A New Path Planning Algorithm for Maximizing Visibility in Computed Tomography Colonography

Dong-Goo Kang and Jong Beom Ra*, *Senior Member, IEEE*

***Abstract*—In virtual colonoscopy, minimizing the blind areas is important for accurate diagnosis of colonic polyps. Although useful for describing the shape of an object, the centerline is not always the optimal camera path for observing the object. Hence, conventional methods in which the centerline is directly used as a path produce considerable blind areas, especially in areas of high curvature. Our proposed algorithm first approximates the surface of the object by estimating the overall shape and cross-sectional thicknesses. View positions and their corresponding view directions are then jointly determined to enable us to maximally observe the approximated surface. Moreover, by adopting bidirectional navigations, we may reduce the blind area blocked by haustral folds. For comfortable navigation, we carefully smoothen the obtained path and minimize the amount of rotation between consecutive rendered images. For the evaluation, we quantified the overall observable area on the basis of the temporal visibility that reflects the minimum interpretation time of a human observer. The experimental results show that our algorithm improves visibility coverage and also significantly reduces the number of blind areas that have a clinically meaningful size. A sequence of rendered images shows that our algorithm can provide a sequence of centered and comfortable views of colonography.**

***Index Terms*—CT colonography, path planning, temporal visibility, virtual flythrough.**

I. Introduction

COLON CANCER is one of the major causes of cancerous death, and colonic polyps are the main cause of colon cancer [1]. Recently, by virtue of the advance of computed tomography (CT) and computer graphics technology, CT colonography has become an important tool for screening colonic polyps. This is mainly because CT colonography, compared to real colonoscopy, minimizes the pain of the patient, and reduces the cost, the risk of perforation, and the limitation on the control of the camera [2]. Among various visualization methods for CT colonography, the virtual flythrough method is an intuitive and popular method that simulates real colonoscopy [3]–[6]. In the virtual flythrough method, polyps are examined in a sequence of rendered images that are taken in the lumen of the colon.

Before using the virtual flythrough method, a camera path should be planned. Automatic path planning is needed because manual planning is difficult and time-consuming due to the complex shape of the human colon. For complete and accurate diagnosis, a planned path should not produce significant blind areas on the colon surface. However, a recent study shows that with existing path planning algorithms more than 20 percent of the colon surface is in a blind area [7].

One approach for reducing the blind area is to devise a new visualization method instead of using the traditional flythrough method. In techniques that involve the virtual unfolding of a colon, the wall of a colon is spread on a flat sheet to enable visualization of most of the colon surface [8], [9]. The spreading process, however, induces distorted shapes that can badly affect the accuracy of diagnosis due to wrong interpretation of the image. In another visualization method, the unfolded cube method, six different views from a camera positioned on a central path are simultaneously displayed [10]. Here, the six views correspond to the six faces of a virtual cube centered at the camera position. The six views are overlaid on the unfolded cube by means of image-based rendering. However, because the size of the displayed image is six times lager than the image in the traditional flythrough method, the unfolded cube method requires more time for rendering and interpreting the images.

In another approach for reducing the blind area, path planning is improved to maximize visibility in the flythrough. Path planning is a procedure for determining the sequence of camera-pose parameters, which are composed of view positions referring to the camera location, view directions referring to the line of sight, and up-vectors referring to the upper direction of the image acquired from a camera or displayed on the screen. These three parameters determine a unique pose of the camera and the corresponding image. In existing path planning methods, the centerline is regarded as the best camera position for efficient and comfortable navigation [11]–[23]. For years, these algorithms have focused on improving the centeredness, robustness, and execution speed of the centerline extraction. In addition, tangential vectors on the centerline are generally used as view directions, and the decision method of up-vectors has been out of favor in other research. When navigating a straight tube having uniform thickness, the conventional centerline path produces no blind area. However, it may produce significantly large blind areas when navigating a human colon object that includes the two following inherent structures: a human colon with haustral folds and a colon with highly curved parts. Fig. 1 depicts the colon with haustral folds. These folds often cause blind areas for unidirectional navigation, while the blind areas are mostly exposed by bidirectional navigation with a round-trip path (see Fig. 1) [7]. On the other hand, a blind area may exist even with bidirectional navigation when

Manuscript received August 5, 2004; revised April 4, 2005. The Associate Editor responsible for coordinating the review of this paper and recommending its publication was W. Higgins. *Asterisk indicates corresponding author.*

D.-G. Kang is with the Department of Electrical Engineering and Computer Science, Korea Advanced Institute of Science and Technology, 373-1 Guseong-dong, Daejeon 305-701, Korea (e-mail: dgkang@issserver.kaist.ac.kr).

