Information Based Complexity Applied to Optimal Recovery of the 2 $^1/_2$-D Sketch.

John R. Kender*, David Lee# and Terrance Boult*

CUCS-170-85

***Department of Computer Science, Columbia University, New York, NY 10027**

#AT&T Bell Laboratories, 600 Mountain Ave, Murry Hill, NJ 07974

0. Abstract

In this paper, we introduce the ***information based complexity*** approach to optimal algorithms as a paradigm for solving image understanding problems, and obtain the optimal error algorithm for recovering the "$2^1/_2$–D Sketch" (i.e. a dense depth map) from a sparse depth map. First, we give a interpolation algorithm that is provably ***optimal*** for surface reconstruction; furthermore the algorithm runs in linear time. Secondly, we show that adaptive information (i.e. the intelligent and selective determination of where to sample next, based on the values of previous samples) can ***not*** improve the accuracy of reconstruction. Third, we discuss properties of an implementation of the algorithm which make it very amenable to parallel processing, and which allow for point-wise determination of surface depth without the necessity for global surface reconstruction. We conclude with some remarks on a serial implementation.

*This work supported in part by DARPA grant N00039-84-C-0165 and in part by NSF grant MCS-782-3676

1. Introduction

The calculation of a full depth map of a scene from information present in an image is a central problem in image understanding. In general, what is desired in the full depth map is some "best" surface (the $2^1/_2$–D Sketch) that fits the sparse and errorful depth data derived from shading, binocularity, motion, texture, and other "shape-from-x" surface cues. Mathematically, this can be cast as an interpolation/approximation problem subject to some error criterion. For the sake of simplicity we shall assume throughout this paper that we require the $2^1/_2$–D Sketch to *interpolate* the given depth data, although they the can be applied to surface approximation as well. Much work has already been done on recovering the $2^1/_2$–D sketch, but generally in a heuristic and non-optimal fashion [Grimson 81, Ikeuchi 81, Terzopoulos 84]. This paper investigates the general problem from a different and relatively new but powerful viewpoint. We attempt to answer the following questions, which address the questions of accuracy, reliability, and efficiency:

1. What *algorithms* are provably optimal with respect to the accuracy of the constructed full depth map?

2. What *properties* does the optimal algorithm have? Do the properties lead to feasible, stable, and even parallelizable computation?

We address the first question in Section 2, and show that spline algorithms are optimal with respect to the worst case error criterion. In Section 3, we first show that adaptive information, which is seemingly much more powerful than nonadaptive information (and certainly more computationally complex), surprisingly does not improve accuracy performance. Thus one only has to seek for optimal information among the classes of nonadaptive information. In Section 4, we construct the spline algorithm. Spline algorithms are linear in their data, and hence favorable for parallel computations. We show in Section 5 that under certain conditions they are also well-behaved.

A note to the reader: In brief, our intention is to show how the problem of transiting from a sparse depth map to a full one can be cast in the framework of the theory of computational complexity and optimal algorithms. Once the problem is posed in the context and terminology of that field, the solution is a straightforward special case of several existing theorems. However, since the methods and terms of that subject are probably foreign to most vision researchers, we also take care in what follows to explicate, step-for-step, the reasoning behind the procedures, explaining the more abstract constructs in terms of the actual vision problem at hand. In what follows we adapt the General Theory of Optimal Algorithms [Traub 80] to the problem of depth computations, retaining much of the specialized notation but with running glosses. In part, our intention is to alert the vision community to the relevance of the research in this area, in the form of the powerful and sometimes surprising results it offers to image understanding. Additionally, we hope to exploit the theory further in later papers for other vision problems, such as optimal surface recovery from a monocular intensity array (optimal shape from shading).

2. What is the Optimal Algorithm?

The analysis proceeds in several steps. To begin, the problem must be restated as a problem of: classes of functions (here, of surfaces in three-dimensions), available information, and classes of algorithms. To find the optimal error approximation to the $2^1/_2$$-D$ sketch the following aspects must be quantified; we discuss each in turn:

1. The space of surfaces in which the reconstructed $2^1/_2$$-D$ sketch must lie.

2. The information available and the dependencies by which it is obtained.

3. The class of allowable algorithms.

4. The measure of error and the meaning of "optimal" algorithm.

5. The specification of splines and spline algorithms.

6. The optimality of spline algorithms.

2.1. Choosing the Space of Surfaces and Their Norms

From the purely mathematical point of view of Information Based Complexity it is sufficient for the space of solutions, referred to as F_1, to have the properties of a semi-Hilbert space. However, for the problem of reconstructing the $2^1/_2$$-D$ sketch we may want F_1 to have some more restrictive properties important to some model of the psychologically plausible surfaces. A number of possible spaces exist but for the sake of concreteness and familiarity, we shall let F_1, our real-world surfaces, be the set of all real-valued functions f defined on R^2, such that f and its first and second order partial derivatives all belong to $L_2(R^2)$ (this is the same class considered in [Terzopoulos 84]). That is, the class of real-world surfaces is smooth at least up to local curvature: their curvatures are square-integrable.

In particular, this class rules out any surfaces that are merely *piece-wise* continuous or differentiable. These two latter exceptions, unfortunately, rule out true occlusions (where depth is discontinuous) and true corners (where the derivative is). Thus, the world to be seen appears as if it were shrink-wrapped: corners are rounded and discontinuities papered over. Inasmuch as the surfaces of objects tend to be locally smooth, however, these appear to be reasonable assumptions, and F_1 as it stands is sufficiently rich for purposes of the general theory. Note that the theory permits other possible classes, some of which do allow the surfaces to have discontinuities, as long as they are on sets of zero measure [Duchon 76].

