On 3D Model Construction by Fusing Heterogeneous Sensor Data *

Y. F. Wang
Department of Computer Science
University of California
Santa Barbara, CA 93106
Email: yfwang@cs.ucsb.edu
Tel: (805) 893-3866

Jih-Fang Wang
United Technologies Research Center
411 Sliver Lane, MS 129-13
East Hartford, Connecticut 06108
Email: jfw@utrc.utc.com
Tel: (203) 647-9752

Abstract

In this paper, we propose a scheme for 3D model construction by fusing heterogeneous sensor data. The proposed scheme is intended for use in an environment where multiple, heterogeneous sensors operate asynchronously. Surface depth, orientation, and curvature measurements obtained from multiple sensors and vantage points are incorporated to construct a computer description of the imaged object. The proposed scheme uses Kalman filter as the sensor data integration tool and hierarchical spline surface as the recording data structure. Kalman filter is used to obtain statistically optimal estimates of the imaged surface structure based on possibly noisy sensor measurements. Hierarchical spline surface is used as the representation scheme because it maintains high-order surface derivative continuity, may be adaptively refined, and is storage efficient. We show in this paper how these mathematical tools can be used in designing a modeling scheme to fuse heterogeneous sensor data.

*This research was supported in part by a grant from the National Science Foundation, IRI-8908627.

1 Introduction

In this paper, we propose a scheme for constructing 3D models by fusing heterogeneous sensor data. The proposed scheme is intended for use in an environment where multiple, heterogeneous sensors operate asynchronously. Surface depth, orientation, and curvature measurements obtained from multiple sensors and vantage points are incorporated to construct a computer description of the imaged object.

The proposed scheme uses Kalman filter to obtain statistically optimal estimates of the imaged surface structure based on possibly noisy sensor measurements. Kalman filter has been widely used to estimate the internal state of a system based on the observation of the system's external behavior [16],[33]. Furthermore, a system's state estimate can be computed and then updated by incorporating external measurements iteratively—*without* recomputing the estimate from scratch each time a new measurement is made available. We also choose hierarchical B-spline surface to record the reconstructed surface structure because hierarchical B-spline surface (1) maintains high-order surface derivative continuity, (2) may be adaptively refined to capture fine surface details while still maintains the smoothness and continuity of the whole structure, (3) has desirable local control property, and (4) is storage efficient because, regardless of the physical size of a patch, only a small number of control vertices need to be recorded.

In this research, we formulate 3D shape modeling by sensor data fusion as a least-square state estimation problem based on Kalman filtering. Control vertices of a B-spline patch comprise the (internal) "state vector," based on which the shape of the imaged surface is described. We allow the depth, orientation, and curvature measurements derived from multiple, heterogeneous sensors to serve as the (external) "observations" to constrain the internal state, or the control vertices, of the surface data structure. We discuss in the paper how a variety of shape constraints are used to determine the imaged surface structure, how an accuracy measure for the reconstructed structure is computed and updated, and how surface structure can be propagated in both the spatial and temporal domains.

Recently, the importance of fusing heterogeneous sensor data in 3D modeling and analysis was recognized. As observed by Professor Takeo Kanade [26], progress of computer vision research a decade ago was severely hampered by the limited computational power and the few imaging devices available then. Hence, only a small amount of sensor data could be gathered and analyzed. To fill in void in the analysis, Researchers then tended to resort to clever mathematical tricks and unrealistic assumptions about the 3D scenes. Algorithms thus developed had limited applicability and broke down easily in real-world applications. More recently, significant advances in microprocessor and disk storage technologies made available ever more powerful computers with large storage capacity. At the same time, a variety of novel sensors—such as thermal, tactile, and laser range sensors—were

introduced into the market. Hence, it is now feasible to process the large amount of data gathered from a heterogeneous collection of sensors for image analysis.

The advantages of sensor data fusion are two-fold: First, as more information about the scenes is acquired and analyzed, the need to make special, unrealistic scene assumptions diminishes. It is thus possible to design algorithms which are robust and generally applicable. Second, it may be noted that different sensors possess distinct characteristics, are designed based on differing physical principles, operate in a wide range of the electromagnetic spectrum, and are geared toward a variety of applications. A single sensor operating alone provides a limited sensing range, and is inherently unreliable due to operational errors. However, a synergistic operation of many sensors provides a rich body of information on the sensed environment from a wide range of the electromagnetic spectrum to aid the image analysis.

Fusing sensor data involves more than choosing the right sensors and averaging sensor output. As pointed out in [15], simple averaging of sensor measurements is prone to distortion by individual noisy measurements, and makes the overall variance of measurements worse. Further, sensor data may be used in such a way that data from one sensor guide the operation and exploration of another sensor. Such is the case in using a visual sensor with a tactile sensor [6],[7],[8],[39], and in intensity guided range sensing [2]. Techniques have also been developed to fuse thermal and visual sensing [35], ladar and visual sensing [17], range and visual sensing [47], thermal and tactile sensing [38], and to fuse multiple sensing modalities for robot navigation [9],[10],[20],[28],[29],[32],[43],[44]. Interested readers are referred to [3],[4] for a more detailed survey of recent research work on sensor data fusion.

Salient features of our modeling technique are: Using Kalman filter and the sensor error models developed in [50], we demonstrate a technique for sensor data fusion in a statistically optimal way; using the spline surface as the recording data structure, higher-order surface smoothness is achieved; our formulation still allows randomly-scattered heterogeneous sensor data to be used in model construction; and finally, the technique of adaptive refinment and state subdivision enables our modeling scheme to adapt to different degrees of complexity of the imaged structures.

Another important issue which our sensor data fusion technique addresses is how to measure the faithfulness or accuracy of a 3D modeling technique. Though the spline functions have been widely used in graphics and solid modeling applications [11],[22],[37], their usage there is mainly for creating artificial structures instead of approximating existing structures. Bound only by the creator's imagination, a structure thus created does not necessarily resemble any real-world object (e.g., an artist's rendering of a concept automobile can be quite futuristic looking indeed). Hence, an objective "accuracy" or "faithfulness" criterion is difficult, if not impossible, to formulate.

On the other hand, in computer vision and robotics applications where real-world objects are modeled, the faithfulness of a computer model can be defined in rigid mathematical terms such as

the deviation in surface or volume measurements, placement error of a body part, misalignment of the orientation of a surface patch, etc. Hence, it is not sufficient just to create a computer model of the imaged object, it is also important to have a measure of how faithful one can expect of such a representation. Our use of Kalman filter to establish such a statistical accuracy measurement and our formulation of the 3D modeling process as a filtering process, we believe, are a novel way which combines the mathematical spline theory with the least-square estimation theory for computer vision applications.

The remainder of this paper is organized as follow: Section 2 contains a brief review of the mathematical tools used. Section 3 discusses the proposed modeling scheme. Section 4 presents implementation results using both synthetic and real image data. Finally, Section 5 contains a concluding remark.

2 Background

To integrate information from multiple sensory sources and viewing directions for 3D model construction, we propose to use Kalman filter as the sensor data integration tool, and hierarchical B-spline surface as the recording data structure. Both the linear estimation theory and the mathematical spline theory are well established and documented. Hence, our discussion here is limited to the extent such that their applications in our modeling scheme can be introduced. Interested readers are referred to [5],[11],[16],[18],[22],[30],[33],[37] for more comprehensive discussions of the linear estimation theory and the mathematical spline theory.

Kalman Filtering Kalman filter has been widely used to obtain statistically optimal estimates of the internal state of a system based on measurements of the system's external behavior. More precisely, a system—be it a biological, mechanical, or electrical one—is usually characterized by a finite collection of variables which are called the "internal state" of the system. As the word internal implies, the state a system is in usually cannot be directly sensed. However, the external behavior of a system—which is heavily influenced by the state the system is in—can usually be observed by some means.

The state inference is complicated by various noise processes which operate to corrupt the state prediction mechanism [16],[33]. Because generally only the statistic properties, not the exact shape, of a noise process are known, one should not expect to obtain a unique estimate of the system's state in a deterministic sense, but should be satisfied with one in a *probabilistic* sense. It can be proven [16],[33] that, under very general assumptions, Kalman filter provides a formulation to obtain such statistically optimal estimations with the minimum variance (or the maximum certainty). Furthermore, Kalman filter obtains such estimates in an iterative manner. I.e., an optimal

estimate can be computed by incorporating observations one at a time—*without* recomputing the best estimate from scratch each time a new measurement is made available and *without* storing all previously-made measurements.

By building a system which uses hierarchical B-spline surface to represent the imaged 3D object, and employing sensors to provide observations or image constraints, we were able to formulate our modeling scheme as a filtering process. Hence, we utilize Kalman filter for 3D structural construction and update.

B-Spline Surface The mathematical spline theory is also well established [5],[11],[18],[22],[37]. Many forms of spline surfaces based on different control structures and blending functions are known, such as the Bézier patch, Coons patch, B-spline patch, and spline surface under tension. In general, spline functions allow a small number of control parameters—such as the position, orientation, and curvature vectors—to be specified. The shape of a spline function is then generated by "blending" the control parameters together according to the parameter type and the function location. For a spline curve, the control parameters are usually specified along a 1D segment, while for a spline surface, they are distributed on a 2D grid. The advantage of using a spline representation over, say, a finite element representation [12],[25] is that the spline surfaces are usually smoother (i.e., spline surfaces maintain the continuity of high order surface derivatives) while still allow randomly distributed sensor data to be used in model construction.

We choose the bi-cubic B-spline patch—which has a relatively simple control structures and maintains up to the second-order surface derivative continuity—to record the reconstructed surface structure. B-spline stands for the *basis* spline. As the word basis suggests, the shape of a spline function is approximated by using a set of spline bases, which are defined in the same function domain and each modulated by a control parameter. The modulated bases are then summed to approximate the shape of the desired function. For example, a B-spline curve $c(x)$ is represented by a set of basis functions $N_{i,k}$ defined over a linear knot vector [22],[37], and each basis function is modulated by a control vertex P_i

$$c(x) = \sum_{i=0}^{n-1} P_i N_{i,k}(x). \tag{1}$$

The knot vector used to define the B-spline basis sets can be of any length and possibly with repeating elements (i.e., the knot vector need be nondecreasing but not strictly increasing). The B-spline basis functions thus defined are polynomials of different degrees k. A B-spline surface is defined as the Cartesian product of two B-spline curves.

An important reason that the hierarchical B-spline surface is chosen as the recording surface data structure is its subdivision and refinement property [19],[22],[37]. Generally speaking, it is not known *a priori* how complicated the imaged surface structure may be. Hence, it is undecidable in

advance how sophisticated the recording data structure should be. To capture surface details as they are revealed by sensing, a recording data structure should adapt to different surface complexities and increases the descriptive power only when the need arises. The B-spline subdivision property allows additional nodes to be inserted into the knot vector to define more basis functions over the same parametric range, and hence, more descriptive power results [19]. Furthermore, the B-spline subdivision procedure enables adaptive refinement of local structures while still maintains the smoothness and continuity of the global solution. Such properties are not known for other popular recording structures such as the superquadric.

3 Sensor Data Fusion for 3D Modeling

To utilize Kalman filter in sensor data fusion, we need to identify (1) a linear system, its internal "state variables," and its external "behavior," and (2) means to measure the external behavior of the system. the spatial and temporal domains. We propose to use hierarchical B-spline surface as our "linear system" and data from multiple, heterogeneous sensors as our "measurements" on the system.

3.1 The Modeling Scheme

In this paper, we use the B-spline basis of order four (or a third-degree polynomial) in both parametric directions

$$\mathbf{S}(u,v,t) = \sum_{i=0}^{m-1}\sum_{j=0}^{n-1} \mathbf{P}_{i,j}(t) M_{i,4}(u) N_{j,4}(v) \ , \tag{2}$$

where $\mathbf{P}_{i,j} = (x_{i,j}, y_{i,j}, z_{i,j})$ denotes the 3D position of the i,j-th control vertex, and $M_{i,4}(u)$ and $N_{j,4}(v)$ denote the spline bases, the expression of which is well known (e.g., see [22],[37]). Since the shape of a B-spline patch is completely specified by a small number of control vertices $\mathbf{P}_{i,j}$, these control vertices, in the terminology of Kalman filtering, comprise the (internal) state vector of the system, based on which the (external) behavior of the system is specified.

The choice of the control vertex positions as the internal state vector deserves some explanation. In [33], the state of a system is defined to be a minimum set of values, which, along with the input to the system, is sufficient to determine completely the behavior of the system. As observed in Equation 2, the shape of the spline function $\mathbf{S}$ at any location (u,v) and at any time t is completely defined in terms of variables $\mathbf{P}_{i,j}$. Furthermore, each $\mathbf{P}_{i,j}$ is specified independently, and all of them have to be known to uniquely determine the shape of the spline surface $\mathbf{S}$. Hence, the collection of all control vertices constitutes a minimum set of values which comprises a state. Certainly, if a different representation scheme is used, then a different composition of the state vector results. For example, if a Coons patch is used, then the state vector consists of the position, tangent, and

twist vectors at the patch's four corners [22],[37], while if a superquadric is used, the state vector comprises then of the eleven parameters which define the position, orientation, size, and shape of the superquadric [24]. The choice of the state vector is not unique though, as any valid linear transformation of the state vector defines a new state vector. However, the transformed state vector defines identical system behaviors and does not necessarily aid the solution of the problem.