*J. B. Ra is with the Department of Electrical Engineering and Computer Science, Korea Advanced Institute of Science and Technology, 373-1 Guseong-dong, Yuseong-gu, Daejeon 305-701, Korea (e-mail: jbra@ee.kaist.ac.kr).

Digital Object Identifier 10.1109/TMI.2005.850551

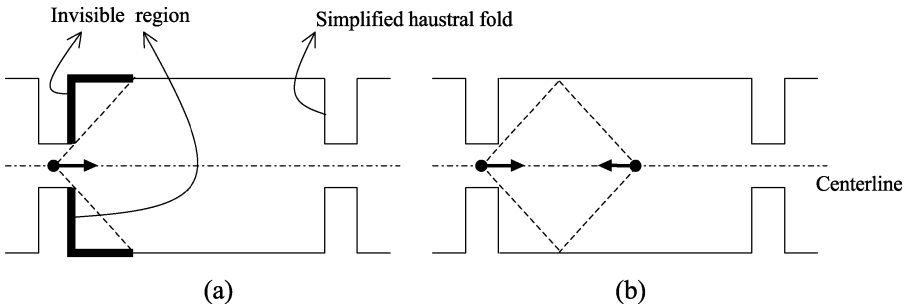

Fig. 1. Blind areas near haustral folds, depending on the navigation type. (a) Unidirectional navigation and (b) bidirectional navigation.

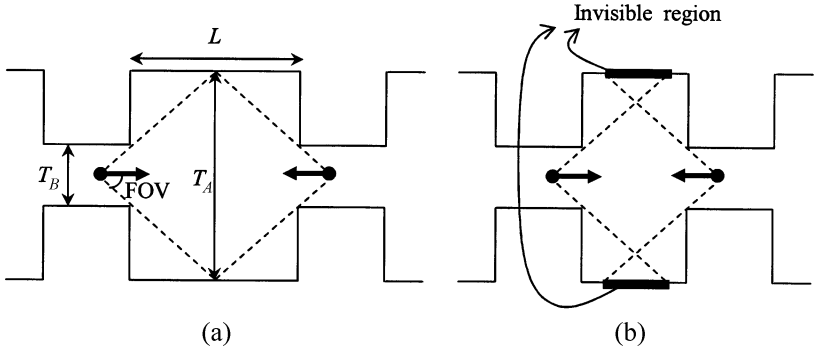

Fig. 2. Blind areas in bidirectional navigation, depending on the geometry of haustral folds. (a) $L = (T_A - T_B)/\tan(\mathrm{FOV})$ and (b) $L < (T_A - T_B)/\tan(\mathrm{FOV})$.

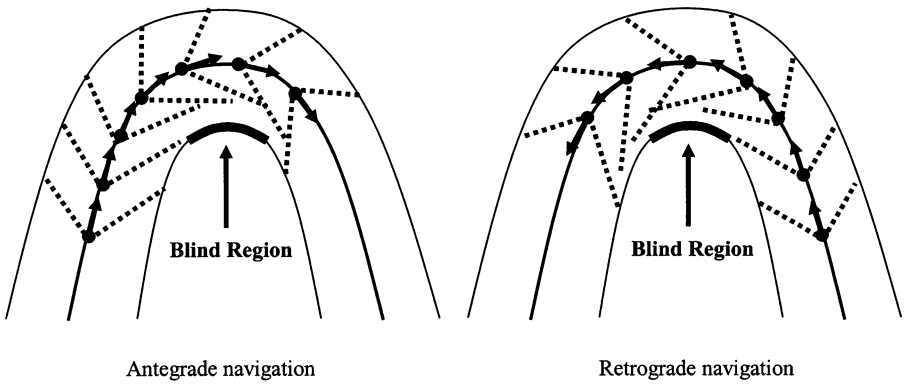

Fig. 3. Blind areas produced by an existing centerline path with a camera FOV of 45°.

the fold-to-fold distance, L, is not long enough [10]. More specifically, as illustrated in Fig. 2, the geometric condition for eliminating the blind areas is $L > (T_A - T_B)/\tan(\mathrm{FOV})$. To reduce this type of blind area, however, the traditional navigation method is unsuitable because a camera should pause and look up and down to observe the entire region between two folds. For a colon with highly curved parts, a portion of the highly curved region is often unobservable, even with bidirectional navigation along a conventional centerline path, because of the limited field of view (FOV) (see Fig. 3). Another method that attempts to generate a path that ensures complete observation is problematic: firstly, it is based on the impractical assumption that the camera has an unlimited FOV; secondly, it requires heavy computations [15]. Moreover, in spite of its significance, it fails to provide any novel method for determining the direction vectors.

