconstruction.

There are many different methods used in recovering a surface from depth and orientation data. The conventional ones involve minimizing a simpler form of the norm given by (4-2), which corresponds to the second regularization scheme. There are many other methods, which will be described in the next section.

5. Survey of surface reconstruction methods.

In section 4 we described ill-posed problems and we outlined work that has been done in formulating them as well-posed. One of these problems is that of surface reconstruction, which we discuss in this section. There are many reasons behind our choice of treating this particular topic in a more extensive manner.

(1) Surface reconstruction is one of the major problems in computer vision. It is encountered as part of the object formation stage.
(2) There has been extensive work done on this problem using a variety of methods, both formal and heuristic, and many different types of information.
(3) It can be severely ill-posed. It does not usually have a unique solution. On the other hand, when the data come from many different sources, and are perturbed, they may overconstraint the problem. In this case there may not be any solution at all.[5]

We will now proceed to discuss various different approaches taken towards the solution of the surface reconstruction problem [7, 8, 29, 30, 34, 51, 52, 53].

[5]See discussion about surface reconstruction in the previous section.

5.1 Smoothness stabilizers and multigrid computation.

In a series of papers [19, 20, 51, 52, 53] the regularization methods developed in [54, 55] have been used to make the surface reconstruction problem well-posed. In particular, the solution to the surface reconstruction problem has to minimize a series of energy functionals [19, 20, 52, 53].

Consider the generalized *smoothness functionals* of a function $u(x, y)$ in $\mathbb{R}^2$,

$$S_m(u) = \int_{\mathbb{R}^2} \sum_{i=0}^{m} \binom{m}{i} \left(\frac{\partial^m u}{\partial x^i \partial y^{m-i}} \right)^2 dx dy. \tag{5-1}$$

These functionals have been used in [11] as (semi-)norms of Hilbert spaces, in which case their minimization gives the solution to a reconstruction problem.

Terzopoulos [52] assumes a space of functions with square integrable second partial derivatives. He uses as the functional to be minimized a convex combination of two functionals of the form (5-1) for $m = 1$ and $m = 2$ respectively. In particular he uses

$$S_1(u) = \int_{\mathbb{R}^2} \left\{ \left(\frac{\partial u}{\partial x} \right)^2 + \left(\frac{\partial u}{\partial y} \right)^2 \right\} dx dy, \tag{5-2}$$

and

$$S_2(u) = \int_{\mathbb{R}^2} \left\{ \left(\frac{\partial^2 u}{\partial x^2} \right)^2 + 2 \left(\frac{\partial^2 u}{\partial x \partial y} \right)^2 + \left(\frac{\partial^2 u}{\partial y^2} \right)^2 \right\} dx dy. \tag{5-3}$$

These two functionals have a physical interpretation. $S_1(\cdot)$ measures the potential energy of deformation of a membrane under tension, and $S_2(\cdot)$ measures the bending energy of a thin plate of infinite extent. In other words, (5-2) permits a rough surface that might have discontinuous derivatives, thus it may crack and crease. This could be desirable when the reconstructed surface is not very smooth, which is the case when we want to reproduce discontinuities. On the other hand, (5-3) imposes smoothness to the surface that minimizes it. This can be very useful for smoothing noisy data. Terzopoulos argues that both (5-2) and (5-3) are very well suited to modelling real world surfaces. Further evidence on the applicability of those can be found in [8, 50] where it is argued that humans can perceive smoothness up to regularity 2. It is further argued in [52] that only one of them is not sufficient. In particular, real world surfaces tend to be smooth throughout their interior, except for certain localities, in the boundaries, where they are discontinuous. If (5-2) were adopted, the discontinuities would be modelled but the interior would be unreasonably rough. If only (5-3) were adopted, the surface interior would be modelled correctly, but the discontinuities would be smoothed over. Furthermore, the reconstruction would oscillate in the vicinity of the discontinuity.

Because of the above considerations, a combination

$$S(u) = (1 - \tau) S_1(u) + \tau\, S_2(u), \qquad \tau \in [0, 1] \tag{5-4}$$

has been proposed. For $\tau = 0$, $S(u) = S_1(u)$ and for $\tau = 1$, $S(u) = S_2(u)$. For any other value of τ the functional (5-4) permits a combination of the properties of the two functionals. We can choose whether to favor a smooth surface, or one that is less regular.

Terzopoulos further proposes to have τ vary across the domain, and by doing that he gives a variable character to $S(u)$. This way, sudden changes in the data across discontinuities can be followed closely while the rest of the surface can retain its smoothness. To obtain this effect of a variable τ on the same surface, he proposes τ to be a function of the variables x and y, which would take a value of 0 near the discontinuities, and a value of 1 away from them, while changing gradually from the one value to the other. The functional $S(u)$ with $\tau(x, y)$, as defined above, is called a *controlled smoothness stabilizer*. Unfortunately, a method that can be used to obtain $\tau(x, y)$ is not given.