We allow the depth, orientation, and curvature measurements derived from multiple sensors to serve as the "observations" on the system. These sensor measurements are used to determine the value of the "state vector" (i.e., the positions of the control vertices). Sensor measurements constrain the surface position ($\mathbf{S}$), orientation ($d\mathbf{S}$), and curvature ($d^2\mathbf{S}$) in space, and in turn, allow the positions of the control vertices, $\mathbf{P}_{i,j}$, to be determined (Equation 2).

Many types of sensors may be used to gather information on the surrounding environment, for example, visual sensors, visual sensors augmented with structured lighting, thermal (infrared) sensors, proximity sensors, tactile sensors, ultrasonic rangefinders, and laser rangefinders. And analysis techniques have been developed to analyze data from these sensors [1],[14] to infer surface position, orientation, and curvature from images. For example, surface depth is obtained from the stereo analysis of visual images or from range sensors directly. The shape from shading and shape from texture methods make available information on surface orientation. In our experiments, we employed the active structured-lighting technique to infer surface orientation and curvature [46],[49] in shape construction.

For example, the following equation can be used to constrain the state vector using the depth measurements supplied by, say, the stereopsis or laser ranging techniques

$$\mathbf{D}(u,v,t) + \epsilon_{pos} = \mathbf{S}(u,v,t) = \sum_{i=0}^{m-1}\sum_{j=0}^{n-1} \mathbf{P}_{i,j}(t) M(u) N(v) \ , \tag{3}$$

where $\mathbf{D}(u,v,t)$ denotes a depth measure at location (u,v) at time t on the surface, and ϵ_{pos} denotes the sensor noise. Similarly, if measurements on surface orientation $\mathbf{N}(u,v,t)$ are made available through, say, shape from shading or shape from texture gradient techniques, then constraints on the state variables $\mathbf{P}_{i,j}$ can be imposed (where ϵ_{ori} denotes the orientation measurement noise)

$$\begin{aligned} (\mathbf{N}(u,v,t) + \epsilon_{ori}) \cdot \mathbf{S}_u(u,v,t) = 0 \quad &\Rightarrow \textstyle\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}((\mathbf{N}(u,v,t) + \epsilon_{ori}) \cdot \mathbf{P}_{i,j}(t))\ M_u(u)\ N(v) = 0 \\ (\mathbf{N}(u,v,t) + \epsilon_{ori}) \cdot \mathbf{S}_v(u,v,t) = 0 \quad &\Rightarrow \textstyle\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}((\mathbf{N}(u,v,t) + \epsilon_{ori}) \cdot \mathbf{P}_{i,j}(t))\ M(u)\ N_v(v) = 0 \ . \end{aligned} \tag{4}$$

If the principal surface curvatures, κ_1 and κ_2, and their directions, are inferred using, say, the structured light technique reported in [49], then constraints on the second surface derivatives, $\mathbf{S}_{uu}$ and $\mathbf{S}_{vv}$, can also be derived. More precisely, we first apply Euler's theorem [21],[41] to compute the normal surface curvatures, κ_{u_n} and κ_{v_n}, along the $\mathbf{S}_u$ and $\mathbf{S}_v$ directions

$$\kappa_{u_n} = \kappa_1 cos^2\theta_u + \kappa_2 sin^2\theta_u \qquad\qquad \kappa_{v_n} = \kappa_1 cos^2\theta_v + \kappa_2 sin^2\theta_v \ , \tag{5}$$

where θ_u and θ_v are the angles between the direction of κ_1 and those of $\mathbf{S}_u$ and $\mathbf{S}_v$, respectively. Then applying Meusnier's theorem [21],[41], we obtain the second derivatives along the $\mathbf{S}_u$ and $\mathbf{S}_v$ directions as

$$\mathbf{C}_{uu} = \kappa_{u_n}/cos\phi_u, \qquad \mathbf{C}_{vv} = \kappa_{v_n}/cos\phi_v \ , \tag{6}$$

where ϕ_u and ϕ_v denote, at the point $\mathbf{S}(u,v,t)$ under consideration, the angles between the surface normal direction and the principal normal direction of the surface curve which is the intersection of the spline surface with the projection of the coordinate axes in space. One can easily show that the second-order surface measurements can be used to constrain the control vertices $\mathbf{P}_{i,j}$ by (where ϵ_{curv} denotes the curvature measurement noise)

$$\begin{aligned} \mathbf{C}_{uu}(u,v,t) + \epsilon_{curv} &= \mathbf{S}_{uu}(u,v,t) &= \textstyle\sum_{i=0}^{m-1}\sum_{j=0}^{n-1} \mathbf{P}_{i,j}\ M_{uu}(u)\ N(v) \\ \mathbf{C}_{vv}(u,v,t) + \epsilon_{curv} &= \mathbf{S}_{vv}(u,v,t) &= \textstyle\sum_{i=0}^{m-1}\sum_{j=0}^{n-1} \mathbf{P}_{i,j}\ M(u)\ N_{vv}(v)\ . \end{aligned} \tag{7}$$

Sensor data are inevitably corrupted by noise which is denoted by ϵ_{pos}, ϵ_{ori}, and ϵ_{curv} in Equations 3, 4, and 7. Error in the sensor measurements can arise from the simplification made in a mathematical sensor model (e.g., using the parallel projection to approximate the image formation process), improper calibration and operation of the imaging equipments (e.g., supplying inaccurate baseline or focal length value in a stereopsis analysis), image processing and numerical computation (e.g., mislocating feature positions), etc. Hence, when heterogeneous sensor data are combined, their contribution in the 3D model construction should be weighted by their associated confidence measure.

Kalman filter formulation thus requires that a model of error in the sensor measurements be specified in addition to sensor data. Sensor error models are used in designing a confidence measure for each sensor used. Generally speaking, a sensor error model is stochastic because only the statistic properties, such as the mean and variance, of the sensor noise can be inferred. In general, the error covariance matrix

$$\mathbf{\Lambda}_\epsilon = E[\epsilon\epsilon^T] \tag{8}$$

should be supplied [16],[33].

Though many image analysis techniques using different sensing modalities have been developed, research on the analysis and modeling of sensor error is still quite lacking. Most researches on error analysis in computer vision have been concerned with the analysis of the stereopsis technique and have considered only the quantization error [13],[31],[34],[36],[45]. We show in [50] how the sensor error model can be designed for the structured light sensing technique, which is the imaging technique used in our experiment. Assume that the error covariance matrix $\mathbf{\Lambda}_\epsilon$ is available [50], the sensor data integration process involves estimating the control vertices and updating the error covariance matrix (an accuracy measure) using the available shape constraints. Or we collect all

the shape constraints specified using Equations 3, 4, and 7, put them in a matrix form $\mathbf{AP} = \mathbf{B}$, and solve the equation by weighing each constraint with its expected accuracy [16],[33]

$$\begin{aligned} \mathbf{\Lambda_p'} &= (\mathbf{I} - \mathbf{KA})\mathbf{\Lambda_p} \\ \mathbf{P'} &= \mathbf{P} + \mathbf{K}(\mathbf{B} - \mathbf{AP}) \ , \end{aligned} \tag{9}$$

and

$$\mathbf{K} = \mathbf{\Lambda_p} \mathbf{A}^T (\mathbf{A} \mathbf{\Lambda_p} \mathbf{A}^T + \mathbf{\Lambda}_\epsilon)^{-1} \ , \tag{10}$$

where unprimed symbols denote quantities before the external measurements are incorporated and primed symbols denote quantities after the external measurements are incorporated. $\mathbf{P}$ represents the state vector composed of all the control vertices $\mathbf{P}_{ij}$, $\mathbf{A}$ is the observation matrix which relates the depth, orientation, and curvature observations—denoted by $\mathbf{B}$—to the state vector $\mathbf{P}$. $\mathbf{\Lambda}_\epsilon$ is the error covariance matrix of the sensor observations and $\mathbf{\Lambda_p}$ is the error covariance matrix of the state vector $\mathbf{P}$. Finally, $\mathbf{K}$ is usually called the Kalman gain matrix.

Intuitively speaking, the external measurements impose constraints on the internal state vector through a linear relation $\mathbf{AP} = \mathbf{B}$. However, the external measurements are usually corrupted by noise and may even be correlated. Hence, in updating the confidence of the internal state estimate $\mathbf{\Lambda_p}$, sensory data uncertainty is taken into consideration by weighing the new data with the observations' error covariance matrix $\mathbf{\Lambda}_\epsilon$. The state vector updates are incremental in the sense that only the information in the new observation $\mathbf{B}$ which cannot be accounted for in the old estimate $(\mathbf{B} - \mathbf{AP})$ is incorporated, weighted by the gain matrix $\mathbf{K}$.

To summarize, we allow the use of the heterogeneous shape cues from multiple sensors for surface reconstruction. External sensor measurements are used to constrain B-spline's internal control vertices through Equations 3, 4, and 7. Furthermore, depth, orientation, and curvature constraints from randomly distributed locations in an image can be included iteratively.

3.2 State Subdivision

As mentioned before, the complexity of the imaged structure cannot be predicted in advance, hence, it is difficult to determine before hand how sophisticated the recording data structure should be. Our choice is to start with a simple B-spline patch defined on a small knot vector with a small number of control vertices. When the surface structure of the imaged object cannot be represented faithfully using a single B-spline patch (most likely not in the real world), the B-spline patch is refined by introducing more knots and more control vertices. In the terminology of Kalman filtering, we construct a more complicated and powerful system by refining a simple one.

Difficulty with the refinment process is that if an unlimited number of knots and control vertices are allowed to be added to the original patch, the patch's structure may become highly complex and

difficult to process. It is easily seen that for a patch defined on $m \times n$ control vertices, increasing m or n by a factor of, say, two doubles the size of the state vector and quadruples the size of the covariance matrix! Or the complexity in processing the covariance matrix is $O(n^4)$. Hence, an alternate, more efficient way to increase the descriptive power of the B-spline surfaces must be found.

In this paper, refinement is accomplished by a process of state subdivision. This subdivision process allows introducing additional knots and control vertices to increase the descriptive power of a spline surface while avoids an $O(n^4)$ growth in complexity. More precisely, in stead of building one big, complex B-spline patch, we break the patch down into a number of component patches, each having the *same* complexity as the original one. Since each component patch resulted from subdivision is of the same structure as the original one, each has the same descriptive power as the original one. However, the combined descriptive power of all the component patches increases while the time complexity increases only linearly in terms of the number of the component patches.

This state subdivision process can be carried out recursively, i.e., a component surface patch resulting from subdivision can itself be subdivided further. A hierarchical quadtree representation based on multiple B-spline patches is thus constructed. Such a representation is advantageous in that fine surface details are captured by recursively subdividing B-spline patches into subpatches to achieve visual realism, while coarse structures are represented by large patches to improve efficiency.

A word on the criteria of subdivision: the decision of whether to divide a patch is based on the variation of the patch's Gaussian curvature. Large variation in Gaussian curvature suggests that the surface structure is relatively rough, and subdivision is used to generate more patches in order to capture more surface details. Gaussian curvature, κ, can be computed using [41]:

$$\kappa = \kappa_1 \, \kappa_2 = \frac{eg - f^2}{EG - F^2} \, , \tag{11}$$

where κ_1 and κ_2 denote the two principal curvatures, E, F, and G are the coefficients of the first fundamental form, and e, f, and g are those of the second fundamental form

$$\begin{aligned} & E = \mathbf{S}_u \cdot \mathbf{S}_u \, , \quad F = \mathbf{S}_u \cdot \mathbf{S}_v \, , \quad G = \mathbf{S}_v \cdot \mathbf{S}_v \, , \\ & e = \mathbf{S}_{uu} \cdot \mathbf{N} \, , \quad f = \mathbf{S}_{uv} \cdot \mathbf{N} \, , \quad g = \mathbf{S}_{vv} \cdot \mathbf{N} \, , \\ & \mathbf{N} = \tfrac{\mathbf{S}_u \times \mathbf{S}_v}{|\mathbf{S}_u \times \mathbf{S}_v|} \, . \end{aligned} \tag{12}$$

To compute the variation of Gaussian curvature over a spline surface, we first compute Gaussian curvature at selected locations on the surface. If we have a spline surface defined by $m \times n$ control vertices (Assume that both m and n are odd. We use $m = n = 7$ in our experiments), a total of $(m-3) \times (n-3)$ bi-cubic patches are defined within the surface. For each component patch, we evaluate Equation 11 at its center, $u = v = \frac{1}{2}$, (or at more locations if desired). Then the variation

(variance) of Gaussian curvature is defined as:

$$\sigma_\kappa = \sum_{i=0}^{m-4} \sum_{j=0}^{n-4} (\kappa_{i,j} - \bar{\kappa})^2 \ , \tag{13}$$

and

$$\bar{\kappa} = \frac{1}{(m-3)(n-3)} \sum_{i=0}^{m-4} \sum_{j=0}^{n-4} \kappa_{i,j} \ ,$$

where $\kappa_{i,j}$ is the curvature at the center of the i,j-th component patch, and $\bar{\kappa}$ is the average curvatures of all component patches. The patch splitting threshold is set on value $\sigma_\kappa/\bar{\kappa}$, which is the variation of Gaussian curvature normalized by the mean. We have arbitrarily selected a value of 0.5, i.e, if the variation is more than half the mean value, the patch is a potential candidate for refinement.