We now propose a path planning algorithm that maximizes visibility in the flythrough. The algorithm is practically applicable and computationally simple. By assuming that the practical blind area between the folds is clinically negligible, which is true for the combined antegrade (starting from the rectum side) and retrograde navigation (starting from the caecum side) [24], [25], we concentrate on the visibility maximization, especially in the highly curved parts, for a camera of a limited FOV.

The rest of this paper is organized as follows. In Section II, we suggest a new measure for visibility coverage. In Section III, we describe the proposed path planning algorithm in detail. In Section IV, by using the measure for visibility coverage, we provide experimental results that show our algorithm to be more effective and clinically useful than a conventional algorithm. We offer our conclusions in Section V.

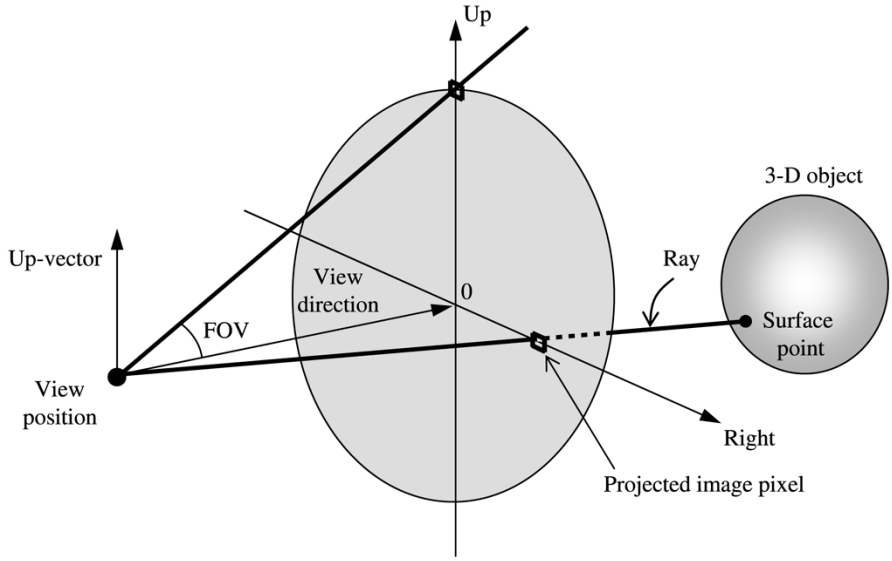

Fig. 4. Camera parameters for defining visibility.

II. New Visibility Measure

A quantitative measurement of the observable area is necessary for the evaluation of a determined path. For the measurement in the previous study [7], the "displayed regions" are defined as those regions that are displayed at least once on a screen during the flythrough. However, even though a region is displayed, an observer may not be able to interpret it if its display time is insufficient. For example, if the rendered images are displayed more than 10 fps, the interval between two successive images becomes less than 10 ms. Then, a region appearing only in a single frame will be displayed for less than 10 ms, which is too short to enable interpretation of a region of interest. Hence, for an observer to examine the region and recognize the surface shape around it, the region should be continuously displayed for an appropriate time. Based on this consideration, we define the visibility coverage for a camera path as follows.

1) Segment a volume of the colon lumen region, and extract its surface points.
2) Detect *spatially visible* surface points. A surface point is considered *spatially visible* if at least one ray reaches the surface point, and a ray starts from the camera view position and resides inside a predefined camera FOV as shown in Fig. 4.
3) Count *temporally visible* or *observable* surface points. A surface point is considered *temporally visible* if it is *spatially visible* during more than the definite time interval $T = N_S \cdot FPS$, where N_S represents the number of successively displayed images and FPS represents the frame rate for display. Note that T is intuitively selected by considering the moving speed of a camera. For example, a small T can be selected for a moving camera of high speed.
4) Calculate the visibility coverage as the ratio of the number of the *observable* surface points to the number of the total surface points on a colon segment.