In [52] the conditions that make the problem of recovering $S(u)$ well-posed are given. Furthermore, it is argued that these conditions always hold in the proposed formulation, hence the solution exists and is unique. Unfortunately, this solution does not have a closed form. To overcome this, it is proposed in [51, 52] that the problem be transformed into a discrete one, by using a finite element method. The discretization process involves constructing an approximate variational principle and minimizing the discrete version of the energy functionals. The whole discrete problem can then be written down as a system of linear equations, which can be solved to yield a solution.

To speed up the process of solving the system of equations, a multigrid approach is proposed. The process starts with a coarse discretization, which results in a small system of equations. A few iterations, of a method that solves this system, are performed. Then, a finer discretization is made and the corresponding system is constructed. This system is solved with the same iterative method as before, using as an initial approximation the solution obtained from the *coarse* phase. Again few iterations are performed. This process is repeated until the grid is sufficiently fine, and then the system is solved to the required level of accuracy.

5.2 Shape from stereo.

Lee [34, 35] uses the framework of information-based complexity discussed in section 3 to model a problem of surface reconstruction from sparse depth data. The data are given in the form of an information vector $N(f) = [f(x_1, y_1), \ldots, f(x_n, y_n)]^{\mathrm{T}}$, and the space of functions is taken to be $F = \{f : D \subseteq \mathbb{R}^2 \longrightarrow \mathbb{R},\ f$ has square integrable second partial derivatives$\}$. The space F is equipped with the semi-norm,

$$\|f\| = \int_D \left\{ \left(\frac{\partial^2 f}{\partial x^2}\right)^2 + 2\left(\frac{\partial^2 f}{\partial x \partial y}\right)^2 + \left(\frac{\partial^2 f}{\partial y^2}\right)^2 \right\} dxdy. \qquad (5\text{-}5)$$

Lee chooses to work on a restriction F_0 of F where $F_0 = \{f \in F : \|f\| \leq 1\}$. The optimal algorithm in this setting is the spline algorithm which is given by,

$$\varphi^s(N(f)) = \sum_{i=1}^{n} f(x_i, y_i)\, \sigma_i(x, y), \qquad (5\text{-}6)$$

where $\{\sigma_i\}_{i=1,\dots,n}$ are the basis splines that were defined in section 3, and which must satisfy $\sigma_i(x_j, y_j) = \delta_{ij}$, δ_{ij} being the Kronecker delta.

This formulation, proposed in [29, 30, 33, 34], yields a well-posed problem. It has been shown [10, 37, 39, 58] that the solution exists and is unique. It can also be seen that the solution depends continuously on the input data, although this point is not explicitly discussed in Lee's work. This work is an example where well-posedness is achieved as a necessary step towards the optimal solution. Therefore, it can be seen that an additional process that achieves well-posedness is not needed.

To calculate the optimal spline function, two approaches are proposed.

In the first, the *basis splines* approach, the coefficients of the basis splines are precomputed and stored. Basis splines are the splines interpolating the basis $(0, \dots, 1, \dots, 0)$,[6] of the space of permissible data vectors. They are constructed by using the reproducing kernel of the space F. The reproducing kernel $K(x, y; s, t)$ of a Hilbert space has the property $\langle K(\cdot,\cdot; s, t), f(\cdot,\cdot)\rangle = f(s, t)$. It has been used in many applications [4, 5, 10, 11, 36, 37, 38]. (See also section 2.)

Then, given the data $f(x_i, y_i)$, the spline algorithm is constructed as a linear combination of the basis splines. Using this procedure the spline algorithm φ^s is very easy and fast to construct. The only problem that might be encountered is lack of sufficient storage space for the coefficients of σ_i, especially when the cardinality of the sampling is high.

In the second approach the problem is remodelled in a *quotient reproducing kernel* setting similar to the one developed in [36, 37]. In this approach the space F is decomposed into two spaces F_H and F_N, so that $F = F_H \oplus F_N$. F_N is the null space of the semi-norm $\|\cdot\|$, created in our case by $\text{lin}\{1, x, y\}$, and F_H is now a Hilbert space with norm $\|\cdot\|$. Then, the reproducing kernel of F_H is given with respect to the reproducing kernel of F (see [34, 36] and also [8].) It is given by,

$$\begin{aligned} K_{F_H}(x, y; s, t) &= K_F(x, y; s, t) \\ &\quad - \sum_{j=1}^{d} q_j(x, y) K_F(x_j, y_j; s, t) - \sum_{j=1}^{d} q_j(s, t) K_F(x, y; x_j, y_j) \\ &\quad + \sum_{j=1}^{d}\sum_{i=1}^{d} q_i(x, y) q_j(s, t) K_F(x_i, y_i; x_j, y_j), \end{aligned} \tag{5-7}$$

where $K_F(x, y; x_i, y_i) = \left[(x - x_i)^2 + (y - y_i)^2\right] \log\left[(x - x_i)^2 + (y - y_i)^2\right]$, $d = 3$, $q_1 = 1$, $q_2 = x$, $q_3 = y$, and $\{(x_j, y_j)\}_{j=1,2,3}$ are any arbitrary non-colinear sample points.