Now we are ready to present the subdivision algorithm. As just mentioned, a spline surface defined by $m \times n$ control vertices comprises $(m-3) \times (n-3)$ bi-cubic patches. We divide each bi-cubic patch into four sub-patches by inserting $(m-3) \times (n-3)$ control vertices. After subdivision, a total of $(2m-6) \times (2n-6)$ bi-cubic patches result. We partition them into four groups, each contains $(m-3) \times (n-3)$ bi-cubic patches defined on $m \times n$ control vertices.

In this paper, we use a "halving" subdivision technique which is a special case of the general Oslo algorithm [19]. If each bi-cubic patch is defined on a parametric grid of unit size $[0..1, 0..1]$, then the division is done along the central lines $u = \frac{1}{2}$ and $v = \frac{1}{2}$, and the four sub-patches are defined on grids $[0..\frac{1}{2}, 0..\frac{1}{2}]$, $[0..\frac{1}{2}, \frac{1}{2}..1]$, $[\frac{1}{2}..1, 0..\frac{1}{2}]$, and $[\frac{1}{2}..1, \frac{1}{2}..1]$. Immediately after subdivision, the combined shape of the four subdivided spline surfaces duplicates that of the original surface. However, since each spline surface resulted from subdivision has the same descriptive power as the original one but spans only a quarter of the original parametric space, the combined descriptive power of the four subdivided surfaces increases. As more sensor data are gathered and processed, the four combined surface patches provide a more faithful representation than the original surface.

We record the new spline surfaces resulted from subdivision in a quadtree structure. We let the original spline surface be the root of a new subtree and create four child nodes to accommodate the four new surfaces. The process is recursively applied to generate sub-patches from a parent patch, and a quadtree structure is constructed in the parameter plane. Each subdivision generates four more nodes and possibly increases the depth of the tree by one. Image shape constraints are filtered down the tree and picked up by surface patches where the constraints apply. Hence, the same set of shape constraints are used at multiple levels of the quadtree by spline surfaces of different physical sizes.

Two more technical details worth mentioning here: firstly during subdivision, the state covariance matrix must also be propagated down the tree to reflect the confidence in the new control

vertex positions. The mathematical spline theory provides a mechanism to propagate the control vertex positions during subdivision, but no provision is made for propagating the covariance matrices which are needed only in a Kalman estimation process. Hence, we derive such a propagation scheme below.

According to the Oslo's algorithm [19], control vertices at a finer level ($\mathbf{P}^{(f)}$) are a linear combination of those at the preceding coarser level ($\mathbf{P}^{(c)}$). Suppose that we have a B-spline curve $c'(x)$, defined on a knot vector $(x'_0, x'_1, \cdots, x'_{m-1})$ of length m $(> n)$ using a set of control vertices P'_j, or

$$c'(x) = \sum_{j=0}^{m-1} P'_j N_{j,k}(x) \ . \tag{14}$$

In order for $c'(x)$ to possess the same shape as $c(x)$ defined in Equation 1, P'_j and P_i are related by:

$$P'_j = \sum_{i=0}^{n-1} \phi_{i,j}^k P_i \ , \tag{15}$$

where $\phi_{i,j}^k$ is given by a recursive formula:

$$\phi_{i,j}^k = \frac{x'_{j+k-1} - x_i}{x_{i+k-1} - x_i} \phi_{i,j}^{k-1} + \frac{x_{i+k} - x'_{j+k-1}}{x_{i+k} - x_{i+1}} \phi_{i+1,j}^{k-1} \ , \tag{16}$$

and

$$\phi_{i,j}^1 = \{ \begin{array}{ll} 1 & if\ x_i \leq x'_j < x_{i+1} \\ 0 & otherwise \ . \end{array} \tag{17}$$

The covariance matrix propagation can also be expressed as such. In 1D, we have

$$\begin{aligned} \Lambda^{(f)} &= E[(\bar{\mathbf{P}}^{(f)} - \mathbf{P}^{(f)})\ (\bar{\mathbf{P}}^{(f)} - \mathbf{P}^{(f)})^T] \\ &= E[(\textstyle\sum_{i=0}^{n-1} \phi_i^k \bar{\mathbf{P}}_i^{(c)} - \sum_{i=0}^{n-1} \phi_i^k \mathbf{P}_i^{(c)})\ (\sum_{j=0}^{n-1} \phi_j^k \bar{\mathbf{P}}_j^{(c)} - \sum_{j=0}^{n-1} \phi_j^k \mathbf{P}_j^{(c)})^T] \\ &= E[(\textstyle\sum_{i=0}^{n-1} \phi_i^k (\bar{\mathbf{P}}_i^{(c)} - \mathbf{P}_i^{(c)}))\ (\sum_{j=0}^{n-1} \phi_j^k (\bar{\mathbf{P}}_j^{(c)} - \mathbf{P}_j^{(c)}))^T] \\ &= \textstyle\sum_{i=0}^{n-1} \sum_{j=0}^{n-1} \phi_i^k E[(\bar{\mathbf{P}}_i^{(c)} - \mathbf{P}_i^{(c)})(\bar{\mathbf{P}}_j^{(c)} - \mathbf{P}_j^{(c)})] \phi_j^k \\ &= \mathbf{\Phi} \Lambda^{(c)} \mathbf{\Phi}^T \ , \end{aligned} \tag{18}$$

where $\bar{\mathbf{P}}$ denotes the true (unknown) state vector.

Secondly, when the image constraints are applied to one patch but not the adjacent ones, the shape of the particular patch, but not those of its neighbors, is modified by the constraints. Hence, the continuity of surface position and derivatives across patch boundaries might be lost and even artificial holes may start to appear. Patch boundaries are adjusted to restore continuity before display. Again, provision must be made for this nonstandard basis spline operation.

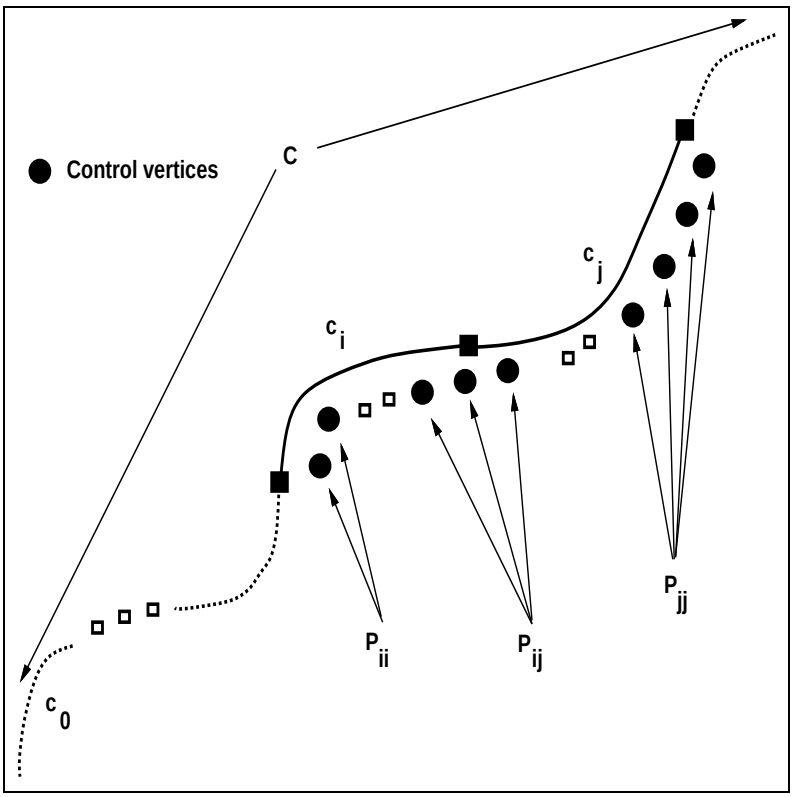


Figure 1: Spline curve subdivision

The boundary adjustment process involves adjusting the locations of control vertices which are shared by adjacent patches and updating the covariance matrices. We illustrate this process using a simple 1D example. Referring to Figure 1, if a spline curve c is broken up into several pieces, $c_0, c_1, \cdots, c_{n-1}$, then to insure a smooth transition between two adjacent segments c_i and c_j (i.e., the function value, first, and second derivatives are continuous), adjacent segments must share some control vertices, which we denote as P_{ij} in Figure 1.

Suppose now that the shape of segment c_i is modified by the image constraints but not that of segment c_j, then the position of P_{ij} reported from c_i (denoted as $P_{ij}^{c_i}$) and c_j (denoted as $P_{ij}^{c_j}$) might differ and shape continuity across the boundary of c_i and c_j will be lost. To restore continuity, we partition vertices P_i in segment c_i into two groups: those are shared with c_j (P_{ij}) and those appear

only in c_i (P_{ii}). Then we can partition the covariance matrix of c_i as

$$\begin{aligned}
\Lambda^{c_i} &= E[(\bar{P}_i - P_i)(\bar{P}_i - P_i)^T] \\
&= \left[\begin{array}{c|c} E[(\bar{P}_{ii} - P_{ii})(\bar{P}_{ii} - P_{ii})^T] & E[(\bar{P}_{ii} - P_{ij})(\bar{P}_{ii} - P_{ij})^T] \\ \hline E[(\bar{P}_{ii} - P_{ij})(\bar{P}_{ii} - P_{ij})^T] & E[(\bar{P}_{ij} - P_{ij})(\bar{P}_{ij} - P_{ij})^T] \end{array}\right] \qquad (19) \\
&= \left[\begin{array}{c|c} \Lambda^{c_i}_{ii,ii} & \Lambda^{c_i}_{ii,ij} \\ \hline \Lambda^{c_i}_{ii,ij} & \Lambda^{c_i}_{ij,ij} \end{array}\right],
\end{aligned}$$

and similar partition can be made of Λ^{c_j}.

Positions of the vertices which belong to only one segment need not be updated. New positions of the shared vertices P'_{ij} are obtained by averaging those reported from c_i and c_j, weighted by their respective confidence measure:

$$P'_{ij} = \Lambda^{c_j}_{ij,ij}(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})^{-1} P^{c_i}_{ij} + \Lambda^{c_i}_{ij,ij}(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})^{-1} P^{c_j}_{ij} \ . \qquad (20)$$