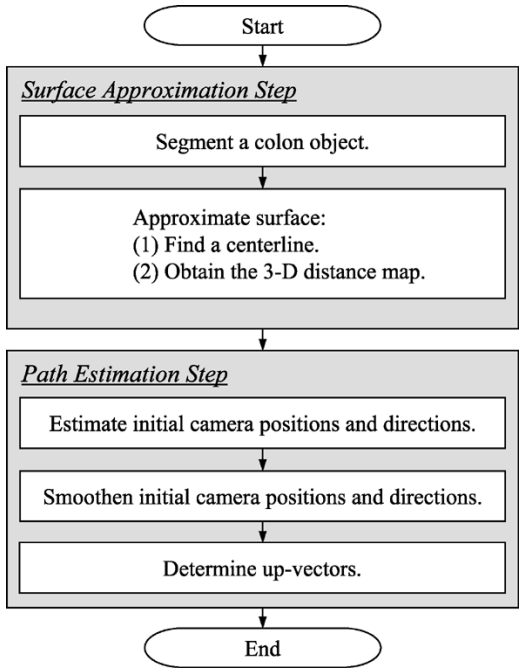

Fig. 5. Flowchart of the proposed algorithm.

III. Proposed Algorithm

To maximize visibility coverage, we propose a novel path planning algorithm. It consists of the two steps shown in Fig. 5. In the first step, we segment a colon lumen region from a CT volume by applying a seeded region-growing algorithm. Then, for the segment, we approximate the surface by determining the centerline and the 3-D distance map. In the second step, we simultaneously determine view positions and the corresponding view directions to maximally observe the approximated surface. We then smoothen the view positions for comfortable navigation. After obtaining all the view positions and view directions, we then determine the up-vectors.

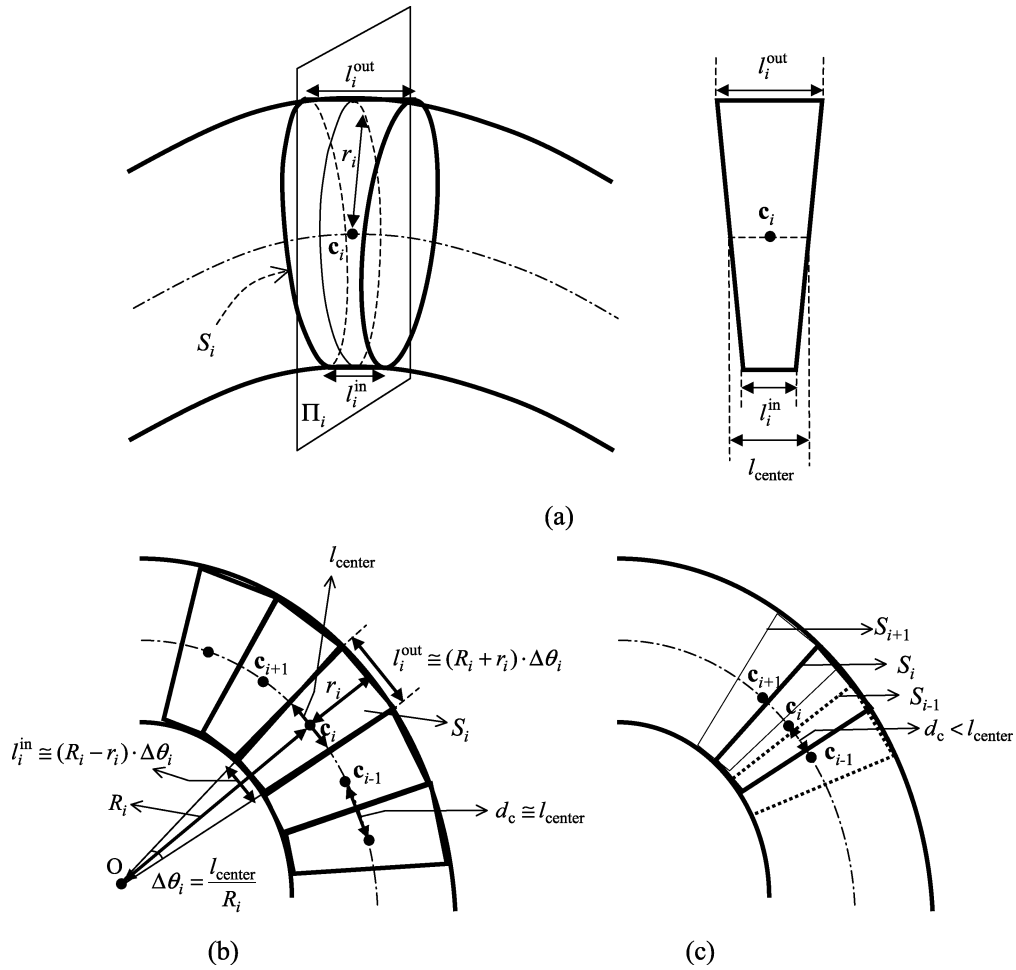

Fig. 6. (a) Cylinder-type surface segment S_i. l_{center} is set to a constant; l_i^{in} and l_i^{out} are determined depending on the center position $\mathbf{c}_i$. (b) Minimal set of consecutive segments S_i for surface approximation. The parameter r_i denotes the radius of the cylinder, and d_{c} denotes the distance between the two consecutive center positions $\mathbf{c}_i$ and $\mathbf{c}_{i+1}$. (c) Surface approximation using overlapping consecutive segments S_i.