The spline function under this formulation is given by,

$$\sigma(x, y) = \sum_{i=1}^{n-d} a_i K_{F_H}(x, y; x_i, y_i) + \sum_{i=1}^{d} f(x_i, y_i) q_i(x_i, y_i). \tag{5-8}$$

[6] Where 1 is located at the i-th position, $\forall\, i = 1, \dots, n$.

The coefficients a_i can be obtained by solving the system of equations,

$$\sum_{i=1}^{n-d} a_i K_{F_H}(x_j, y_j; x_i, y_i) + \sum_{i=1}^{d} f(x_i, y_i) q_i(x_i, y_i) = f(x_j, y_j), \qquad j = 1, \ldots, n. \tag{5-9}$$

The matrix $\mathbf{K} = \{k_{ij}\}_{i,j=1}^{n-d}$ is dense, symmetric and positive definite. For the solution of the system (5-9) Choleski factorization is used in [34] which yields the solution at a cost of $O(n^3)$.

5.3 Reproducing kernel splines in surface reconstruction.

Boult in [7, 8] extends the approach taken in [30, 33, 34]. He defines the space $D^{-2}L^2$ of functions with square integrable second partial derivatives, and equips this space with the semi-norm given by (5-5). This setting is similar to the one discussed in [3, 4, 10, 11] and implicitly assumed in [51, 52, 53]. In contrast to the work of Terzopoulos, a direct solution is obtained without the need to resort to discretization. In particular, the actual norm is minimized instead of its discrete version. Furthermore, discretization into a finite element space is not performed, hence, further errors are avoided.

The first problem addressed in [8] is *semi-reproducing kernel interpolation* of the data, where it is assumed that the data are exact, hence they are fitted. Then, the spline function is given by,

$$\sigma(x, y) = \sum_{i=1}^{n} a_i K(x, y; x_i, y_i) + a_{n+1} x + a_{n+2} y + a_{n+3}, \tag{5-10}$$

where $K(x, y; x_i, y_i) = \left[(x - x_i)^2 + (y - y_i)^2\right] \log\left[(x - x_i)^2 + (y - y_i)^2\right]$ is the semi-reproducing kernel of the space F with semi-norm (5-5).

The coefficients a_i can be obtained by solving the system of equations,

$$\sum_{i=1}^{n} a_i K(x_j, y_j; x_i, y_i) + a_{i+1} x_j + a_{i+2} y_j + a_{i+3} = f(x_j, y_j), \qquad j = 1, \ldots, n.$$

$$\sum_{i=1}^{n} a_i\, x_i = 0,$$

$$\sum_{i=1}^{n} a_i\, y_i = 0, \tag{5-11}$$

$$\sum_{i=1}^{n} a_i = 0.$$

The next problem addresed in [8] is *semi-reproducing kernel approximation*, where it is assumed that the data are noisy, hence they are smoothed over. In this approach, one tries to minimize the functional,

$$\theta(\cdot) = \lambda \parallel \cdot \parallel + \sum_{i=1}^{n} |L_i(\cdot) - L_i(f)|^2. \tag{5-12}$$

where the data are given by $N(f) = [L_1(f), \ldots, L_n(f)]^{\mathsf{T}}$.

Equation (5-12) is a compromise between closeness of fit to the data and smoothness, and the bias is determined by the value of the parameter λ. This value can be viewed as our belief in the quality of the data. If the data are good they should be approximated as close as possible, whereas if the data are noisy they should be smoothed. The spline algorithm that minimizes (5-12) and solves the approximation problem is given by (5-10). The coefficients a_i are obtained in a similar manner; only the system of equations is slightly different. In particular,

$$\lambda\, C + \sum_{i=1}^{n} a_i K(x_j, y_j; x_i, y_i) + a_{i+1}x_j + a_{i+2}y_j + a_{i+3} = f(x_j, y_j), \qquad j = 1, \ldots, n.$$

$$\begin{aligned} \sum_{i=1}^{n} a_i\, x_i &= 0, \\ \sum_{i=1}^{n} a_i\, y_i &= 0, \\ \sum_{i=1}^{n} a_i &= 0, \end{aligned} \tag{5-13}$$

where C is a constant depending on the choice of the model. Note that as $\lambda \longrightarrow 0$ the above system is the same as system (5-11), and the problem becomes similar to the interpolation problem.