Once the shared vertex positions are updated, their covariance matrices are updated next. Note that the covariance matrix in Equation 19 is always symmetrical, hence, only three out of the four submatrices are independent ($\Lambda^{c_i}_{ii,ij}$ appears twice in Λ^{c_i} at the upper right and lower left corners). Of the three independent submatrices ($\Lambda^{c_i}_{ii,ii}, \Lambda^{c_i}_{ii,ij}$, and $\Lambda^{c_i}_{ij,ij}$), $\Lambda^{c_i}_{ii,ii}$ does not involve shared vertices and need not be updated. $\Lambda^{c_i}_{ii,ij}$ and $\Lambda^{c_i}_{ij,ij}$ are updated using the following equations:

$$
\begin{aligned}
{\Lambda^{c_i'}_{ij,ij}}^{-1} &= \\
{\Lambda^{c_j'}_{ij,ij}}^{-1} &= \{E[(\bar{P}_{ij} - P'_{ij})(\bar{P}_{ij} - P'_{ij})^T]\}^{-1} \\
&= \{E[(\bar{P}_{ij} - \Lambda^{c_j}_{ij,ij}(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})^{-1} P^{c_i}_{ij} - \Lambda^{c_i}_{ij,ij}(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})^{-1} P^{c_j}_{ij}) \\
&\qquad (\bar{P}_{ij} - \Lambda^{c_j}_{ij,ij}(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})^{-1} P^{c_i}_{ij} - \Lambda^{c_i}_{ij,ij}(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})^{-1} P^{c_j}_{ij})^T]\}^{-1} \\
&= \{E[(\Lambda^{c_j}_{ij,ij}(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})^{-1}(\bar{P}_{ij} - P^{c_i}_{ij}) + \Lambda^{c_i}_{ij,ij}(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})^{-1}(\bar{P}_{ij} - P^{c_j}_{ij})) \\
&\qquad (\Lambda^{c_j}_{ij,ij}(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})^{-1}(\bar{P}_{ij} - P^{c_i}_{ij}) + \Lambda^{c_i}_{ij,ij}(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})^{-1}(\bar{P}_{ij} - P^{c_j}_{ij}))^T]\}^{-1} \\
&= \{\Lambda^{c_j}_{ij,ij}(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})^{-1} E[(\bar{P}_{ij} - P^{c_i}_{ij})(\bar{P}_{ij} - P^{c_i}_{ij})^T](\Lambda^{c_j}_{ij,ij}(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})^{-1})^T \\
&\qquad + \Lambda^{c_i}_{ij,ij}(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})^{-1} E[(\bar{P}_{ij} - P^{c_j}_{ij})(\bar{P}_{ij} - P^{c_j}_{ij})^T](\Lambda^{c_i}_{ij,ij}(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})^{-1})^T\}^{-1} \\
&= \{\Lambda^{c_j}_{ij,ij}(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})^{-1} \Lambda^{c_i}_{ij,ij}(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})^{-1} \Lambda^{c_j}_{ij,ij} \\
&\qquad + \Lambda^{c_i}_{ij,ij}(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})^{-1} \Lambda^{c_j}_{ij,ij}(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})^{-1} \Lambda^{c_i}_{ij,ij}\}^{-1} \\
&= \{(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})^{-1} \Lambda^{c_i}_{ij,ij}(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})^{-1} \Lambda^{c_j}_{ij,ij}\}^{-1} \\
&= \{\Lambda^{c_i}_{ij,ij}(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})^{-1} \Lambda^{c_j}_{ij,ij}\}^{-1} \\
&= (\Lambda^{c_j}_{ii,ij})^{-1}(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})(\Lambda^{c_i}_{ij,ij})^{-1} \\
&= (\Lambda^{c_i}_{ii,ij})^{-1} + (\Lambda^{c_j}_{ij,ij})^{-1} \ .
\end{aligned}
\tag{21}
$$

In the above derivation, we use the fact that covariance matrices are symmetric to simplify the equation. We also assume that the cross correlation between $P^{c_i}_{ij}$ and $P^{c_j}_{ij}$ is small comparing to the auto correlations, hence the cross correlation terms are neglected in the above derivation. Similarly, one can show that

$$
\begin{aligned}
\Lambda^{c_i'}_{ii,ij} &= \Lambda^{c_i}_{ii,ij}\Lambda^{c_j}_{ij,ij}(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})^{-1} \\
\Lambda^{c_j'}_{ii,ij} &= \Lambda^{c_j}_{ii,ij}\Lambda^{c_i}_{ij,ij}(\Lambda^{c_i}_{ij,ij} + \Lambda^{c_j}_{ij,ij})^{-1} \ .
\end{aligned}
\tag{22}
$$

$\Lambda^{c_i'}$ and $\Lambda^{c_j'}$ can then be assembled using Equations 19, 21, and 22. Equations 20, 21, and 22 represent what is usually called the maximum likelihood estimate, or the weighted least-square estimate [33]. Under the assumptions that the noise process is white and Gaussian, the weighted least-square estimator gives the "best" estimation, in the sense that its variance is less than that of any other linear unbiased estimator [33]. Interested readers are referred to [16],[33] for the proof

of the above claim.

For a spline surface patch, the above procedure has to be applied to the patch's boundaries with all its neighbors. Basically, we divide the control vertices of a patch into nine classes: those belonging only to the patch, those shared with the north, south, east, and west neighbors, and those also shared with the northeast, northwest, southeast, and southwest neighbors. The boundaries adjustment procedures are then applied to all shared vertices using appropriate neighbors.

3.3 State Propagation

One can notice from Equation 2 that we define an imaged surface structure as a function of both space and time, which implies that the underlying control vertex structure is also spatially and temporally variant. Propagation of the surface structure in both spatial and temporal domains are therefore possible. Again, we use Kalman filter formulation to make predictions on the imaged surface structure at a location at a certain time instance from that at the same or a different location at the same or an earlier time instance. Such structure prediction may be necessary because:

- If the union of the fields of view of all sensors does not cover the whole viewing universe, portions of the imaged object may not be represented (or hidden from all sensors). As the goal is to construct a complete 3D surface description, the structure parameters of the hidden surface, i.e., its control vertex positions, need to be estimated. Such prediction can be based on specific domain knowledge (for example, the imaged object is symmetrical about an axis) or on certain reasonably general shape assumptions (for example, the smoothness assumption). If a prediction framework can be translated into shape constraints—with an associated confidence measure—on the hidden surface structure, then Kalman filter can be used to incorporate such predictions in building a complete 3D model.
- If the imaged object is flexible or is moving, temporal changes of the surface structure do occur. It is not feasible to reconstruct a 3D description from scratch each time the imaged object moves or deforms. A more efficient structure update is to track the motion and deformation of the imaged object over time by propagating the state vector of the spline representation.

In the terminology of Kalman filtering, the state vector at a spatial location at one time instant can be related to that at the same or a neighboring location at the same or a later time instant. We will assume that such a state propagation is linear and is described by a state transition equation in a matrix form with an associated confidence measure,

$$\mathbf{P}_{u',v'}(t') = \mathbf{M}(u', v', t', u, v, t)\mathbf{P}_{u,v}(t) \ . \tag{23}$$

Then the knowledge of $\mathbf{P}_{u,v}(t)$ (e.g., the control vertices of a sensed surface patch at time t) can be used to estimate $\mathbf{P}_{u',v'}(t')$ (e.g., the control vertices of a hidden surface patch at a later time t').

Spatial Propagation of the State Vector In our formulation, the state transition equation is kept linear by restricting such transition to be composed of translation and rotation (i.e., an affine transform). It is well known that the B-spline patches are invariant under the affine transformation [37], i.e., translating and rotating the polygon net composed of the control vertices result in the identical transformation being applied to the spline surface. Hence, a linear affine transform is basically a "similarity" transform.

Currently, we allow the state transition equation to be designed based on (1) the symmetry assumption and (2) the smoothness assumption, both of which can be represented in the matrix form shown in Equation 23. The state transition based on symmetry requires the specification of a symmetry relation in the parameter plane (u, v). For example, suppose that the object is symmetrical about the v axis in the parameter plane, and the u dimension ranges from 0 to 2π. Then a point $\mathbf{S}$ and its image $\mathbf{S}$'—created by rotating $\mathbf{S}$ through an angle π around an axis of direction $\mathbf{S}_v$ and passing though the centroid $\mathbf{O}$ of the object—should both be on or not on the surface. Hence, if $\mathbf{S}$ is on a sensed surface patch, then $\mathbf{S}$' should also be on the object

$$\mathbf{S}' = \left[\begin{array}{lll} S_{v_x}^2 + (1 - S_{v_x}^2)cos\Omega & S_{v_x}S_{v_y}(1 - cos\Omega) + S_{v_z}sin\Omega & S_{v_x}S_{v_z}(1 - cos\Omega) - S_{v_y}sin\Omega \\ S_{v_x}S_{v_y}(1 - cos\Omega) - S_{v_z}sin\Omega & S_{v_y}^2 + (1 - S_{v_y}^2)cos\Omega & S_{v_y}S_{v_z}(1 - cos\Omega) + S_{v_x}sin\Omega \\ S_{v_x}S_{v_z}(1 - cos\Omega) + S_{v_y}sin\Omega & S_{v_y}S_{v_z}(1 - cos\Omega) - S_{v_x}sin\Omega & S_{v_z}^2 + (1 - S_{v_z}^2)cos\Omega \end{array} \right]_{\Omega=\pi} (\mathbf{S}-\mathbf{O})+\mathbf{O}. \tag{24}$$

Coordinates of $\mathbf{S}'$ can be treated as a depth measure—just like an observation from an external position sensor in Equation 3—and are filtered down the structure tree to propagate the symmetry constraint.

State transition equation can also be writtened based on the smoothness assumption. If the imaged surface structure is smooth and of a low order over a small neighborhood, then the local surface structure can be faithfully approximated by a planar, a spheric, or a cylindrical patch. It can be easily shown that the transition of control vertices on such simple surfaces is affine. For example, transition of control vertices on a planar surface involves a simple translation of the control vertices, while that on a spheric surface involves a simple rotation. Transition of control vertices on a cylindrical surface involves both translation and rotation.

Hence, we first identify the primitive surface type a B-spline patch most resembles. We then compute the shape parameters (the normal to a plane, the center and radius of a sphere, the axis of a cylinder, etc.) of the patch. Based on the computed shape parameters, we compute the surface locations adjacent to the patch under consideration. The computed surface point locations can again be treated as external depth measures and are filtered down the structure tree to propagate the smoothness constraints.

The surface type classification is based on the principal surface curvatures and Gaussian curvature computed during the state subdivision. It is well known [21],[41] that for planar surface $\kappa_1 = \kappa_2 = 0$, for spherical surface $\kappa_1 = \kappa_2 \neq 0$, and for cylindrical surface $\kappa_1 \neq 0$ and $\kappa_2 = 0$.

Hence, patches are classified into the three primitive types using their curvature signatures. Only those patches which bear a large degree of resemblance to a primitive surface (i.e., they have the curvature signatures discussed above with a small curvature variation) are used to propagate the smoothness constraints.

Shape parameters are then computed. For example, if a B-spline patch is classified as cylindrical, the radius ($1/\kappa_1$) and the axis direction (the direction of κ_2) are computed during the state subdivision. To compute the location of the axis, we observe that the axis of the cylinder is at a distance $R = 1/\kappa_1$, in the direction of $-\mathbf{S}_u \times \mathbf{S}_v$, from a surface point $\mathbf{S}$. Therefore, each surface point $\mathbf{S}$ votes for a point on the axis by

$$\mathbf{S} - (\mathbf{S}_u \times \mathbf{S}_v)/(\kappa|\mathbf{S}_u \times \mathbf{S}_v|). \tag{25}$$

Ideally, the cylinder's axis should pass through all these points and has the direction of κ_2. Hence, a least-square estimation determines the location of the axis. With the shape parameters of the cylinder computed, surface points adjacent to the cylindrical patch can be generated to propagate the smoothness constraint.

A word of caution on the state propagation: even though general domain knowledge such as the symmetry assumption and smoothness constraint can be helpful in building a complete model, its validity must be taken with a few grains of salt. If it is discovered that the imaged object does not possess a symmetrical structure after the constraints imposing the symmetry assumption have been applied, the effect may not be undone. On the other hand, if the object does possess a symmetrical structure, the state propagation will enforce the symmetry constraint and may even correct errors in the sensor data which make the structure deviating from a symmetrical one. As the effect a shape constraint has on the final structure is weighted by its error covariance, the constraints derived from the domain-specific assumptions should be given a large error covariance. Hence, the structure imposed by the domain-specific assumptions will prevail only if no other, more reliable sensor information is available. When sensor data are made available, the image constraints should be weighed more heavily to update the structure. To summarize, the state propagation scheme is a convenient tool provided with the algorithm, whether such a tool is applicable (e.g., whether the imaged object possesses a symmetrical structure for the state propagation based on symmetry to be valid) is left to the decision of the user.

4 Experimental Results

4.1 Computer Simulation

To gain a better understanding of the accuracy and applicability of the proposed algorithm, computer simulation was conducted using synthetic data. Purpose of the simulation was to obtain

a statistical evaluation of the performance of the modeling algorithm and to test the recursive Kalman filtering and state subdivision processes.

A simulation was conducted to model the structure of a sphere. The modeling program was supplied with the position and orientation measurements at random sampling locations on the surface of a sphere. The 3D shape measurements were perturbed by random noise to simulate the uncertainty in the real image data. The modeling program started with an initial surface representation shown in Figure 6—which was a B-spline patch with 7×7 control vertices and happened to be the same initial structure used in the experiments with the real image sequences (in Section 4.2). The 3D shape measurements were incorporated iteratively to estimate the surface structure using the Kalman filter formulation presented in Section 3. Further, the program was allowed to employ the state subdivision process (Section 3.2) to introduce more patches into the representation to verify if a more faithful representation could be achieved.

The simulation was carried out in the following manner: Several input parameters which determined the conditions of the simulation were specified. These parameters were the number of position measurements, the number of orientation measurements, the maximum perturbation in these shape measurements, and the maximum depth of subdivision. In the case of the position measurements, a sampled point was randomly perturbed by a distance which was a small percentage of the sphere's radius along the surface normal direction, and in the case of the orientation measurements, the surface normal at a sampled point was displaced by a small angular error.