A. Surface Approximation

In this step, we extract the surface information of a segmented object so that we can use the information in the next step of estimating the path. The surface is approximated by estimating the central axis and the corresponding cross-sectional thicknesses. The shape of a tubular object such as a human colon is well represented by the centerline of the object. Assume that $\mathbf{c}_i$, $1 \leq i \leq N$, is uniformly sampled with interval d_c on the centerline. We then define plane Π_i; it includes point $\mathbf{c}_i$ and its normal direction is the tangential vector of the centerline at $\mathbf{c}_i$. We then assume that the object cross-section cut by Π_i approximates a circle with a radius of r_i, which represents the thickness of the object at $\mathbf{c}_i$. As shown in Fig. 6(a), we then define, on the basis of the circle, a cylinder-type slice-surface S_i for $\mathbf{c}_i$. S_i has a symmetric shape for plane Π_i and can be defined by the inner, centric, and outer lengths, i.e., l_i^{in}, l_{center}, and l_i^{out}. Note here that l_{center} is set to a constant. And we determine l_i^{in} and l_i^{out} based on the extrinsic curvature κ_i at $\mathbf{c}_i$. In Fig. 6(b), if $\Delta\theta_i$ is defined as

$$\Delta\theta_i = \frac{l_{\text{center}}}{R_i} = \frac{l_{\text{center}}}{\frac{1}{|\kappa_i|}} \tag{1}$$

where R_i is the radius of curvature, we can obtain l_i^{in} and l_i^{out} as follows:

$$\begin{aligned} l_i^{\text{in}} &\cong (R_i - r_i)\Delta\theta_i \\ l_i^{\text{out}} &\cong (R_i + r_i)\Delta\theta_i. \end{aligned} \tag{2}$$

Then, the condition to approximate the total surface points with a minimal set of slices is

$$l_{\text{center}} = d_{\text{c}}. \tag{3}$$

By using (1) and (3), we can rewrite (2) as

$$\begin{aligned} l_i^{\text{in}} &\cong d_{\text{c}}\,(1 - r_i|\kappa_i|) \\ l_i^{\text{out}} &\cong d_{\text{c}}\,(1 + r_i|\kappa_i|)\,. \end{aligned} \tag{4}$$

It is interesting to note that $l_i^{\text{in}} = l_i^{\text{out}} = d_{\text{c}}$ for straight segments, since κ_i is zero in (4). If d_c becomes smaller than l_{center}, neighboring slices start to overlap as shown in Fig. 6(c). Note here that l_i^{in} and l_i^{out} need not to be computed for path planning because they are irrelevant to the visibility. Meanwhile, the set comprising the centerline position $\mathbf{c}_i$ and its corresponding thickness r_i, is needed for planning the path.

To obtain the centerline, we adopt an existing algorithm that is accurate, robust and not too complex [18]. By using the cross-section cut by Π_i, we may estimate the thickness r_i. But the result is sensitive to small changes in the centerline. Hence, for a robust estimation, we estimate r_i as a 3-D distance-transform value, T_i, at position $\mathbf{c}_i$ (see Fig. 7). In the distance-transform procedure, we assign to each object point a Euclidean distance value between the object point and the nearest nonobject point. For the 3-D distance transform, we use the 3-4-5 chamfer-based method, which is an approximated version of the Euclidean distance transform that reduces the computational complexity [26]. Because the distance-transform value continually varies, the thickness values that we estimate with aid of the distance transform also change continually. Hence, in some places where the thickness is abruptly changed, the estimated thickness does not accurately represent the real thickness (see Fig. 8). In the visualization, however, a smooth variation of the estimated thickness is preferred because it eventually improves the comfortableness of the navigation (will be discussed later.).

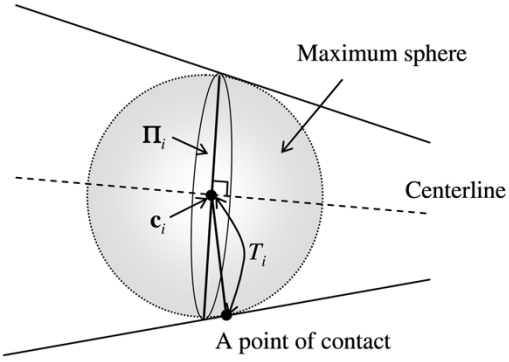

Fig. 7. Estimation of the thickness at $\mathbf{c}_i$ by the Euclidean distance-transform value T_i.