We attempt to "see" a subimage of these real-world surfaces in F_1. They are supported in a finite region D of the xy plane; visually speaking, it is the xy plane that forms the background, and D that forms a finite subimage. For simplicity we assume that D is a closed compact simply connected region. Then the class of surfaces F_2 that we want to recover is the restriction of the surfaces in F_1 to D:

$$F_2 = \{ f|D : f \in F_1 \} \tag{1}$$

2.2. Quantifying the Idea of Information

To recover the surface (a member of F_2), we start with samples of depth data (in region D) ; these samples we would normally call ***information***. More precisely, information is defined in the general theory as a ***function*** of the following form.

$$N: F_1 \rightarrow R^k$$

That is, each type of information function is a mapping from the *class* of given surfaces to k-vectors of image primitive values. Each f (each real-world surface) in F_1 is a member of the domain of N; N(f) is the vector of samples. In terms of the vision problems, a given N(f) tells in what way a smooth surface, f, has been captured into a k-vector of extracted image primitives. (Thus, the theory uses "information" to make more precise the concept of "intrinsic image"; information can be velocities, surface orientations, brightness, etc.)

An important class of information is what is called linear information. Here the word "linear" refers only to certain properties of the information gathering process, and not to the underlying class of surfaces. The theories of information based complexity are most powerful when dealing with linear information. The notation employed therein is:

$$N(f) = [\, L_1(f), L_2(f), \ldots, L_k(f)], \quad i=1, \ldots, k. \tag{2}$$

where each L_i is a linear functional (that includes information like depth values, derivatives of the surface, integrals of the surface, gradients, etc. but not things like shading, most texture, etc..) For the case of depth values, which we have for the current problem, we simply have

$$L_i(f) = f(x_i, y_i), \quad i=1, \ldots, k. \tag{3}$$

One can easily check that each $L_i(\cdot)$ is a linear functional on F_1. For ease of analysis, we require that $k > 3$, and further, that the k depth values not be coplanar. This rules out the trivial case, i.e. when the interpolating surface is a plane.

Here N(f) is implicitly restricted to be a collection of samples taken "in parallel" from f at points that can be predetermined independently. That is, the i-th component of N(f) depends only on f, rather than on some dynamically changing sampling method based on the previous (i-1) components of N(f). In short, this information is *nonadaptive*. It is in contrast to the information used in many optimum-seeking algorithms (such as root-finding), which selectively sample promising areas increasingly more densely, based on their nearness to an optimum.

Superficially, an implicit restriction of N(f) to nonadaptivity seems to be a restriction to a less powerful set of information-gathering strategies. It will turn out, however, that it has absolutely no effect. Even if we extend the definition of information to allow the use of *any* adaptive sampling of linear functionals, no matter how intelligent, the intrinsic error is no less than that obtained by using appropriate nonadaptive information. (We will address this issue in Section 3.) Thus, what is important in terms of the general theory is simply that the information be linear, such as depth values of surfaces are. *Note that information like that of depth values or local gradient information is linear even when the underlying surface is very complex; the linearity restriction applies only to the information function and not to the surface function itself.*

2.3. Defining the Class of Algorithms

Knowing the information N(f), which is a k-vector of information samples (here, a vector of depth values), we now choose an *algorithm* ϕ to recover f in F_2 (here, that part of the real-world surface we attempt to "see"). The algorithm ϕ is defined in a very general way as a member of a class, Φ, of mappings:

$$\phi\colon R^k \to F_2, \;\; \phi \in \Phi. \tag{4}$$

Thus, in the general theory an algorithm maps information (of a function f from F_1) into a solution function (in F_2). Note that in this general definition, F_1 and F_2 are usually different classes. For example, in classical quadrature algorithms for numerical integration, F_1 is the class of functions to be integrated, N(f) are sample function values, ϕ is the quadrature formula, and F_2 is simply R^1, the set of reals. For the problem of reconstructing the $2^1/_2{-}D$ sketch, F_1 are real-world surfaces and F_2 are their restrictions to D.

2.4. Defining Algorithm Error, and Optimal Error Algorithm

In order to compare solutions and to measure the accuracy of ϕ, F_2 can be equipped with a norm, $||\cdot||_2$. By applying this norm, $|| f|D - \phi(N(f)) ||_2$ now quantifies the difference between $f|D$, which is that portion of the real-world surface to be recovered over D, and $\phi(N(f))$, which is the surface we construct by first gathering the information N and then applying the reconstructive algorithm ϕ.

There are many choices for $||\cdot||_2$. For our problem of depth interpolation, we may use the L_2-norm, defined as:

$$||f||_2 = \{ \int_D f^2 \; dx \, dy \}^{1/2} \tag{5}$$

Although this norm suffices, if we intend to simulate human judgment, we really would prefer a norm that is in some sense the most natural measure of algorithm error. To our knowledge there has been no work done in human psychology to indicate what is a truly "natural" scale of surface accuracy. However we need not worry; surprisingly it will turn out that the determination of the optimality of the algorithm is largely ***independent*** of the choice of a norm to measure error.

We are nearly ready to define the error of a given algorithm. Based on this definition, we will seek an algorithm that reconstructs the surface with minimum error. We would like to define the error of a particular algorithm in the worst case to be something like the following:

$$e(N,\phi,f) = \sup ||f^{*}|D - \phi(N(f))||_2 \tag{6}$$

where the supremum is taken over all f^{*} in F_1 that satisfy the same information. That is, the supremum should be taken over the set V(N,f), where

$$V(N,f) = \{ f^{*} \in F_1 | \;\; N(f^{*}) = N(f) \}. \tag{7}$$

This definition measures the distance between the actual computed solution, $\phi(N(f))$, and all functions f^{*} in F_1 that could possibly have been the source of the observed information. Since the f^{*} in V(N,f) are completely indistinguishable (we know nothing besides the information N(f)), we cannot tell which of them could have been the original function. Thus, we take the supremum to handle the worst case.