The accuracy of the algorithm was established under a variety of simulation conditions—with (i) different amounts of error in surface measurements and (ii) different numbers of patches used in the representation—by averaging the results from different runs of the simulation. In our simulation, the number of shape measurements used in each data set ranged from 50 to 150, the magnitude of the additive position noise ranged from 0% to 12% of the sphere's radius, and the magnitude of the additive orientation noise ranged from 0^o to 5^o. For each simulation run, we computed the spline surface position based on the estimated control vertex positions, and compared it with the true position of the sphere. An average structural error, as a percentage of the sphere's radius, was computed for each data set. For each error magnitude setting, we tested the algorithm on ten to twenty data sets and averaged the results. The simulation results are shown in Figures 2 and 3.

In Figure 2, we show the simulation results using the positional measurements alone. As expected, the average structural error increased as the perturbation in the simulated image data increased. The slow rate of increase of the structural error may be attributed to the smoothing effect of the spline functions. An interesting thing to note was that when refinment was allowed—hence more patches were used—more faithful representations were obtained. As observed in Figure 2, the error curves corresponding to larger numbers of state divisions dipped lower in the plot. For a simple structure like a sphere, very fine subdivisions might be excessive. As shown in Figure 2,

curve D, with 17 patches and each patch defined on a 7 $\times$7 grid, was similar in shape and position to curves B and C. When the synthetic error increased, a 17-patch representation (curve D) did perform better, though not significantly, than a 5-patch representation (curve B), which lead us to suspect that a point of diminishing return was reached. However, the refinment scheme definitely helped when complicated real-world objects were used (e.g., those shown in Figures 4 and 9).

In Figure 3, we show how additional shape measurements can improve an existing representation. Basically, we continued with the simulation by incorporating the orientation measurements (with a 2^o maximum angular deviation) for the structural update. Hence, we first estimated the imaged structure using the positional measurements, (the results were depicted in Figure 2), then updated the surface structure by incorporating additional orientation measurements using Kalman filtering. As can be seen by comparing plots in Figures 2 and 3, improvements were made in the average structural error by fusing heterogeneous sensor data.

4.2 Experiments with Real Image Sequences

We also conducted several experiments using real image sequences. We used two different sensors—visual sensors and structured light sensors. These sensors were positioned at up to three different directions around the imaged objects. Depth constraints derived from stereopsis, and orientation and curvature constraints derived from structured lighting were used in the 3D modeling process.

The imaging configuration is described below: The imaged objects were placed on a table which was about 50" in front of the camera. To generate the stereo image pairs, we slid the camera on an optical rail by about 4" and took two snapshots of the objects. The objects were then rotated to reveal different views to the camera. To generate the structured light images, we used a projector to project a regular grid pattern marked on a $35mm$ slide. The grid pattern consisted of two sets of parallel stripes: each stripe set made a 45^o angle with respect to the ground and they were mutually orthogonal. The slide projector was placed at about 60" from the objects. The distance from the camera to the projector (or the baseline separation) was about 24". The focal axes of both the camera and the projector were parallel to the ground and made a 30^o angle.

Figures 4 (a) through 4 (f) show the stereo image pairs taken from three different viewing directions around a ceramic pen holder in the shape of a bird. Figures 5 (a) through 5 (c) show the structured light images taken from the same viewing directions as those in the stereo imaging, with the structured light images registered to the left images in the stereo pairs.

Depth constraint was derived from the stereo analysis of Figures 4 (a) through 4 (f). We used the stereo matching algorithm developed in [27] for the stereo analysis. Features used were different zero-crossing patterns in a 3×3 neighborhood. Multiple matches were resolved by a corporative relaxation process which imposed additional disparity and figural continuity requirements. 3D

surface positions were then derived through triangulation.

The structured-light images in Figures 5 (a) through 5 (c) were analyzed to infer the orientation and curvature of the imaged surface at each stripe junction. Intuitively speaking, when a regular grid pattern was projected onto a curved surface structure, the perceived image pattern was distorted (e.g., see Figures 5 (a) to (c) and Figures 10 (a) to (c)). The distortion was induced by the underlying surface structure, hence, it should be possible to relate the pattern distortion to the structure of the imaged object. In [46],[49], we developed methods for inferring *3D surface* orientation and curvature from *2D image pattern* orientation and curvature. Such techniques made available orientation and curvature constraints for 3D shape modeling.

In more details, we presented a technique for computing surface orientation using structured lighting in [46]. The technique computed surface orientation by taking cross product of the direction cosines of the projected stripes on the imaged object's surface. The direction cosines of the projected stripes were computed using the stripe orientations in the image and the grid projection planes. To compute principal surface curvatures and their directions using the structured lighting, we again projected a pattern to encode the imaged object surface for analysis. At a grid junction, curvatures of the projected stripes on the imaged object's surfaces were computed. The computed curvatures were then related to those of the normal sections which shared the same tangential direction as the projected curves using the Meusnier's theorem in differential geometry [21],[41]. The principal curvatures and their directions at the grid junction under consideration were then recovered using Euler's theorem [21],[41]. Details can be found in [49].

We started with a B-spline surface defined on a 7×7 grid shown in Figure 6. We iteratively incorporated shape cues from all sensors and all viewing directions to refine the surface structure. If the variance of Gaussian curvature of a patch was above a preset threshold, the patch was refined by breaking it into four component patches. The control vertex positions and error covariance matrices of component patches were computed from those of the original patch using the technique introduced in Section 3.2. A hierarchical quadtree data structure was constructed this way. Each time sensor data were made available, the data were filtered down the tree and picked up by the patch where the constraints applied.

This structure construction and update process for the ceramic bird is illustrated in Figure 8. The first row in Figure 8 ((a), (d), and (g)) shows the structures after the shape cues from the first view were incorporated, and the second row ((b), (e), and (h)) and the third row ((c), (f), and (i)) show the structures after the shape cues from the second and the third views were incorporated. The first column in Figure 8 ((a), (b), and (c)) show the structures without subdivision (i.e., only the initial 7×7 patch was used), while the second and third columns show the structures with subdivision. By comparing the first column with the other two columns in Figure 8, we see that much more faithful approximations to the imaged structure were obtained using refinement. By

comparing the first row with the second and third rows in Figure 8, we observe how additional structures were captured and represented by processing additional views. For example, Figures 8 (d) and (g), which show the structures after the information from the first view was incorporated, do not show the tail structure. This was because the tail was not visible from the first viewing direction. Also, the structures of the neck and head which were not visible from the first view were not updated (hence, the back side of the model stayed in the original cylindrical shape). After the second view was processed, the tail structure started to appear. However, by comparing Figures 8 (e) and (f), one see that the tail structure was not faithfully modeled in Figure 8 (e) because the width of the tail was not determined unless the third view was processed. Also, Figure 8 (e) does not show the concave structure of the neck as viewed from the third direction (However, the structure of the neck visible from the first and second views was correctly modeled, see Figure 8 (h)). Finally, as depicted in Figures 8 (f) and (i), the shapes of the tail and neck were better approximated when the third view was processed.

Another example, using a soap dispenser, is shown in Figures 9 through 11. We started with the same initial structure in Figure 6 and graduately refined the structure. As can be seen in Figure 11, most of the structure refinement occurred at the area around the sprout where the structure was most complicated. The state refinement process was again useful as can be seen in Figure 11 (a), the original patch was not able to represent the sprout structure faithfully.

The final example demonstrates the use of the state propagation. Figure 12 (a) shows the image of a wine glass, which possesses a symmetrical structure about its vertical axis. Since the structure was known to be symmetrical, it should be possible to construct a complete 3D description with a minimum amount of image constraints. Hence, instead of taking pictures from three different directions as in the previous examples, we took only one pair of stereo images, and the left image of the stereo pair is shown in Figure 12 (a). The depth constraints obtained from the stereo analysis were duplicated using Equation 24, and the occluded structure was predicted by propagating the state vector of the visible structure. The structures before state propagation and after state propagation are shown in Figures 12(b) and (c), respectively. As can be seen that Figure 12 (b) captured the visible structure well, but the occluded structure was not updated. In Figure 12 (c), we observe that by propagating the state vector, we were able to construct a complete 3D description, even when part of the surfaces was occluded. These experimental results demonstrate the potential of our algorithm in 3D modeling using heterogeneous sensor data.

5 The Concluding Remarks

To summarize, the use of Kalman filtering on hierarchical B-spline surface provides a new framework for integrating multiple sensory data. The ability to adapt to different imaged structure complexity, to maintain high-order surface smoothness, to propagate the structure in the spatial domain, and to

integrate the least-square estimation with the spline theory are powerful mechanisms which should prove useful in 3D shape modeling.

Currently, we are investigating how the state vector can be propagated through time. Such propagation is necessary when the imaged object either moves or deforms. For propagating a B-spline surface's state vector over time, it is important that certain internal constraint on the propagation be formulated. That is, it is not sufficient just to predict the possible motion or deformation of the imaged object based on the current and previous motion and deformation parameters. (Certainly, such prediction should be performed using the same Kalman filtering technique.) It is also needed that certain integrity constraints be imposed on the temporal propagation. For example, if the object is known to be rigid, propagation of the state vector must maintain the rigidity of the resulting structure. If the object is known to be flexible but not stretchable (e.g., a piece of paper), propagation of the state vector must maintain the resulting Gaussian curvature.

Our future research plan also includes investigating methods for generating realistic animation sequence of a dynamic modeling process. We conjecture that deforming a B-spline surface is equivalent to *deforming* the defining polygon net. Since the defining polygon net is a rectangular grid structure, the reconstruction equation developed in [48]—which also operates on a grid structure—is readily applicable. Hence, realistic animation sequences can be simulated.

References

[1] J. K. Aggarwal and C. H. Chien, "3-D Structures from 2-D Images," in *Advances in Machine Vision*, Springer-Verlag, 1989, pp. 64-121.

[2] J. K. Aggarwal and M. J. Magee, "Determining Motion Parameters Using Intensity Guided Range Sensing," *Pattern Recognition,* vol. 19, no. 2, 1986, pp. 169-180.

[3] J. K. Aggarwal and Y. F. Wang, "Sensor Data Fusion in Robotics Systems," in *Advances in Control and Dynamic Systems,* edited by C. T. Leondes, Academic Press, 1991, pp. 435-462.

[4] J. K. Aggarwal and Y. F. Wang, "Sensors and Sensor Fusion," in *Encyclopedia of Artificial Intelligence, 2nd edition*, edited by Stuart C. Shapiro, John Wiley & Sons, Inc., New York, vol. 2, 1992, pp. 1511-1526.

[5] J. H. Ahlberg, E. N. Nilson, and J. L. Walsh, *The Theory of Splines and Their Applications*, Academic Press, New York, NY, 1967.

[6] P. Allen, *Robotic Object Recognition Using Vision and Touch,* Kluwer, MA, 1988.

[7] P. Allen and R. Bajcsy, "Two Sensor Are Better Than One: Example of Integration of Vision and Touch," *Proceedings of the 3rd International Symposium on Robotics Research*, Gouvieux, France, 1986, pp. 59-64.

[8] P. Allen and P. Michelman, "Acquisition and Interpretation of 3-D Sensor Data from Touch," in *Proceedings of the Workshop on Interpretation of 3-D Scene*, Austin, TX, no. 27-29, 1989, pp. 33-40.

[9] M. Asada, "Building a 3-D World Model for a Mobile Robot from Sensory Data," *Proceedings of the IEEE International Conference on Robotics and Automation*, Philadelphia, PA, Apr. 24-29, vol. 2, 1988, pp. 918-923.

[10] N. Ayache, *Artificial Vision for Mobile Robots: Stereo Vision and Multisensory Perception*, MIT Press, Cambridge, MA, 1991.

[11] B. A. Barsky, *Computer Graphics and Geometric Modeling Using Beta-Splines,* Springer-Verlag, Berlin, 1988.

[12] K. J. Bathe, *Finite Element Procedures in Engineering Analysis,* Prentice-Hall, Englewood Cliffs, NJ, 1982.

[13] S. D. Blostein and T. S. Huang, "Error Analysis in Stereo Determination of 3-D Point Positions," *IEEE Trans. PAMI*, vol. 9, no. 6, Nov. 1987, pp. 752-765.

[14] M. Brady, "Computational Approaches to Image Understanding," *ACM Computing Surveys*, vol. 14, no. 1, 1982, pp. 3-71.

[15] M. Brady, "Foreword, Special Issue on Sensor Data Fusion," *International Journal of Robotics Research*, vol. 7, no. 6, 1988, pp. 2-4.

[16] R. G. Brown, *Introduction to Random Signal Analysis and Kalman filtering*, Wiley, New York, NY. 1983.

[17] C. C. Chu and J. K. Aggarwal, "Image Interpretation using Multiple Sensing Modalities," *IEEE Trans. PAMI*, vol. 14, no. 8, Aug. 1992, pp. 840-847.

[18] A. K. Cline, "Scalar- and Planar-Valued Curve Fitting Using Splines Under Tension," *Communications of the ACM*, Vol, 17, pp. 218-220, 1974.

[19] E. Cohen, T. Lyche, and R. Riesenfeld, "Discrete B-Splines and Subdivision Techniques in Computer-Aided Geometric Design and Computer Graphics," *Computer Graphics and Image Processing*, vol. 14, pp. 87-111, 1980.