B. Initial Estimation of View Positions and View Directions

As aforementioned, the total surface of an object can be approximated by a set of slice surfaces. Hence, if we choose each camera position and its corresponding view direction so that the camera can view the corresponding slice surface, most of surface points along the path are *spatially visible*. Furthermore, when we sequentially view consecutive overlapping slice-surfaces as the camera moves, the points on the surfaces may be *temporally visible* because the slice surfaces overlap.

For convenience, we assume that the camera position, $\mathbf{p}(t)$, the view direction, $\mathbf{d}(t)$, the centerline, $\mathbf{c}(t)$, the thickness value at the centerline, $T(t)$, and the slice surface, $S(t)$, are continuous functions of time t. We then choose $\mathbf{p}(t)$ and $\mathbf{d}(t)$ as follows:

$$\mathbf{p}(t) = \mathbf{c}(t) - \mathbf{d}(t) \tag{5}$$

where

$$\mathbf{d}(t) = kT(t)\frac{\frac{\partial \mathbf{c}(t)}{\partial t}}{\left|\frac{\partial \mathbf{c}(t)}{\partial t}\right|} \tag{6}$$

and k is a constant value. The magnitude of $\mathbf{d}(t)$, or $d(t)$, equals $k \cdot T(t)$ and denotes the distance from the camera position, $\mathbf{p}(t)$, to the center of the slice surface, $\mathbf{c}(t)$. Fig. 9 graphically illustrates (5) and (6) for the case that $k = k_0 = 1/tan(\mathrm{FOV})$. It is interesting to note that as k approaches 0, (5) and (6) provide the same path as a conventional method. In Fig. 10, we compare our proposed path obtained by (5) and (6) with a conventional path, for a camera FOV of 45°.

In the strict sense, if k equals k_0, the algorithm ensures the visibility of only the farther half of each surface slice. In practice, however, we set k to a value larger than k_0 so that the other half nearer to the camera may become visible. Also, its visibility can be ensured by navigation in the reverse direction. In addition, we should note that the colon surface has irregular structures whose cross-section is not a circle with a radius obtained from the distance transform. The circle obtained is always smaller than the actual cross-section and is located inside it. Hence, k, which is larger than k_0, can also help reduce invisible areas.

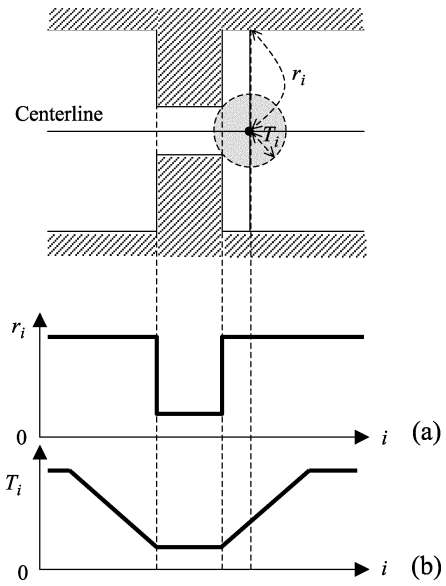

Fig. 8. (a) Radius values and (b) distance-transform values at the centerline, near the abrupt thickness change. Note that the distance-transform values at the centerline cannot accurately represent the real radii but roughly approximate them. Meanwhile, the distance-transform values guarantee the continuity.

Note that if $k < 1$, position $\mathbf{p}(t)$ is always inside the object because the distance from $\mathbf{p}(t)$ to $\mathbf{c}(t)$ becomes equal or smaller than $T(t)$, which is equivalent to the maximum radius of the possible inner spheres centered at $\mathbf{c}(t)$. However, if $k > 1$, position $\mathbf{p}(t)$ resides outside the maximum sphere, and two problems can arise as shown in Fig. 11. First, the estimated $\mathbf{p}(t)$ may be located outside the object as in Fig. 11(a). Second, a portion of the points on $S(t)$ are probably obstructed by a part of the colon surface as in Fig. 11(b). To avoid the first problem, we restrict $d(t)$ not to exceed the upper bound, $d_{\mathrm{AU}}(t)$, of $d_{\mathrm{A}}(t)$ that enables $\mathbf{p}(t)$ to be inside the object. To avoid the second problem, we prevent $d(t)$ from exceeding the upper bound, $d_{\mathrm{BU}}(t)$, of $d_{\mathrm{B}}(t)$ that enables all the points in $S(t)$ to be reached by a ray from $\mathbf{p}(t)$. We satisfy these two restrictions by adjusting $d(t)$ to the minimum value among $d_{\mathrm{AU}}(t)$, $d_{\mathrm{BU}}(t)$, and $d(t)$ itself. In other words, the upper-bound of $d(t)$ is $d_{\mathrm{U}} = \min\{d_{\mathrm{AU}}(t), d_{\mathrm{BU}}(t)\}$.