The problem with such a definition is that the class of functions in F_1 is usually too large. Given N(f), there are infinitely many interpolating f^{*} in V(N,f). Unless the F_2-norm is, in a sense, very weak, the above supremum and hence the worst case error may be very large, even infinite. However, in terms of the physics of the image understanding problem, many of these surfaces would also be physically impossible as well: some would have to pass through the camera itself; others would be impossible to fabricate under any known natural or artificial manufacturing process.

What we usually prefer instead is a solution function that comes as close as possible to "reasonable" members of V(N,f), rather than to all of them. Intuitively, a function may be considered "reasonable" if it satisfies some desirable side conditions. Mathematically, such a function is often characterized by expressing the desired properties in terms of another norm (or semi-norm), and defining "reasonable" to mean that this side norm is within certain bounds. We will denote the reasonableness norm by the F_4-norm, $||.||_4$. Just as there are many choices for the F_2-norm, the actual definition of the F_4-norm is determined by the problem. (It is an interesting psychological problem to find the most appropriate "reasonableness norm" for recovering the $2^1/_2{-}D$ sketch; to our knowledge this also remains an open problem [Boult 86].)

One example of a desirable property for functions in V(N,f) is smoothness. One way this can be quantified is by applying to elements of F_1 the quadratic variation semi-norm. This semi-norm is defined to be:

$$||f||_4 = \{\int_{R^2}[(f_{xx})^2+2(f_{xy})^2+(f_{yy})^2]\, dxdy\}^{1/2} \tag{8}$$

Given that second derivatives are closely related to surface curvature, this semi-norm has an appealing intuitive physical analogy: it measures the bending energy in an ideally thin and elastic plate which has been forced into the shape of f. The reader may note that this is the same semi-norm which appears in the work of [Grimson 81, Terzopoulos 84].

We can use this semi-norm (or any other "reasonable" one) to better define algorithm error. There are many ways to do so; one way would be to define a new space of functions $F_0 = \{ f \in F_1 : ||f||_4 \leq c \}$ for some arbitrary constant c. Then by restricting the interpolating surface to lie in the space F_0 we simply rule out any surface that is too "unreasonable". Now our definition of algorithmic error makes more sense :

$$e(N,\phi,f) = \sup ||f^{`}|D - \phi(N(f))||_2 \tag{9}$$

where the supremum is taken over all $f^{`} \in F_0$ satisfying the information N (that is, $f^{`} \in V(N,f) \cap F_0$). Importantly, one of the results of the general theory is that optimality is ***independent*** of the value of the constant c, used in the definition of F_0. [Traub 83].

Now we can define what an optimal algorithm is in the worst case model. The algorithm ϕ^* in Φ, the class of algorithms, is strongly optimal if and only if for each f in F:

$$e(N,\phi^*,f) = \inf \ e(N,\phi,f) \tag{10}$$

where the infimum is taken over all algorithms ϕ in Φ. That is, an algorithm ϕ^* is optimal if and only if the algorithm error from using ϕ^* (for any f in F_1) is no more than the algorithm error from using any other algorithm $\phi \in \Phi$.

Importantly, optimality is largely independent of how *error itself* is defined. The theory holds not only for our case (absolute error), but it also holds for relative error, or any other definition of error monotonically weighted by the "reasonableness" function, see [Traub 83] and below. This is good news from the stand point of psychology. Roughly it says that the optimal results are the same no matter how error is defined or perceived.

2.5. Spline Functions

We next define a particular interpolating function which is based on the reasonableness norm, $||\cdot||_4$. Although this function is derived from what appears to be only a problem-dependent side condition, it will be the primary function leading to the optimal solution of the interpolation problem.

Recall that V(N,f) is the class of all functions in F_1 that share the same information with f under the information extraction function N. In purely visual terms, it is the set of all surfaces defined on the infinite background that coincide with the depth data. Let $\sigma_{N(f)}$ be that member of V(N,f) with minimum F_4-norm. That is,

$$||\sigma_{N(f)}||_4 = \inf \ ||f^*||_4. \tag{11}$$

where the infimum is taken over all f^* in V(N,f). We call $\sigma_{N(f)}$ the spline function interpolating the data N(f). For the example problem it is not hard to show that such a spline function exists and is unique, if there are at least four non-coplanar data points [Duchon 76]. (For more details on the general spline algorithm, see [Traub 83] page 71.)

Thus, this unique spline function $\sigma_{N(f)}$ interpolates the information, and, because it minimizes the F_4-norm, of all such interpolants it is the most "reasonable". This need not imply that it is also the most "accurate". That is it need not necessarily minimize worst case error, since error is measured by a different norm and in a possibly more restricted space. Thus it may be surprising that under very general conditions, $\sigma_{N(f)}$ directly provides the optimally accurate algorithm, as we show below.

2.6. Spline Algorithms and their Optimality

Our last step is to define a class of algorithms based on the spline functions; this class will contain the optimally interpolating algorithm we seek.

The algorithm ϕ^s that takes information N(f), chooses the interpolating spline function $\sigma_{N(f)}$, and then restricts it to D, we call a spline algorithm. More precisely, the spline algorithm is defined as:

$$\phi^s(N(f)) = \sigma_{N(f)}|D, \tag{12}$$

where $\sigma_{N(f)}$ is the spline function. In our example with quadratic variation as semi-norm, the spline function is unique, so the spline algorithm is well defined.

The importance of ϕ^s is given by the following theorem.