[20] M. J. Daily, J. G. Harris, and K. Reiser, "An Operational Perception System for Cross-Country Navigation," *Proceedings of the IEEE CVPR Conference*, Ann Arbor, MI, June 5-9, 1988, pp. 794-820.

[21] M. P. Do Carmo, *Differential Geometry of Curves and Surfaces,* Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1976.

[22] G. Farin, *Curves and Surfaces for Computer Aided Geometric Design,* Academic Press, San Diego, CA, 1988.

[23] D. R. Forsey and R. H. Bartels, "Hierarchical B-Spline Refinement," *Computer Graphics*, vol. 22, pp. 205-212, 1988.

[24] M. Gardiner, "The superellipse: A curve that lies between the ellipse and the rectangle," *Scientific America*, vol. 213, pp. 222-234, 1965.

[25] K. H. Huebner and E. A. Thornton, *The Finite Element Method for Engineers*, 2nd Edition, John Wiley & sons, New York, NY, 1982.

[26] T. Kanade, "The KISS Solutions of Shape Recovery Problems in Computer Vision," invited talk, *the 1992 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Raleigh, NC, Jul. 7-10, 1992.

[27] Y. C. Kim and J. K. Aggarwal, "Positioning Three-Dimensional Objects using Stereo Images," *IEEE Transactions on Robotics and Automation,* Aug. 87, 1987, pp. 361-373.

[28] D. Kuan, G. Phipps, and A-C Hsueh, "Autonomous Robotic Vehicle Road Following," *IEEE Transaction on PAMI*, vol. 10, no. 5, 1988, pp. 648-658.

[29] I. S. Kweon and T. Kanade, "High-Resolution Terrain Map from Multiple Sensor Data," *IEEE Transaction on PAMI*, vol. 14, no. 2, 1992, pp. 278-292.

[30] C. Loop and T. DeRose, "Generalized B-Spline Surfaces of Arbitrary Topology," *Computer Graphics*, vol. 24, pp. 347-356, 1990.

[31] L. Matthies and S. A. Shafer, "Error Modeling in Stereo Navigation," *IEEE Journal of Robotics and Automation,* vol. 3, no. 3, 1987, pp. 239-248.

[32] L. Matthies and A. Elfes, "Integration of Sonar and Stereo Range Data Using a Grid-Based Representation," *Proceedings of the IEEE International Conference on Robotics and Automation*, Philadelphia, PA, Apr. 24-29, vol. 2, 1988, pp. 727-733.

[33] P. S. Maybeck, *Stochastic Models, Estimation, and Control, vol. 1*, Academic Press, New York, NY, 1979.

[34] E. S. McVey and J. W. Lee, "Some Accuracy and Resolution Aspects of Computer Vision Distance Measurements," *IEEE Trans. PAMI*, vol. 4, no. 6, Nov. 1982, pp. 646-649.

[35] N. Nandhakumar and J. K. Aggarwal, "Integrated Analysis of Thermal and Visual Images for Scene Interpretation," *IEEE Transactions on PAMI*, vol. 10, no. 4, 1988, pp. 469-481.

[36] J. J. Rodriguez and J. K. Aggarwal, "Quantization Error in Stereo Imaging," *IEEE Proceedings of CVPR*, Ann Arbor, Michigan, June 5-9, 1988, pp. 153-158.

[37] D. F. Rogers and J. A. Adams, *Mathematical Elements for Computer Graphics* 2nd Edition, McGraw-Hill, New York, NY, 1990.

[38] R. A. Russell, "A Thermal Sensor Array to Provide Tactile Feedback for Robots," *International Journal of Robotics,* vol. 1, no. 3, 1985, pp. 35-40.

[39] S. A. Stansfield, "A Robotic Perceptual System Utilizing Passive Vision and Active Touch," *International Journal of Robotics,* vol. 7, no. 6, 1988, pp. 138-161.

[40] G. Strang, *Introduction to Applied Mathematics*, Wellesley-Cambridge Press, Cambridge, MA, 1986.

[41] D. J. Struik, *Differential Geometry*, 2nd Edition, Addison-Wesley, Reading, MA, 1961.

[42] D. Terzopoulos, "Integrating Visual Information from Multiple Sources," in *From Pixels to Predicates,*, edited by A. Pentland, Ablex, Norwood, NJ, 1986.

[43] C. Thorpe, M. H. Hebert, T. Kanade, and S. A. Shafer, "Vision and Navigation for the Carnegie-Mellon Navlab," *IEEE Transactions on PAMI*, vol. 10, no. 3, 1988, pp. 362-373.

[44] M. A. Turk, D. G. Morgenthaler, K. D. Gremban, and M. Marra, "VITS—A Vision System for Autonomous Land Vehicle Navigation," *IEEE Transactions on PAMI*, vol. 10, no. 3, 1988, pp. 342-361.

[45] A. Verri and V. Torre, "Absolute Depth Estimate in Stereopsis," *Journal of the Optical Society of America,* vol. 3, no. 3, 1986, pp. 297-299.

[46] Y. F. Wang, A. Mitiche and J. K. Aggarwal, "Computation of Surface Orientation and Structure of Objects Using Grid Coding," *IEEE Transactions on PAMI,* vol. PAMI-9, pp. 129-137, Jan. 1987.

[47] Y. F. Wang and J. K. Aggarwal, "Integration of Active and Passive Sensing Techniques for Representing Three-Dimensional Objects," *IEEE Journal of Robotics and Automation*, vol. 5, no. 4, 1989, pp. 460-471.

[48] Y. F. Wang and Jih-Fang Wang, "Surface Reconstruction Using Deformable Models With Interior and Boundary Constraints," *Proceedings of the third International Conference on Computer Vision,* Osaka, Japan, pp. 300-303, Dec. 4-7, 1990.

[49] Y. F. Wang, "Characterizing 3-D Surface Structures from Visual Images," *IEEE Transactions on PAMI*, vol. 13, pp. 52-60, Jan. 1991.

[50] Z. Yang and Y. F. Wang, "Error Analysis of 3D Shape Construction From Structured Lighting," submitted for publication in *Pattern Recognition*

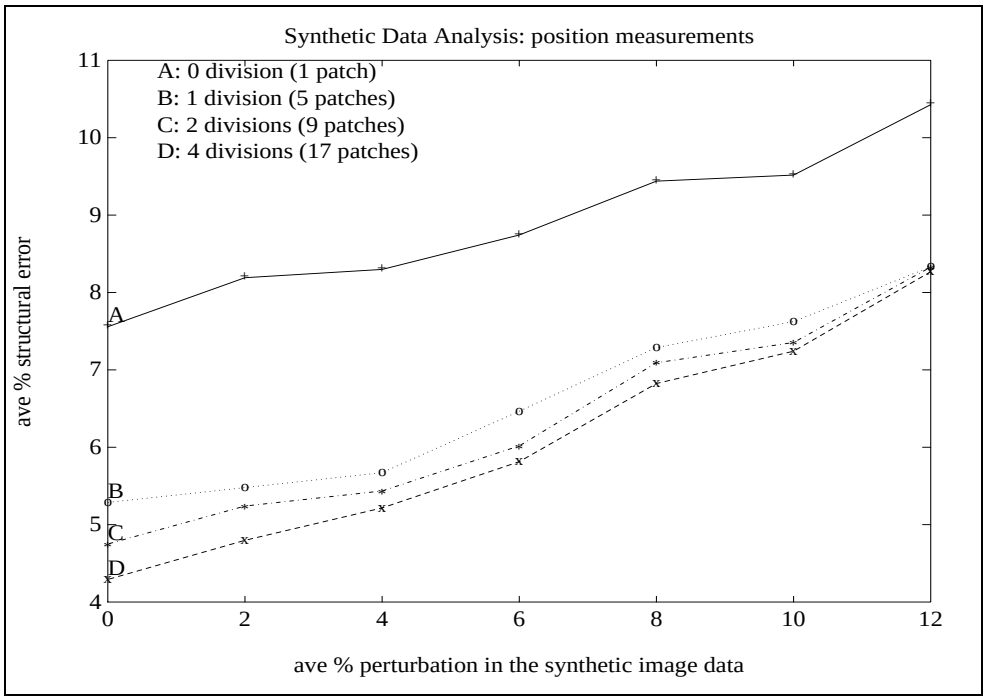


Figure 2: Simulation results: shape construction using position measurements

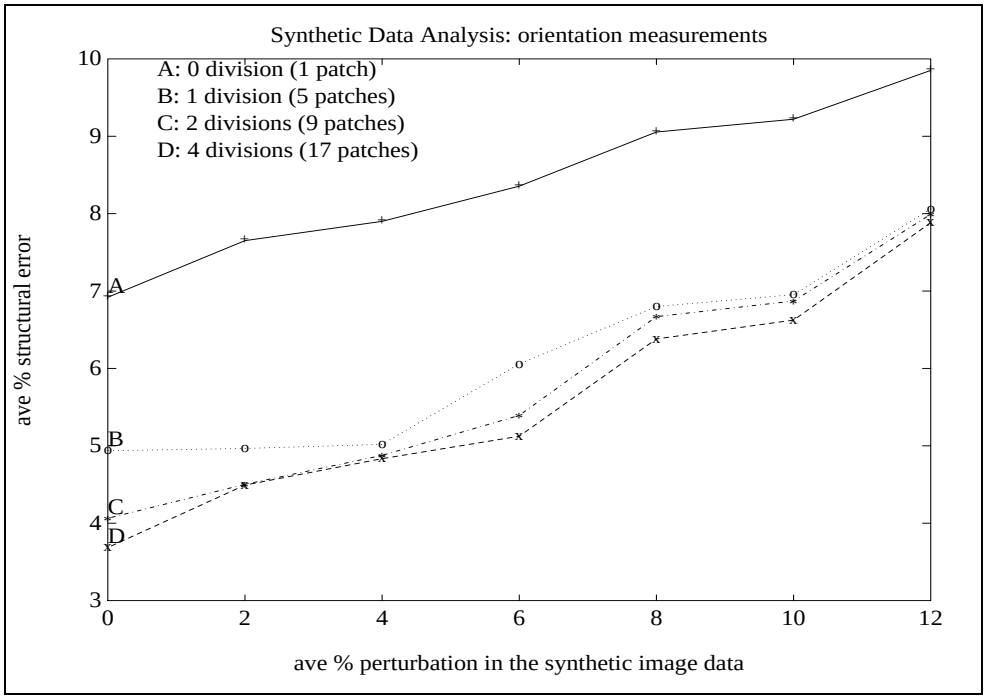
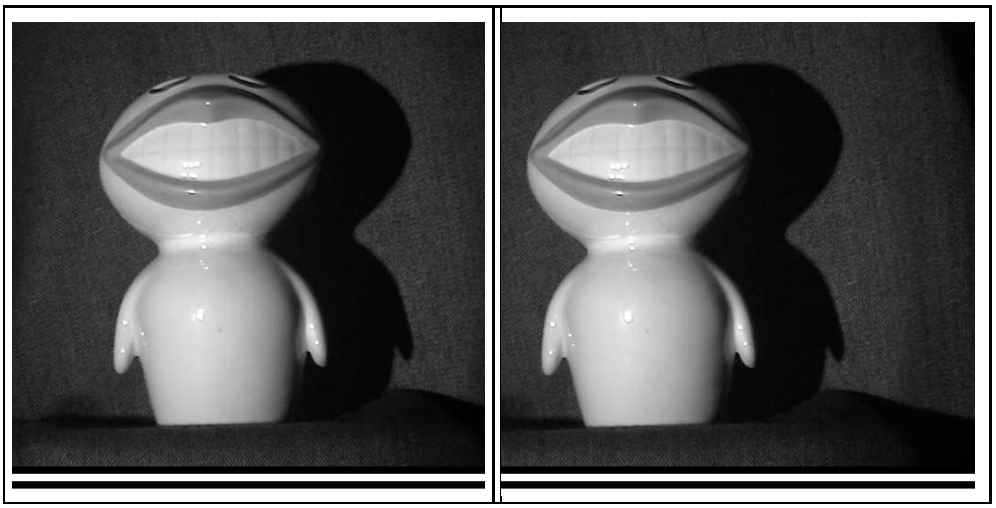
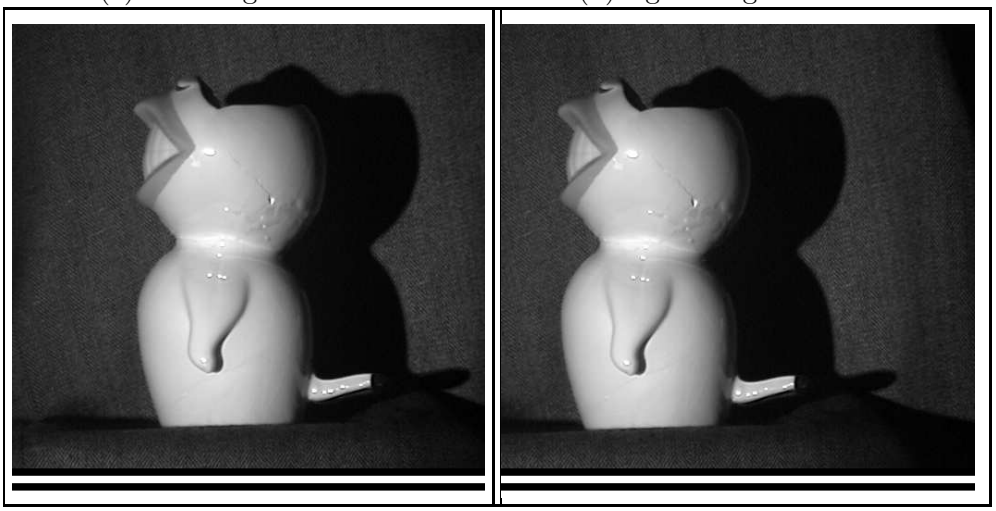