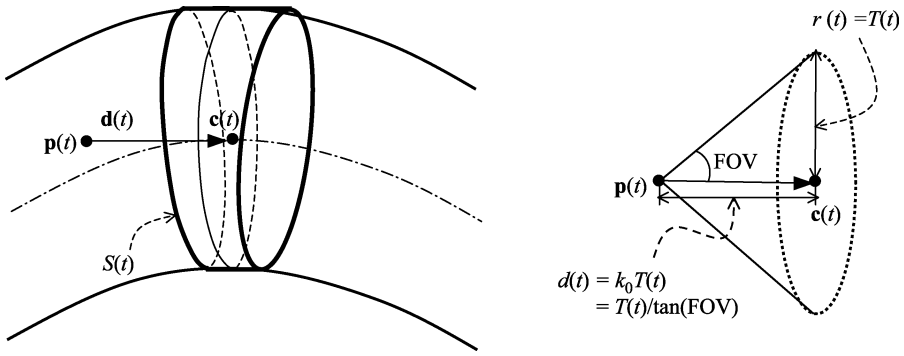

Fig. 9. Estimation of a view position $\mathbf{p}(t)$ and view direction $\mathbf{d}(t)$. The parameters $\mathbf{p}(t)$ and $\mathbf{d}(t)$ are estimated with the minimum k of 1/tan(FOV).

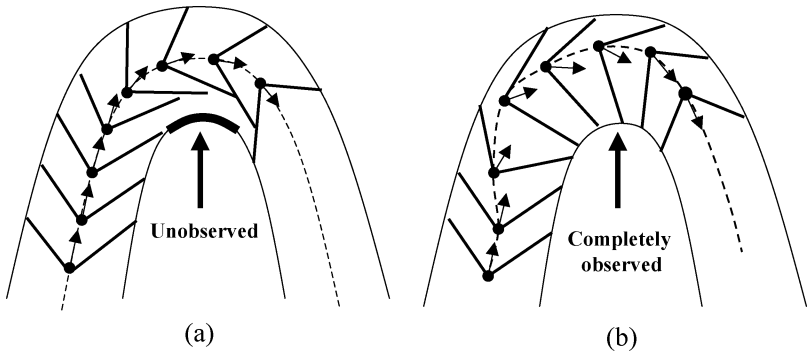

Fig. 10. Comparison of coverage efficiency for a curved object with a nearly uniform thickness. (a) Conventional path and (b) the proposed path.

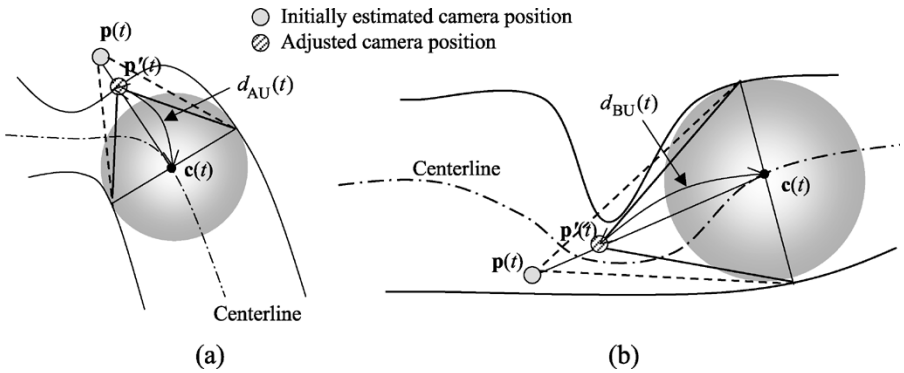

Fig. 11. Two examples of improper camera positions when $k > 1$. The adjusted camera position can be determined with (a) $d_{\mathrm{AU}}(t)$ and (b) $d_{\mathrm{BU}}(t)$, respectively.