Theorem. The spline algorithm is strongly optimal (in the worst case). That is,

$$e(N,\phi^s,f) = \inf\ e(N,\phi,f), \quad \forall f \in F_1. \tag{13}$$

where the infimum is taken over all ϕ in Φ. Further, the optimality of the spline algorithm is independent of the choice of the error norm $||\cdot||_2$.

For the detailed proof of the theorem, see [Traub 80], Theorem 5.1, page 76. The proof is based on the following two observations. First, it can be shown that the definition of the algorithm error as defined in Equation (9) is equivalent to many including absolute, relative and other definitions. This allows one to define a class of "reasonable" functions directly, and to measure error merely on an absolute basis, using the F_2-norm alone (for more detail see [Traub 83], Appendix E).

Secondly, the spline function $\sigma_{N(f)}|D$ can be shown to be the exact center of the set $V_4(N,f)|D$, and thus it must minimize the worst case error, where

$$V_4(N,f) = \{f^* \in V(N,f), ||f^*||_4 \leq c\}, \tag{14}$$

and the finite value c is arbitrary. (The class $V_4(N,f)$ contains those "reasonable" functions satisfying the given information.) The centrality is proven by showing that every $f^*|D$ in $V_4(N,f)|D$ can be expressed as the sum $\sigma_{N(f)}|D$ + h, where the properties of h are sufficient to show that the difference $\sigma_{N(f)}|D$ - h is also in $V_4(N,f)|D$. Intuitively, the spline function is found to be a type of "mean" about which all other information - satisfying surfaces are symmetrically placed; like other centers of symmetry, it minimizes worst case error.

Note that this spline ϕ^s is the same spline which is heuristically and *approximately* sought in [Grimson 81, Terzopoulos 84]. The primary importance of the theorem is that it shows this spline to be optimal. But additionally, given its development in the general theory, other known results can now be applied to it in a straightforward manner, with useful theoretic and practical effect as shown below.

3. Adaptive Information Does Not Help

In Subsection 2.2, we defined information as samples of depth data:

$$N(f) = [L_1(f), L_2(f), \ldots, L_k(f)], \quad i=1,\ldots,k. \tag{15}$$

where $L_i(f) = f(x_i, y_i)$, and $(x_i, y_i) \in D$. We call this nonadaptive information, since the i-th component of N(f), $L_i(f)$, depends only on f. Adaptive information, on the other hand, attempts to exploit whatever was learned while obtaining the (i-1) components of N(f). More precisely, adaptive information N^a is

$$N^a(f) = z = [z_1, \ldots, z_k] \tag{16}$$

where $z_i = L_i(f, z_1, \ldots, z_{i-1})$, $i=2,\ldots, k$. In the case of depth values,

$$z_i = f(x_i, y_i) \tag{17}$$

where $x_i = x_i(z_1,\ldots,z_{i-1})$ and $y_i = y_i(z_1,\ldots,z_{i-1})$, with $(x_i, y_i) \in D$.

The structure of adaptive information is much richer than nonadaptive information, and one might hope that by virtue of adaption some intelligence might determine the location for (x_i, y_i) on the basis of the results of the (i-1) prior samplings. Nevertheless, theory shows that, against intuition, for the class of linear information adaptive information cannot aid surface interpolation. For detailed discussions and proof, see [Traub 83] pages 57-62. The formal proof is based on the concept of ***radius of information***. Intuitively, the radius estimates the intrinsic error of the problem. For several classes of problems (including approximation), the radius cannot be reduced by adaptive strategies, heuristic or otherwise, in large part because there exist fixed but universal strategies. This is perhaps the strongest result of all: one cannot do better in collecting data than a simple snapshot does. Not only can the data be collected in parallel, it should.

(In fairness to existing research, and in accordance with common sense, it should be pointed out that this result does not imply that heuristics have no place in image understanding. It does, however, help delineate those areas where heuristics are useful, and perhaps required. If these non-linear information sources (e.g. intensity arrays, most texture, "high-level" knowledge, etc.) are allowed, heuristics may very well come into play. Note also that the results given above apply strictly to the recovery of a single "object":

given a region of the image, what will be approximated is a single smooth surface. Within that region, no heuristics are necessary. However, the determination of the proper boundaries of the region--algorithmically or heuristically--remains an open problem.)

At this point, we have shown that the spline algorithm is the optimal interpolation algorithm, and that nonadaptive information suffices. However, we have not attempted to optimize *where* to obtain the information itself. It is apparent that all sampling strategies are not equal. If we are free to select the location of information points, what points are optimal?

Restating this problem, suppose we are allowed to choose a k-vector of information samples: $N(f) = [f(x_1,y_1), \ldots, f(x_k,y_k)]$. Suppose, too, that no matter what information we select, we always use the optimal spline algorithm. Since the algorithm is tailored to the information, any error that would remain is intrinsically irreducible. We then can define optimal information, denoted by N^*, to be that information with the minimum intrinsic error.

Although in general the determination of optimal information is difficult, some limited results are known. In particular, it has been shown [Babenko 79] that for recovering a full depth map (with somewhat different F_2 and F_4 norms than those given here), the optimal choices for (x_i,y_i) can be shown to lie on a regular grid. More precisely, let the subimage D be the open rectangle: $(0, (n+1)h) \times (0, (n+1)h)$. Then the following information N^* is optimal (up to a constant factor) for surface recovery:

$$N^*(f) = [\ f(h,h),\ f(h,2h), \ldots, f(h,nh),\\ f(2h,h),\ f(2h,2h), \ldots, f(2h,nh),\\ \ldots,\\ f(nh,h),\ f(nh,2h), \ldots, f(nh,nh)] \qquad (18)$$

These are simply interior mesh points. Notice that the optimality of this information (and, of course the resulting error of the optimal algorithm using this optimal information) depends also on the norm $\|\cdot\|_2$; in general, however, the intrinsic error is monotonically decreasing in h. For a full proof of the optimality of this particular N^* mesh, and the exact specification for F_2 and F_4, see [Babenko 79].