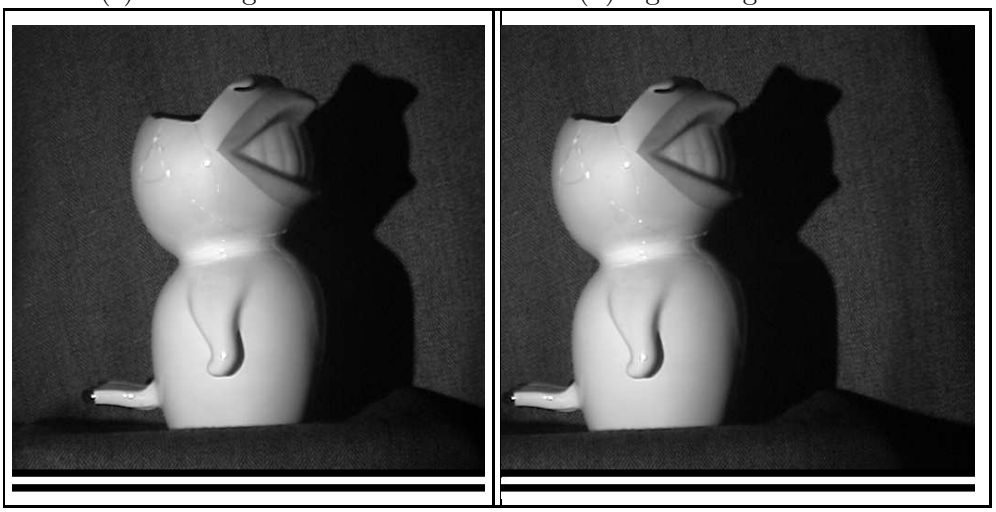
Figure 3: Simulation results: structural update using orientation measurements



(a) left image: frame 1 (b) right image: frame 1

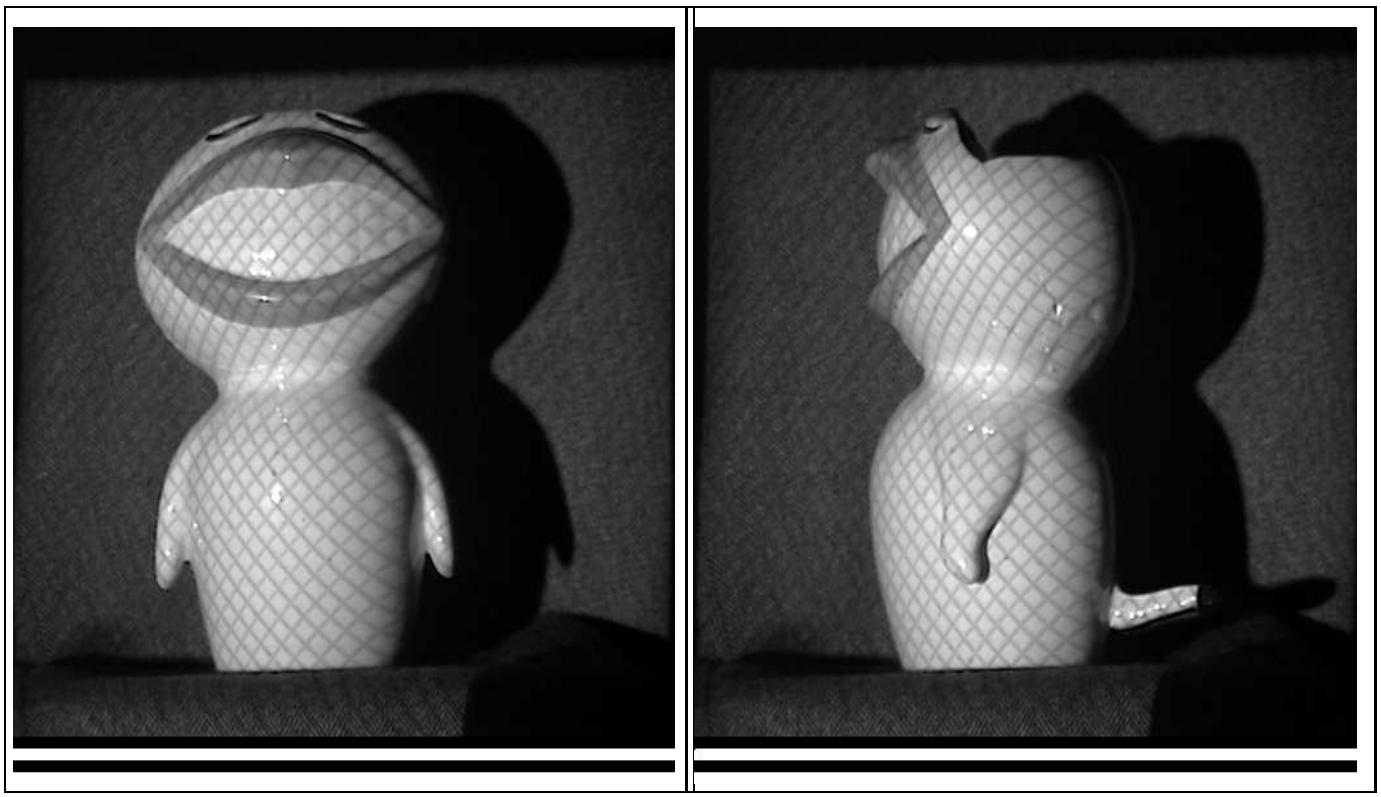


(c) left image: frame 2 (d) right image: frame 2



(e) left image: frame 3 (f) right image: frame 3

Figure 4: Stereo image pairs of a ceramic bird.



(a) structured light image: frame 1

(b) structured light image: frame 2

(c) structured light image: frame 3

Figure 5: Structured light images of the ceramic bird.

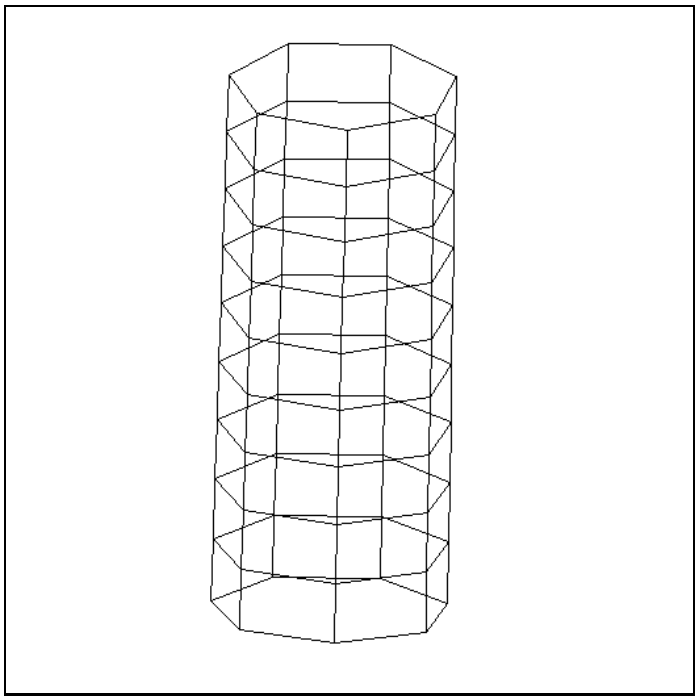


Figure 6: Initial surface structure of the ceramic bird.

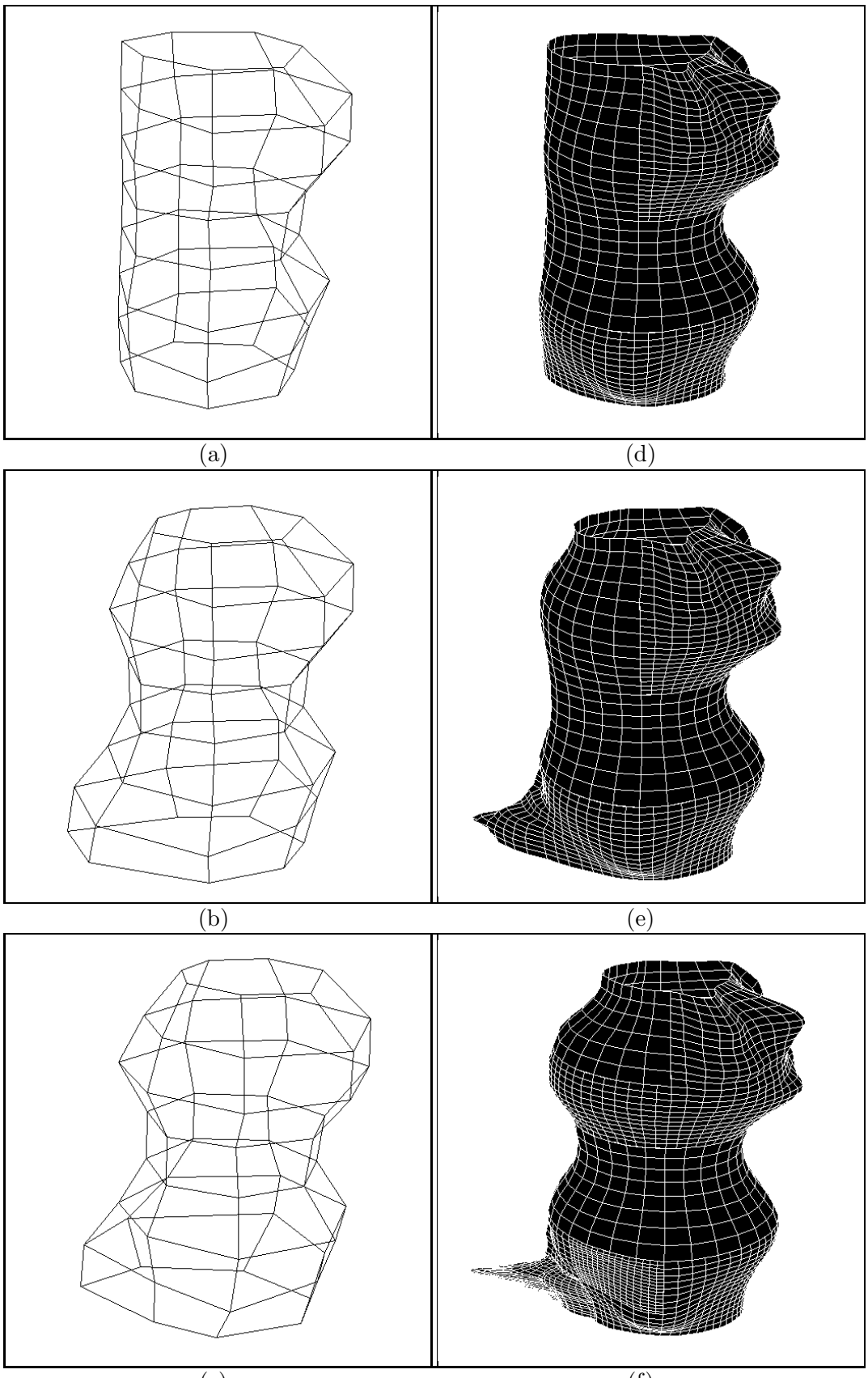
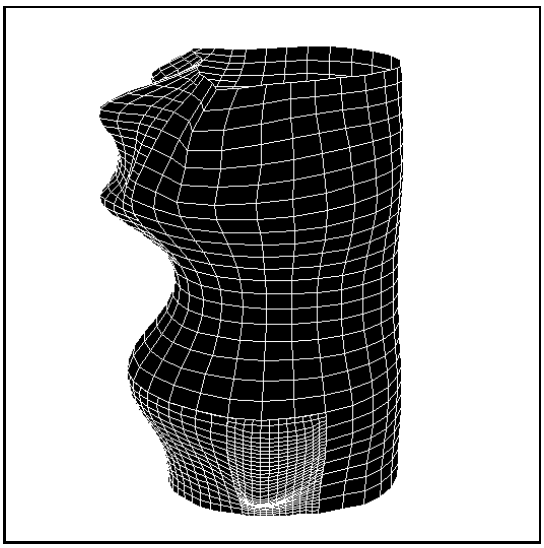
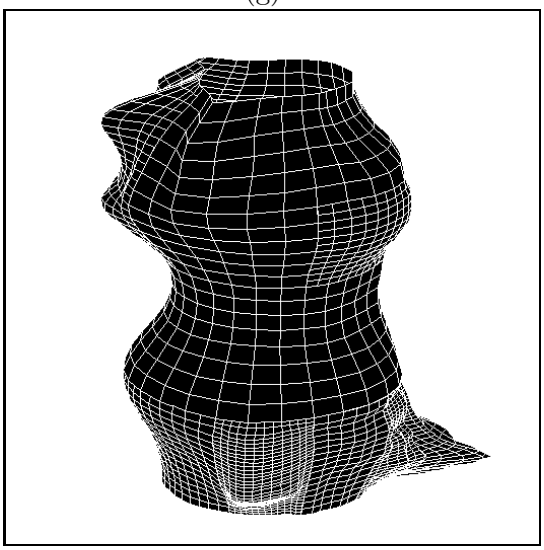