Boult further extended the above work to include more general spaces of the form $D^{-m}L^p = \{f\colon D \subseteq \mathbb{R}^2 \longrightarrow \mathbb{R},\ f$ has p-th integrable m-th partial derivatives$\}$ which are then equipped with various (semi-)norms. He consequently proceeds into solving the problem in a manner similar to the one described above. With this approach, surfaces with different regularity properties can be modelled.

5.4 Other Approaches.

The methods that were discussed in this section give a good indication of the work done in the area of surface reconstruction. They are by no means the only work available. In particular, in a series of papers [12, 13, 60, 61] the problem of interpolating a set of noisy data has been addressed. This approach is similar to the reproducing kernel approximation discussed in section 5.3 and has been covered in section 2.6 where we discussed spline interpolation for noisy data. The minimization functional is given by (2-14) and the minimizing spline is given by (2-16). In many papers the spline that smoothes over the data is called the *smoothing spline*.

One interesting issue in the above approach, which is also relevant to the approaches of Grimson and Terzopoulos in [19, 20, 51, 52], is the choice of the parameter λ. One can choose λ heuristically. A more elaborate mathematical method is *cross-validation* which is based on the statistical interpretation of the data. The properties of the data are analyzed to give a value for λ. More information on cross-validation can be found in [32, 61, 64, 71].

There are many more methods used for reconstructing a surface, known under the name of *shape from ...* methods. We could not possibly cover all of them in this survey, but we mentioned some in section 4 and discussed others in section 5. The discussion has been restricted to these problems that can be solved using regularization theory and information-based complexity. To show that information-based complexity can be used for the solution of other computer vision problems, we will describe in the next section a new problem that is formulated and solved using the proposed methodology.

6. Shape from shadows.

In this section we will define and propose a solution to the *shape from shadows* problem. In this problem the shadows created by a light falling on a surface will be used to recover the surface itself.

Very little work has been done using shadows for the reconstruction of the surface shape. The only existing work we are aware of is the one proposed in [31] where the problem is solved using a relaxation method. A new approach to the solution of this problem is taken in [22]. That approach follows the ideas of the general theory discussed in section 3.

Shadows are a very strong piece of information for many reasons. The process that uses shadows is not affected by texture or by surface reflectance. Furthermore, the imaging system does not need a grey scale or color capabilities. It is sufficient to be able to distinguish between black and white. Also, noise in the form of bright spots inside a dark area or dark spots inside a light region can be filtered out easily. It is evident therefore that shadows yield a powerful tool to be used in the reconstruction process.

6.1 Formulation of the problem.

This problem is approached in [22] in the following way. A 1-dimensional slice of a 2-dimensional surface is approximated. A 1-dimensional slice can be seen as a function of one variable $f : B \subseteq \mathbb{R} \longrightarrow \mathbb{R}$ belonging in a space of functions F_0. Let,

$$F_0 = \{ f \mid f : [0,1] \longrightarrow \mathbb{R},\ f' \text{ absolutely cont.},\ \|f''\|_{L_2} \leq 1 \}, \tag{6-1}$$

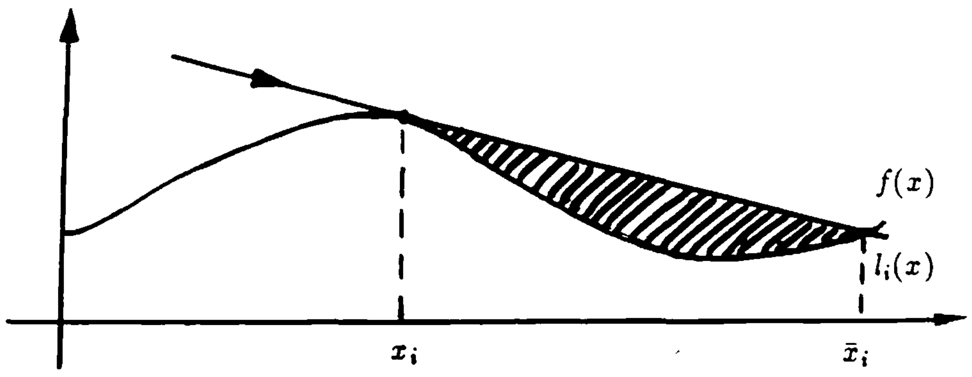

Figure 3.

be the space that contains the functions f that we want to approximate.[7,8] The norm $\|\cdot\|_{L_2}$ is defined as $\|f\|_{L_2} = \sqrt{\int_0^1 |f(x)|^2 dx}$.