4. Implementation of the Optimal Algorithm

In the previous sections we have shown the existence and uniqueness of the spline interpolating given depth data, and we have shown its optimality for surface recovery problems. In this section, we show how the spline functions can be constructed with the side condition of minimizing quadratic variation. Note that in what follows, we do not necessarily require optimal information; the depth samples can appear anywhere within the subimage D:

in a mesh, aligned on contours (such as those derived from zero crossings), clustered, or even at random.

There are many ways to implement the optimal algorithm; we are pursuing two methods based on the reproducing kernels of the space F_1. For brevity we shall discuss the details of only one. The interested reader may consult [Boult 85] for a discussion of both.

As is common in work with splines, a key step to their construction is the determination of the proper reproducing kernel; it differs with each different class F_1. It can be shown [Meinguet 83] that for our F_1 the appropriate reproducing kernel here is

$$K(x,y;u,v) = (1/16\pi) \times \{(x-u)^2 + (y-v)^2\} \times \log \{(x-u)^2 + (y-v)^2\}^{1/2}$$

Given the kernel, the spline (i.e. the reconstruction of the $2^1/_2$–D sketch) which interpolates depth data, $z = [z_1, \ldots, z_k] = [f(x_1,y_1), \ldots, f(x_k,y_k)]$ can be developed as:

$$\sigma_z = \sum_{i=1}^{k} \alpha_i K(x,y;x_i,y_i) + \beta_1 x + \beta_2 y + \beta_3 \tag{19}$$

where $\{\alpha_i\}$ and $\{\beta_i\}$ can be determined from the linear system of equations: (20)

$$\sum_{i=1}^{k} \alpha_i \, K(x_j,y_j;x_i,y_i) + \beta_1 x_j + \beta_2 y_j + \beta_3 = z_j,$$

$$\sum_{i=1}^{k} \alpha_i \, x_i = 0,$$

$$\sum_{i=1}^{k} \alpha_i \, y_i = 0,$$

$$\sum_{i=1}^{k} \alpha_i = 0.$$

From equations (21) and (22), it should be apparent that the splines are linear in the data. That is, if σ_1 interpolates information $z^{(1)}$, and σ_2 interpolates information $z^{(2)}$, then $c_1\sigma_1 + c_2\sigma_2$ interpolates information $c_1 z^{(1)} + c_2 z^{(2)}$. In terms of image understanding, this means that if two surfaces are superimposed so that their depths samples accumulate, than the super-position of the two full depth maps derived independently create a valid full depth map for the ensemble.

Since the spline algorithm is linear in depth data, it can be easily rewritten as the weighted sum of basis splines, as follows. Suppose that the information is merely i-th unit vector for R^k, that is, $N(f) = e_i = [0,\ldots,0,1,0,\ldots,0]$, where the unit is in the i-th coordinate position. This simpler information constraint is satisfied by a unique basis spline function σ_i, with the property that $\sigma_i(e_j) = \delta_{ij}$ (the Kronecker delta). In terms of the depth interpolation problem, σ_i generates a surface that has a value of 1 at sample point (x_i,y_i), is identically zero at all other sample points, and is smoothly rippled in all the space between, in order to minimize its bending energy. The spline interpolating any given depth data $z = [z_1,\ldots,z_k]$ then can be represented by the weighted sum of these individual basis splines, with z-values as the weights:

$$\sigma_z = \sum_{i=1}^{k} z_i \sigma_i. \tag{21}$$

Therefore the problem of computing σ_z may be decomposed into that of solving the k independent subproblems of computing each σ_i. Conceptually this is done by simply inverting the matrix given by (20); the elements of the i^{th} column of this inverse are the coefficients of σ_i. Depending on the physical imaging situation, this decomposition into basis splines can introduce powerful time savings. But in any case, the fact that the solution is a spline has several important consequences for implementation. We detail these below, together with those additional consequences that can result from exploiting the decomposition into basis splines.

1. Since the heart of the method is the solution of a system of linear equations, the depth interpolation problem reduces to a problem in numerical linear algebra, about which much is known. In general, running time on a serial machine will be $\approx {}^1/_3\, k^3$.

2. Given the coefficients $\{\alpha_i\}$ for the interpolating spline σ_z (whether obtained directly, or from precomputed basis splines), the desired interpolation points can be computed in parallel with a simple O(k) SIMD algorithm. Essentially, each SIMD processor can evaluate Equation (19) to recover the surface values for each desired interpolation point. (Of course, a serial implementation would take $O(n \times k)$ where n is the number of reconstruction points.)

3. Given the coefficients $\{\alpha_i\}$ for the interpolating spline σ_z, *any* individual value of the solution surface can be recovered locally, without the need for global recovery of the full surface: if just

one is needed, just one is computed. Likewise, if the system is asked to "focus" its attention on a particular area, the spline can be evaluated only at these extra points, without the need for increased accuracy (and solution) everywhere.

4. Suppose the location of the information can be fixed beforehand, as it could be with a laser range-finder or other direct depth-measuring devices. Then the coefficients of the basis spline functions σ_i can be precomputed and stored, since they depend only on the *location* of the information and not on the depth values themselves. The precomputation on a serial machine takes $O(k^3)$; this, however, is a one-time, off-line cost.

5. Given fixed locations and precomputed basis splines, the coefficients of any particular interpolating spline σ_z can be calculated using Equation (21), in $O(k^2)$ on a serial machine, or $O(k)$ on a k-processor SIMD machine.