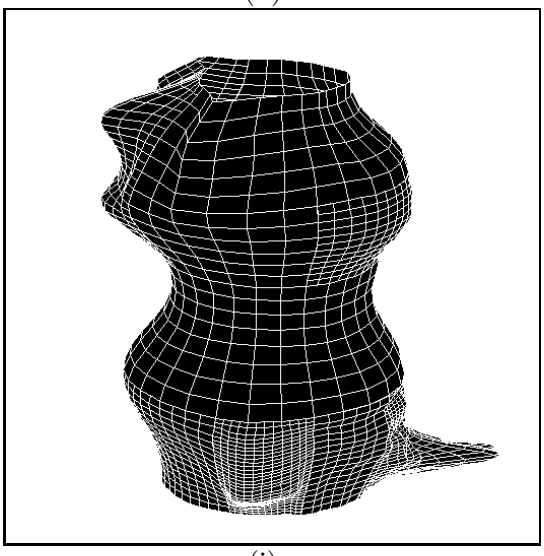
Figure 7: Reconstruction sequence of the ceramic bird.



(g)



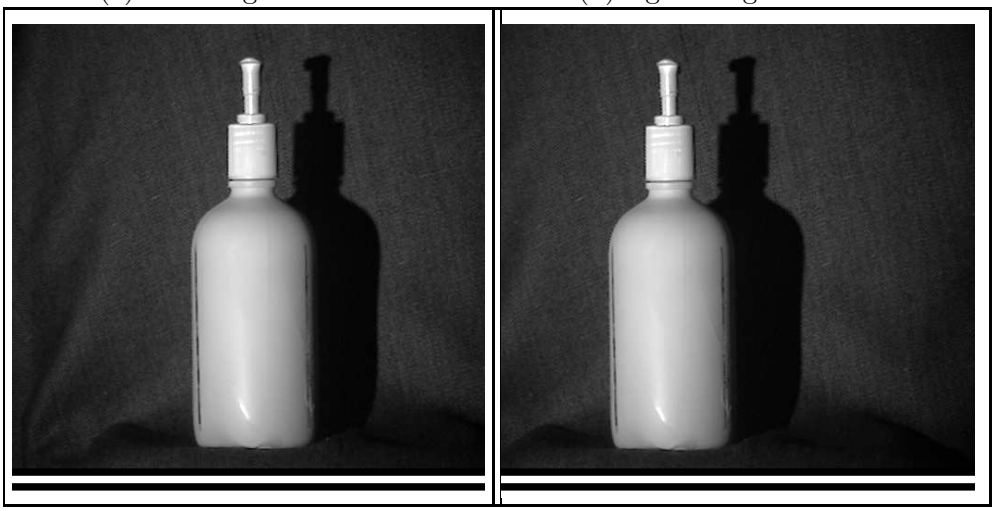
(h)



(i)

Figure 8: Reconstruction sequence of the ceramic bird.

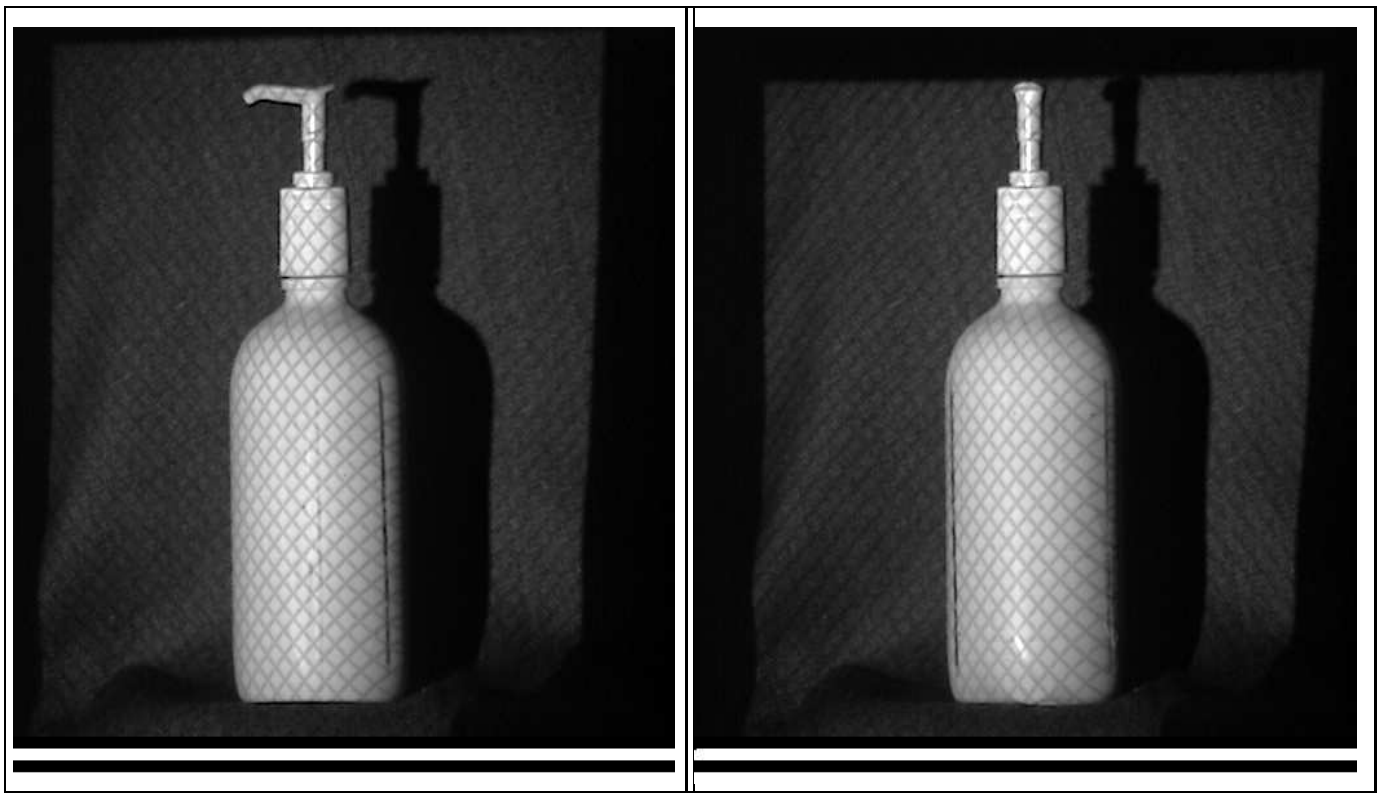
(a) left image: frame 1 (b) right image: frame 1



(c) left image: frame 2 (d) right image: frame 2

(e) left image: frame 3 (f) right image: frame 3

Figure 9: Stereo image pairs of a soap dispenser.

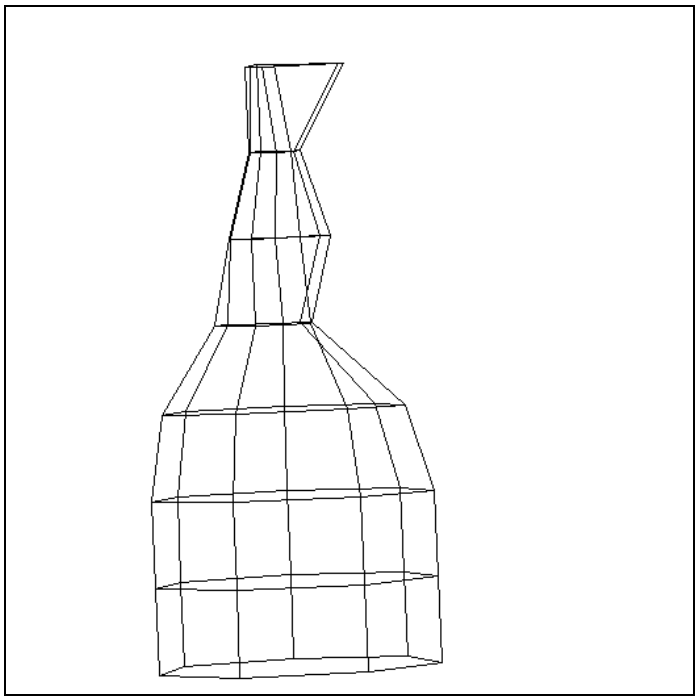


(a) structured light image: frame 1

(b) structured light image: frame 2

(c) structured light image: frame 3

Figure 10: Structured light images of the soap dispenser.

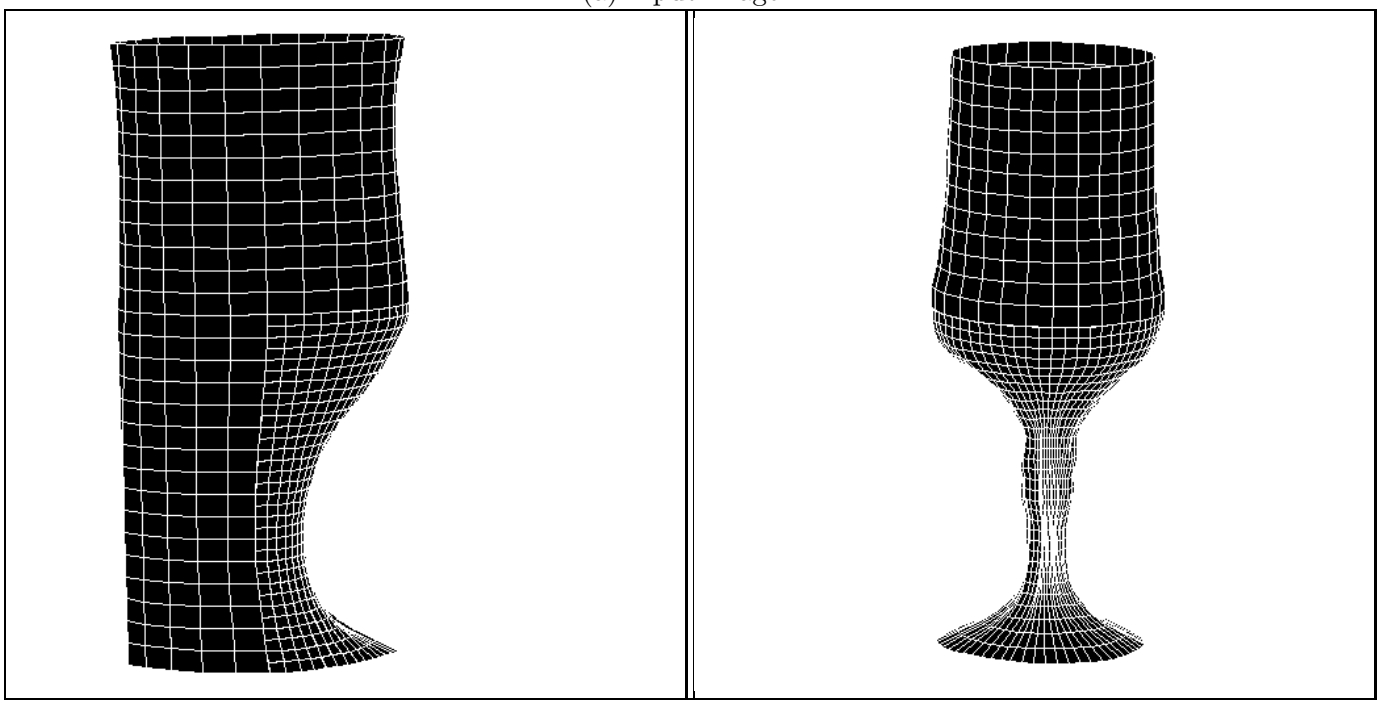


(a) root-patch structure

(b) structure with subdivision

Figure 11: Reconstruction surface structure of the soap dispenser.

(a) input image



(b) structure before state propagation (c) structure after state propagation

Figure 12: Reconstructed surface structure of the wine glass.