Also, define the bilinear form $\langle\cdot,\cdot\rangle$ to be such that

$$\langle f, g\rangle = \int_0^1 f''(x)\, g''(x)\, dx, \tag{6-2}$$

and $\|\cdot\|$ to be such that

$$\|f\| = \langle f, f\rangle^{1/2}. \tag{6-3}$$

Clearly $\langle\cdot,\cdot\rangle$ defined above is a semi-inner product and $\|\cdot\|$ is a semi-norm. If we pose the additional constraints $f(0) = 0$ and $f'(0) = 0$, on the function f, then $\langle\cdot,\cdot\rangle$ becomes an inner product and $\|\cdot\|$ a norm. Consequently, F_0 equipped with $\langle\cdot,\cdot\rangle$ is a semi-Hilbert or a Hilbert space respectively.

The information contained in the shadows is derived as follows. Clearly, from the position of the light source, we can immediately obtain the derivative of the function f at the point x_i, x_i being the beginning of the shadow (see Fig. 3.) We can also obtain the difference, $f(x_i) - f(\bar{x}_i)$, between the two function values at the beginning and at the end of the shadow respectively. This difference can be obtained as follows. Denote by $l_i(x)$ the straight line defined by the light ray creating the shadow and passing through the points $x_i, \bar{x}_i$. We know the derivative of the line, $l_i'(x_i) = f'(x_i)$ and also that $f(x_i) - f(\bar{x}_i) = l_i(x_i) - l_i(\bar{x}_i)$. But $l_i(x_i) - l_i(\bar{x}_i) = l_i'(x_i)(x_i - \bar{x}_i)$ which is known, hence $f(x_i) - f(\bar{x}_i)$ is known.

An additional piece of information can be obtained from each shadowed area. It holds (Fig. 3) that,

$$f(x) < l_i(x), \qquad \forall x \in [x_i, \bar{x}_i]. \tag{6-4}$$

[7]The bound of 1 in $\|f''\|_{L_2}$ is assumed without loss of generality. Any fixed bound can be treated in a similar manner.

[8]The use of the interval $[0, 1]$ is not restrictive either. Any interval $[a, b]$, for some a and b. is equally good.

From this inequality we obtain data in the following manner. We select a few points t_j, $j = 1, \ldots, m_i$, in the i-th shadowed interval and we require that (6-4) is satisfied for these points. Clearly, $f(x_i) - f(t_j)$ is not known. We will assume that it is known and will derive its value later when we solve a minimization problem.

In each one of the images in our sample, there are 0, 1 or more shadowed areas. From each one of those shadowed areas we can obtain a derivative value, a displacement, and a set of unknown displacements satisfying (6-4). If we group all the data resulting from this sampling we obtain the vector,

$$N(f) = \big[f'(x_1), \ldots, f'(x_n), f(x_1) - f(\bar{x}_1), \ldots, f(x_n) - f(\bar{x}_n),$$
$$f(x_1) - f(t_1), \ldots, f(x_1) - f(t_{m_1}), \ldots, f(x_2) - f(t_{m_2}), \ldots, f(x_n) - f(t_{m_n})\big]^{\mathsf{T}}, \quad (6\text{-}5)$$

where m_i is the number of points t_i in the interval $[x_i, \bar{x}_i]$ that satisfy (6-4), and $m_1 + \cdots + m_n = m$.

6.2 Problem solution.

In the setting discussed in section 6.1, the spline algorithm is given by,

$$\varphi^s(x) = \sum_{i=1}^{2n} a_i g_i(x) + \sum_{j=1}^{m} c_j h_j(x), \qquad (6\text{-}6)$$

where $\{g_i\}_{i=1,\ldots,2n}$ and $\{h_j\}_{j=1,\ldots,m}$ are such that,

$$g_i''(x) = \frac{(x_i - x)_+^0 - (x_{i-1} - x)_+^0}{\sqrt{x_i - x_{i-1}}}, \qquad i = 1, \ldots, n \qquad (6\text{-}7)$$

where $(x_i - x)_+^0 = 1$ for $x_i > x$ and 0 otherwise,

$$g_{n+i}''(x) = (\bar{x}_i - x)_+ - (x_i - x)_+ - (x_i - x)_+^0(\bar{x}_i - x_i), \qquad i = 1, \ldots, n \qquad (6\text{-}8)$$

where $(x_i - x)_+ = x_i - x$ for $x_i > x$ and 0 otherwise, and

$$h_j''(x) = (t_j - x)_+ - (x_i - x)_+ - (x_i - x)_+^0(t_j - x_i), \quad i = 1, \ldots, n, \; j = 1, \ldots, m. \quad (6\text{-}9)$$

The functions $\{g_i\}_{i=1,\ldots,n}$ and $\{h_j\}_{j=1,\ldots,m}$ are the *representers* of the functionals $L_i(f)$ that make up the information $N(f)$, i.e. $\langle g_i, f\rangle = L_i(f)$. The coefficients a_i and c_j are chosen so that the spline φ^s interpolates the data, and minimizes the norm $\|\cdot\|$.