6. Using basis splines (whether they are precomputed are not), the surface can be incrementally updated. Any sample value that changes over time has only a linear effect on existing interpolated data. This incremental update property is useful in some instances of motion understanding especially those involving articulated objects. The updated value at each point can be computed in parallel in constant time with a straightforward SIMD algorithm.

7. The $2^1/_2-D$ sketch computed by this algorithm is invariant under translation, rotation and scaling (independently in each direction).

5. Experimentation

These results suggest that it would not be hard to construct a special-purpose machine for surface interpolation that would be very quick and accurate. Using active imaging, it could obtain depth samples on a square grid of k total points, by ranging or by triangulation. The position of these sample points would remain fixed, so all coefficients could be precomputed. Run-time computation would entail only the distribution of input data and the calculation and collection of output data, using weighted sums of precomputed coefficients. Thus, the interpolated values at any point (x,y) are given by:

$$\sigma_z(x,y) = \sum_{i=1}^{k} z_i\,\sigma_i(x,y), \qquad (22)$$

where $z = [z_1,\ldots,z_k] = [f(x_1,y_1),\ldots,f(x_k,y_k)]$ and each σ_i has been previously precomputed. A SIMD algorithm evaluating Equation (22) would require about k multiplications, plus k units of local storage per process. If special purpose hardware were available, the data could be circulated in a type of toroidal systolic array. All output would be complete in roughly k cycles. If necessary, precomputation could be achieved in k parallel streams as well.

We have simulated much of this behavior on a standard uniprocessor. We briefly list some of our experimental results. They suggest there may be further algorithmic or computational efficiencies to be exploited.

1. If one uses a regular grid for the location of information, then the Gram matrix used to solve for the spline coefficients is highly regular. It is block Toeplitz, with each block itself being Toeplitz; thus, it contains only approximately $k/2$ distinct entries (rather than k^2). Further, an efficient solution is possible in only $O(k^{2.5})$ time; see [Rissanen 73].

2. If one uses a regular grid, the Gram matrix inverse (that is, the matrix of coefficients for the basis splines) is also highly regular. The basis splines require approximately $k^2/16$ units of storage, rather than k^2.

3. Experimentally, the Gram matrix derived from a regular grid of information locations appears to be rather well-conditioned with respect to computing its inverse, thus fairly large systems (e.g. a 100x100 matrix, corresponding to a grid of 10x10 depth data) will show only little loss of precision. An initial estimation of the condition number, for appropriately scaled grid-type data, is $\approx 19k^2$.

4. The stability of the surface solution appears to be critically dependent on the location of the information. In general, it appears that more closely spaced depth samples yield a less stable system, since small local depth changes then have greater effect on local smoothness.

5. Although the basis splines do not have compact support, their values appear to fall off rather rapidly. The speed of their asymptotic decay is roughly inversely proportional to the density of the information locations. That is, dense information means slower fall-off.

Some of these results are demonstrated in Figures 1 through 4.

Figure 1 shows one of the basis splines for the 10x10 regular grid. The calculation of the optimal basis splines is independent of the anticipated information values themselves; the splines are precomputable solely from their locations. The figure shows the basis spline which has been precomputed for one of the most central points, at $(x,y)=(5,5)$. This spline can also be viewed as the optimal solution for that surface whose information values polating the 16 data points given by:

0	4	4	0
0	4	4	0
0	4	4	0
0	4	4	0

This is an extended parabolic surface. Note how smooth the interpolation is, even with such sparse depth data. The reconstruction is done with 50x50 quadrilateral patches.

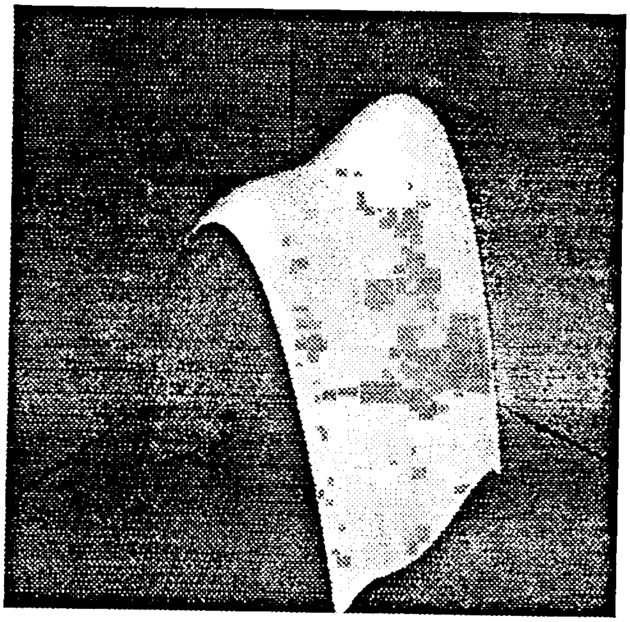

Figure 4

6. Related Problems in Computer Vision

The theoretic results reported above—the optimality and linearity of spline algorithms, the sufficiency of nonadaptive information, and others—apply to other vision problems that can be cast in the same framework. The theory requires that F_1 be an arbitrary linear space, and that the information be a vector of linear functionals on F_1. In terms of vision, the linearity of F_1 is rarely a problem, since object surfaces superimpose well; however, only some classes of image features can be considered to be linear information. The trouble is that linear information must also superimpose: the features derived from "sum" of two surfaces must be the sum of the features independently derived. This is often contrary to the laws of geometry, physics, and optics; for example, shading clearly does not sum well. Nevertheless, there are several important types of linear vision information, among which are:

1. The depth values themselves, at any place in the image: $L_i f = f(x_i, y_i)$. This is the problem just analyzed. Note that there

are zero at every grid point except at (5,5), where it has a value of 1. Note that it rapidly and smoothly falls off away from its peak, interpolating the zero data around it. Because of symmetries, the basis splines for (5,6), (6,5), and (6,6) are rotations of it; other basis splines look similar.