Two very important results about the spline algorithm (6-6) follow from theorem 3.1. First, the shape from shadows problem under the proposed formulation is well-posed. Regularization of the form discussed in sections 4 and 5 is not needed.

Second, the spline algorithm is optimal. If a problem is linear, the spline algorithm φ^s has a worst case error equal to the radius of information. An algorithm having error equal to the radius of information is called ***strongly optimal.*** If the problem is not linear, then (3-10) holds, and the algorithm is called ***almost strongly optimal*** [57].

The shape from shadows problem is linear, only if the cardinality of the second part of the information $N(f)$ is 0, i.e. $m = 0$. If $m > 0$ then φ^s is not strongly optimal, but almost strongly optimal.

6.3 Algorithm implementation.

The calculation of the spline algorithm given by (6-6) is done in 2 stages using the following procedure.

Stage 1:

First, assume that the cardinality of the non-linear part of the information is zero.[9] Therefore, ignore the coefficients c_j and only construct the values of the coefficients a_i. This is done by solving the system of equations,

$$\mathbf{G}\,\vec{a} = \vec{y}, \tag{6-10}$$

where $\mathbf{G} = \{< g_i, g_j >\}_{i,j=1}^{2n}$. The system is solved by a direct method without the need for pivoting since it is symmetric, positive definite and has a special structure that can reduce the number of calculations.

In the next step, the computed values of the a_i' s are used to construct the spline algorithm.

Third, we check to see whether the non-linear constraints are violated. This can be done on-line while $\varphi^s(x)$ is plotted, or can be done in a separate step.

If the non-linear constraints (6-4) are not violated, the process terminates. We have already obtained the approximation $\varphi^s(x)$ to the function f and we have plotted it. Since we have not used the non-linear part of the information, the problem is linear and the error of the spline algorithm is equal to the radius of information.

Stage 2:

If the constraints (6-4) are violated, then we do not have a sufficiently good approximation. Unfortunately, there is no a priori known formula for the calculation of c_j. Instead, $\|\varphi^s\|$ must be minimized directly by solving a quadratic minimization problem [22].

To construct c_j we proceed as follows. We take a few points from the shadowed intervals where the constraints are violated. Then, for these points we solve the minimization problem and obtain the coefficients c_j and also modify the old coefficients a_i. Then we check again for violations of the non-linear constraints. If there are violations we repeat Stage 2. We select a few more points from the interval(s) where (6-4) is violated, and we add them to the sample. The minimization is repeated for the new set of points and the new coefficients are derived. At the same time, the a_i' s and the old c_j' s are modified.

[9]This assumption is very plausible, and is supported by our experience. A good approximation can almost always be obtained without the need to use the non-linear information.

7. Conclusion.

We reviewed various mathematical methods that can be used to model and solve computer vision problems. Particular emphasis was placed on the surface reconstruction problem which is of significant importance to the scene recovery process.

We first discussed various classical methods used for the approximation and interpolation of functions. We considered methods that can be used to recover functions of one variable, and consequently extended the discussion to functions of two variables.

In section 3 we presented a survey of information-based complexity which deals with problems that have to be solved approximately. It provides methods for the approximate solution of these problems, and studies the error and the complexity of the solution. Many computer vision problems can be modelled and solved using this methodology.

We discussed ill-posed problems and analyzed the reasons that make computer vision problems ill-posed. Regularization theory has often been used to model computer vision problems. We showed, in section 4, some ill-posed problems that can be solved using the methods proposed by regularization theory.

In section 5 we reviewed different solutions to the surface reconstruction problem. Some of the solutions have been obtained using regularization theory, and some using information-based complexity.

Finally we discussed, in section 6, the shape from shadows problem, a problem which has not been tackled until recently. We formulated it in the theoretical framework of section 3, and we proposed a solution which always exists, is unique and achieves minimum error. We can thus suggest that information-based complexity may be employed to solve other open computer vision problems requiring approximate solutions.

References

[1] Ahlberg J. H., Nilson E. N., and Walsh J. L., "The theory of splines and their applications," Academic Press, 1967.

[2] Anselone, P. M., and Laurent, P. J., *A general method for the construction of interpolating or smoothing spline functions*, Nummer. Math. **12** (1968).

[3] Atteia, M., *Fonctions spline généralisées*, C.R. Acad. Sci. Paris **261** (1965).

[4] ———, *Etude de certains noyaux et théorie des fonctions "Spline" en analyse numérique*, Universite Grenoble, Thèse (1966).

[5] ———, *Fonctions spline et noyaux reproduisants d' Aronszajn-Bergman*, Revue d' Informatique et Reserche Operationel **4e année, R-3** (1970).

[6] Bernstein, S. N., *Démonstration du théorème de Weierstraß fondée sur le calcul de probabilité*, Proc. Kharkow Math. Soc. **13** (1912).

[7] Boult, T., *Reproducing kernel for visual surface interpolation*, Computer Science dep., Columbia University, Tech. Rep. (1985).

[8] ———, *Information-based complexity in nonlinear equations and computer vision*, Computer Science dep., Columbia University, Ph.D. Thesis (1986).

[9] DeBoor, C., "A practical guide to splines," Springer Verlag, 1978.

[10] Duchon, J., *Interpolation des fonctions de deux variables suivant le principe de le flexion des plaques minces*, R.A.I.R.O Analyse Numerique **10** (1976).

[11] ———, *Splines minimizing rotation-invariant semi-norms in Sobolev spaces*, in "Constructive Theory of Functions of Several Variables," Springer Verlag, 1977.

[12] Dyn N., Levin D., and Rippa S., *Surface interpolation and smoothing by "Thin Plate" splines*, in "Approximation Theory IV," Academic Press, 1983.

[13] Dyn, N., and Wahba, G., *On the estimation of functions of several variables from aggregated data*, SIAM J. Math. Anal. **13, No 1** (1982).

[14] Franke, R., *A critical comparison of some methods for the interpolation of scattered data*, Naval Postgraduate School, Tech. Rep. (1979).

[15] Gaffney, P. W., *Optimal interpolation*, Oxford University, D. Phil. Thesis (1976).

[16] ———, *The range of possible values of $f(x)$*, Computer Science and Systems Division, AERE, Harwell, Tech. Rep. (1977).

[17] ———, *To compute the optimal interpolation formula*, Computer Science and Systems Division, AERE, Harwell, Tech. Rep. (1977).

[18] Gaffney, P. W. and Powell, M. J. D., *Optimal interpolation*, in "Numerical Analysis, Lecture Notes in Mathematics," Springer Verlag, 1976.

[19] Grimson, W. E. L., "From images to surfaces : A computational study of the human early visual system," MIT Press, 1981.

[20] Grimson, W. E. L., *An implementation of a computational theory of visual surface interpolation*, Computer Vision, Graphics, and Image Processing **22** (1983).

[21] Hadamard, J., *Sur les problèmes aux derivées partielles et leur signification physique*, Princeton Univ. Bulletin **13** (1902).

[22] Hatzitheodorou, M. G., *Deriving shape from shadows. A Hilbert space setting*, Computer Science dep., Columbia University, Tech Rep. (1987).

[23] Hildreth, E. C., "The measurement of visual motion," MIT Press, 1984.

[24] ———, *Computation of the velocity field*, Proc. R. Soc. Lond. B **221** (1984).

[25] Holmes, R., *R-splines in Banach spaces : I. Interpolation of linear manifolds*, J. Math. Anal. Appl. **40** (1972).

[26] Horn, B. K. P., and Schunk, B. G., *Determining optical flow*, Artificial Inteligence **17** (1981).

[27] Ikeuchi, K., and Horn, B. K. P., *Numerical shape from shading and occluding boundaries*, Artificial Inteligence **17** (1981).

[28] Kender, J., *Shape from texture : A brief overview and a new aggregation transform*, Proceedings of the ARPA Image Understanding Workshop (1978).

[29] Kender, J., and Lee, D., *The information centered approach to optimal algorithms applied to the* 2 1/2-*D Sketch*, Proceedings of the ARPA Image Understanding workshop (1984).

[30] Kender, J., Lee, D., and Boult, T., *Information-based complexity applied to optimal recovery of the* 2 1/2-*D sketch*, Proceeding of the IEEE Conference on Representation and Control (1985).

[31] Kender, J., and Smith, E., *Shape from darkness. Deriving surface information from dynamic shadows*, Proceedings AAAI (1986).

[32] Kimeldorf, G. S., and Wahba, G., *A correspondence between Bayesian estimation on stochastic processes and the smoothing by Splines*, The Annals of Mathematical Statistics **41-2** (1970).

[33] Lee, D., *Optimal algorithms for image understanding*, Journal of Complexity **1** (1985).

[34] ———, *Contributions to information-based complexity, image understanding and logic circuit design*, Computer Science dep., Columbia University, Ph.D. Thesis (1986).

[35] ———, *Algorithms for shape from shading and occluding boundaries*, (to appear) (1987).

[36] Mansfield, L. E., *On the optimal approximation of linear functionals in spaces of bivariate functions*, SIAM J. on Num. Anal. **8** (1971).

[37] Meinguet, J., *Multivariate interpolation at arbitrary points made simple*, Journal of Applied Mathematics and Physics **30** (1979).

[38] ———, *Surface spline interpolation; Basic theory and computational aspects*, Inst. de Mathematique Pure et Appliquee, Univ. Catholique de Louvain Rapport No 35 (1983).

[39] Michelli, C. A., and Rivlin, T. J, *A survey of optimal recovery*, in "Optimal Estimation in Approximation Theory," Plenum Press, 1977.