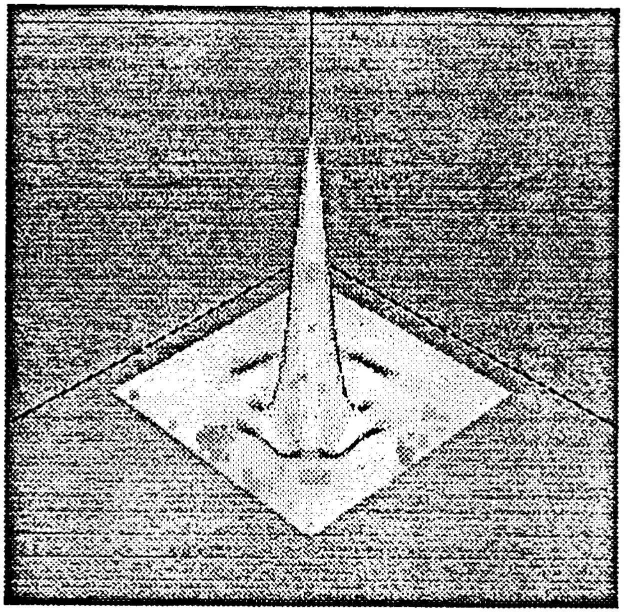

Figure 1

The following table gives the information samples used to generate figures 2 and 3:

9	9	9	6	6	6	3	3	3	0
9	9	9	6	6	6	3	3	3	0
9	9	9	6	6	6	3	3	3	0
6	6	6	6	6	6	3	3	3	0
6	6	6	6	6	6	3	3	3	0
6	6	6	6	6	6	3	3	3	0
3	3	3	3	3	3	3	3	3	0
3	3	3	3	3	3	3	3	3	0
3	3	3	3	3	3	3	3	3	0
0	0	0	0	0	0	0	0	0	0

The data is located on the same 10x10 regular grid used in Figure 1. The data is meant to form one-quarter of a four-layer wedding cake, with equal spacing between layers. (Thus the bottom layer is a one sample wide ring of zeros, the second is a three sample wide ring of threes, etc.. This is similar to objects interpolated in [Grimson 81].)

Figure 2 shows the interpolated surface for the quarter wedding cake. The Gram matrix is derived for the location information and is inverted, giving the coefficients of the 100 basis splines. The product of this inverse with the information values gives the 100 coefficients for the interpolating spline. In each dimension, the interpolation is four times as dense as the given information; thus the surface shown is derived from the dense grid of 41x41 points.

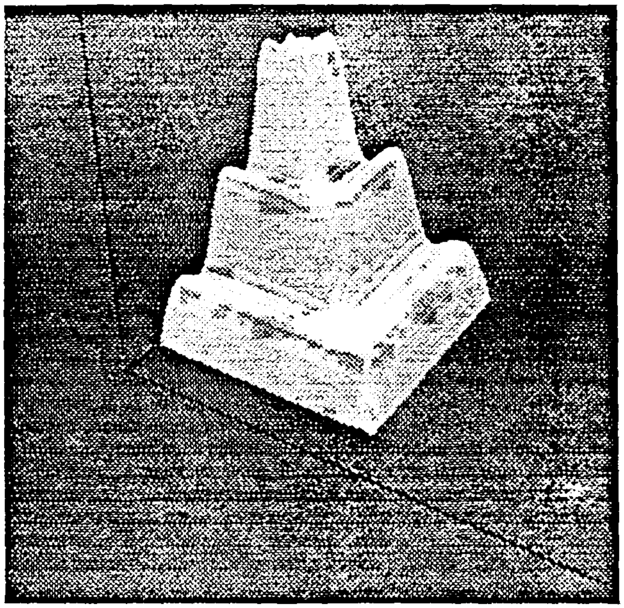

Figure 2

Figure 3 shows the same surface from a different perspective, highlighting the behavior near the edges of the region.

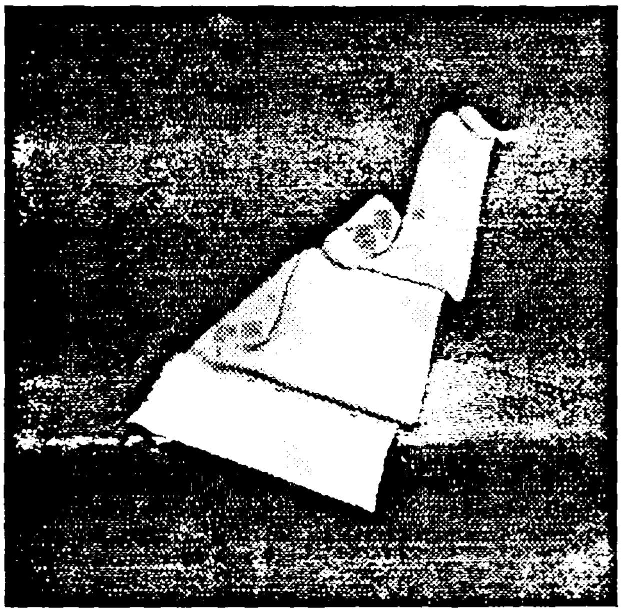

Figure 3

Figure 4 shows the surface generated when this method is applied to interneed not be any restriction on the location of (x_i, y_i); they can even be chosen randomly.

2. The depth values derivable from a contour: $L_i f = C_i$, where C_i is a particular curve. This is the usual result of triangulation based on the first stage of edge detection methods (zero-crossing contours, etc.), or work on silhouettes.