[40] ———, *Lectures on optimal recovery*, in "Lecture Notes in Mathematics 1129, Numerical Analysis," Springer Verlag, 1984.

[41] Michelli, C. A., Rivlin, T. J., and Winograd, S., *The optimal recovery of smooth functions*, Numer. Math. **26** (1976).

[42] Poggio, T., *Computer vision*, Artificial Inteligence Lab., MIT (1986).

[43] Poggio, T., and Torre, V., *Ill-posed problems and regularization analysis in early vision*, Artificial Inteligence Lab., MIT (1984).

[44] Poggio, T., Voorhees, H., and Yuille, A., *Regularizing edge detection*, Artificial Inteligence Lab., MIT (1984).

[45] Powel, M. J. D., "Approximation theory and methods," Cambridge University Press, 1981.

[46] Prenter, P., *Lagrange and Hermite interpolation in Banach spaces*, J. Approx. Theory **4**, No 4 (1971).

[47] ———, "Splines and variational methods," John Wiley and Sons.

[48] Ralston, A., and Rabinowitz, P., "A first course in numerical analysis," McGraw Hill, 1978.

[49] Schoenberg, I. J., *On best approximations of linear operators*, Nederl. Akad. Wetensch. Indag. Math. **67** (1964).

[50] Späth, H., "Spline-Algorithmen zur Konstruktion glatten Kurven und Flächen," R. Oldenbourg Verlag, 1973.

[51] Terzopoulos, D., *Multilevel computational processes for visual surface reconstruction*, Computer Vision, Graphics and Image Processing **24** (1983).

[52] ———, *Multiresolution computation of visible surface representation*, Dep. of Electr. Eng. and Comp. Science, Massachusets Inst. of Technology, Ph.D. Thesis (1984).

[53] ———, *Controlled smoothness stabilizers for the regularization of ill-posed visual problems involving discontinuities*, Proceedings of the ARPA Image Understanding workshop (1984).

[54] Tikhonov, A. N., *Solution of incorrectly formulated problems and the regularization method*, Soviet Math. Dokl. **4** (1963).

[55] Tikhonov, A. N., and Arsenin, V. Y., "Solutions of ill-posed problems," V.H.Winston and Sons, 1977.

[56] Torre, V., and Poggio, T., *On edge detection*, Artificial Inteligence Lab., MIT (1984).

[57] Traub, J. F., Wasilkowski, G., and Woźniakowski, H., "Information, uncertainty, complexity," Addison-Wesley, 1983.

[58] Traub, J. F., and Woźniakowski, H., "A general theory of optimal algorithms," Academic Press, 1981.

[59] Ullman. S., *Maximizing rigidity : The incremental recovery of 3-D structure from rigid and rubbery motion*, Artificial Inteligence Lab., MIT (1983).

[60] Wahba, G., *Practical approximate solutions to linear operator equations when the data are noisy*, SIAM J. Numer. Anal. **14**, No 4 (1977).

[61] ——, *Surface fitting with scattered noisy data on Euclidian D-space and on the sphere*. Rocky Mountain Journal of Mathematics **14**, No 1 (1984).

[62] ——, *Cross-validated spline methods for the estimation of multivariate functions from data on functionals*, Proc. 50th Anniv. Conference Iowa Statistical Lab., The Iowa State Univ. Press (1984).

[63] ——, *Design criteria and eigensequence plots for sattelite-computed tomography*, Journal of Atmospheric and Oceanic Technology **2**, No 2 (1985).

[64] Wahba, G., and Wendelberger, J., *Some new mathematical methods for variational objective analysis using splines and cross-validation*, Montly Weather Review **108** (1980).

[65] Wasilkowski, G. W., *Local average error*, Computer Science dep., Columbia University, Tech. Rep. (1983).

[66] ——, *Average case optimality*, Journal of Complexity **1** (1985).

[67] ——, *Optimal algorithms for linear problems with a Gaussian measure*, Rocky Mt. J. Math. (1986).

[68] ——, *Information of varying cardinality*, Journal of Complexity **2** (1986).

[69] Wasilkowski, G. W., and Woźniakowski, H., *Optimality of spline algorithms*, Computer Science Dep., Carnegie-Mellon Univ., Tech. Rep. (1978).

[70] ——, *Average case optimal algorithms in Hilbert spaces*, Journal of Approximation Theory **47** (1986).

[71] Wendelberger, J., *Smoothing noisy data with multidimensional splines and generalized cross-validation*, Department of Statistics, University of Wisconsin-Madison, Ph. D. thesis (1982).

[72] Woźniakowski, H., *A survey of information-based complexity*, Journal of Complexity **1** (1985).

[73] ——, *Information-based complexity*, Annual Review of Computer Science **1** (1986).