3. A given directional derivative: $L_i f =$ the directional derivative of $f(x_i, y_i)$ with respect to l_i, where l_i is a direction vector in R^2. This would be a one-dimensional variant of the shape-from-shading problem, where the available information samples are the uniquely determined surface slopes in a particular direction.

4. The integral (generally line integrals) of density functions:

$$L_i f = \int_{(x_{i1}, y_{i1})}^{(x_{i2}, y_{i2})} \rho(x, y).$$

   This is the type of information obtained by the processes of computer Axial tomography (CAT scans). Here the problem is not to recover just a surface in three space, but rather to recover the density distribution.

The linearity of information is important: it appears to be a key determinate of many of the existing results of the general theory. Inasmuch as many image observables are non-linear functionals of the surface, these important cases of non-linear problems remain to be thoroughly treated, although there are hopeful signs. Recently, it has been proved that non-linear continuous information is not more powerful than linear continuous information [Kacewicz 84]. Additionally, although most results deal with worst case models, the average and asymptotic cases are also under investigation: see [Traub 84, Wasilkowski A 84, Wasilkowski B 84].

7. Future Work

We see several areas of great interest. We plan to investigate the effects of missing or errorful information. The general theory is being pursued along those lines as well, so some results may be straightforward corollaries.

More practically, our interest is in finding a more efficient algorithm for evaluating the basis spline coefficients. Since an exact solution may be difficult, we are also exploring various approximate techniques, particularly with regard to replacing the basis splines with ones that are finitely supported, or with ones that are only asymptotically correctly shaped for their position. One approach centers on finding a single basis spline function which can accurately approximate all the others. Such a spline would have the computational advantage that only one set of coefficients need be stored, since all the other basis splines would be simple translations of it.

A second overriding concern is the thorough investigation of the numerical properties of both the exact and approximate algorithms. In particular, we are interested in the exploration of the numerical stability of these algorithms when they are applied to depth information along contours: that is, the usual passive stereo based on a primal sketch. We wish to obtain bounds on the absolute error of these algorithms, and find the optimal information for our particular case.

Thirdly, we plan to implement the algorithm for a variety of different classes of F_1, and to investigate the psychological plausibility of these choices for F_1. We hope to find solid psychological grounds for a particular choice of that class.

8. Summary

We believe that the information-centered approach to algorithms can be applied to many vision problems with powerful results. In this paper, we introduced the method and have shown how results pertinent to depth map interpolation are corollaries of the general theory. The major results are that spline interpolations are provably optimal in the worst case; that the resultant linear algorithms are exceedingly simple and parallelizable given some precomputation; and that adaption does not help. Our hope is that the application of this approach to other vision problems will provide similar insight and computational power, and will similarly help ground other existing heuristic methods in provably optimal algorithms.

9. Acknowledgments

We thank J. F. Traub, H. Woz'niakowski, and G. W. Wasilkowski for their friendly and valuable comments.

References

[Babenko 79] Babenko, K.I. (editor).
Theoretical Foundations and Construction of Computational Algorithms for the Problems of Mathematical Physics, (in Russian).
, Moscow, 1979.

[Boult 85] Boult, T.
Reproducing Kernel for Visual Surface Interpolation.
Technical Report CUCS155-85, Department of Computer Science, Columbia University, March, 1985.

[Boult 86] Boult, T.
Examples of Information Based Complexity in Non-Linear Equations and Computer Vision.
PhD thesis, Department of Computer Science, Columbia University, in preperation for 1986.

[Duchon 76] Duchon, J.
Interpolation de Fonctions de Deux Variables Suivant le Principe de la Flexion des Plaques Minces.
Revue Francaise d'Automatique, Informatique et Recherche Operationnelle :5-12, December, 1976.

[Grimson 81] Grimson, W. E. L.
From Images to Surfaces.
MIT Press, 1981.

[Ikeuchi 81] Ikeuchi, K. and Horn, B. K .P.
Numerical Shape from Shading and Occluding Boundaries.
Artificial Intelligence 17:141-184, 1981.

[Kacewicz 84] Kacewicz, B. and Wasilowski, G.W.
How Powerful is Continuous Nonlinear Information?.
Technical Report, Department of Computer Science, Columbia University, 1984.

[Meinguet 83] Meinguet, J.
Surface Spline Interpolation: Basic Theory and Computational Aspects.
Institut de Mathematique Pure dt Appliquee, Universite Catholique de Louvain (35), 1983.

[Rissanen 73] Rissanen, J.
Algorithms for Triangular Decomposition of Block Hankel and Toeplitz Matrices with Application to Factoring Positive Matrix Polynomials.
Mathematics of Computation 27(121):147-154, 1973.

[Terzopoulos 84]
Terzopoulos, D.
Multiresolution Computation of Visible-Surface Representations.
PhD thesis, Massachusetts Institute of Technology Artificial Intelligence Laboratory, 1984.

[Traub 80] Traub, J. F., and Woz'niakowski, H.
A General Theory of Optimal Algorithms.
Academic Press, 1980.

[Traub 83] Traub, J.F., Wasilkowski, G.W., and Woz'niakowski, H.
Information, Uncertainty, Complexity.
Addison-Wesley,Reading,MA, 1983.

[Traub 84] Traub, J. F. and Woz'niakowski, H.
Information and Computation.
In *Advances in Computers*, pages 35-92. Vol.23, 1984.

[Wasilkowski A 84]
Wasilkowski, G. W.
Local Average Errors.
Technical Report , Department of Computer Science, Columbia University, 1984.

[Wasilkowski B 84]
Wasilkowski, G. W.
Optimal Algorithm for Linear Problems with Gaussian Measures.
To appear in Rocky Mt. J. Math. , Computer Science Department, Columbia University, 1984.

Table of Contents