The adjusted $d(t)$ may have discontinuity and it may cause the discontinuity of $\mathbf{p}(t)$ and $\mathbf{d}(t)$. Consequently, it makes the generated video uncomfortable to watch. For the comfortable watch, we must ensure that $\mathbf{p}(t)$ and $\mathbf{d}(t)$ are continuous and sufficiently smooth. In addition, by shortening $d(t)$, we can still produce an unwanted blind area with a limited FOV. However, because we use bidirectional navigation, we can hopefully reduce this area by observing it in the opposite direction.

C. Smoothening

From (5) and (6), $\mathbf{p}(t)$ and $\mathbf{d}(t)$ are represented by the functions $\mathbf{c}(t)$, $\partial\mathbf{c}(t)/\partial t$, and $d(t)$. To obtain a smooth path, we therefore need to sufficiently smoothen $\mathbf{c}(t)$ and $d(t)$. Hence, we first smoothen $\mathbf{c}(t)$ by filtering out the high-frequency components that cause the camera to joggle. The minimum-cost method [18] adopted for centerline extraction provides a good centerline with high centeredness. But due to its voxel-based centerline definition and roughness of the segmented surface, the centerline inevitably contains undesired small alterations which correspond to high frequency components in the spectrum of centerline positions. Thus, the objective of the smoothening process is to remove high frequency components in the centerline but to preserve the local curvedness. By using a low-pass-filter with an appropriate cutoff-frequency, unwanted high frequency components can be suppressed while preventing a big shift of the original centerline. The case of $\mathbf{c}_i$ being pushed outside the colon by the smoothing process rarely happens in practice. To handle this problem, however, we use a finite-impulse-response low-pass-filter so that we may adaptively smoothen the centerline with a different kernel according to a sampled position on the centerline. In other words, when the smoothened centerline is pushed outside the colon, we detect the over-smoothened part and reapply the filter with a higher cutoff-frequency to the corresponding original

centerline so that the whole centerline may be located inside the colon lumen.

The smoothening of $d(t)$ might not be accomplished in the same manner as in $\mathbf{c}(t)$, because the smoothened result of $d(t)$ should not exceed the upper-bound, $d_{\mathrm{U}}(t)$, for all t. Thus, we define the smoothening problem of $d(t)$ as the process of finding the band-limited function, $d_{\mathrm{BL}}(t)$, that is most similar to the original $d(t)$ but does not exceed the $d_{\mathrm{U}}(t)$ for all t. The function $d_{\mathrm{BL}}(t)$ is band-limited with a cutoff frequency, f_c. Because we can use low-pass filtering to get the band-limited function that is most similar to the arbitrary function, our $d(t)$-smoothening procedure also involves low-pass filtering.

The low-pass filtering has several steps. First, we low-pass filter the original $d(t)$ or $d_0(t)$ with a cutoff frequency of f_{c} to obtain $d_1(t)$, where $d_1(t) = LPF[d_0(t)]$. Here, $LPF[\cdot]$ denotes a low-pass-filtering operation. If $d_1(t)$ does not exceed the upper-bound, $d_{\mathrm{U}}(t)$, for all t, it is set to the final smoothened result. Otherwise, to satisfy the upper-bound constraint, we clip $d_1(t)$ as follows: $d_1^{\mathrm{clip}}(t) = \min\{d_1(t), d_{\mathrm{U}}(t)\}$. Because $d_1^{\mathrm{clip}}(t)$ is not a smooth function, we again subject it to a low-pass filter with the same f_{c} to obtain $d_2(t)$. Hence, in the nth repetition, we can express this procedure as follows:

$$d_n(t) = LPF\left[d_{n-1}^{\mathrm{clip}}(t)\right]. \tag{7}$$

We repeated the procedure until the smoothened path does not exceed the upper bound; that is, $e_n(t)$ converges to zero, where

$$e_n(t) = d_n(t) - d_n^{\mathrm{clip}}(t). \tag{8}$$

We then set either $d_n(t)$ or $d_n^{\mathrm{clip}}(t)$ to the final result. From (7) and (8), we can write the $d(t)$-smoothening procedure as follows:

$$d_n^{\mathrm{clip}}(t) = LPF^{(n)}[d_0(t)] - LPF^{(n-1)}[e_1(t)] - LPF^{(n-2)}[e_2(t)] \cdots - e_n(t), \tag{9}$$

where $LPF^{(m)}[x(t)]$ is obtained by repeatedly applying the low-pass filter to function $x(t)$ m times. Theoretically, if we repeatedly apply ideal low-pass filtering with a cutoff frequency of f_{c} to the arbitrary function $x(t)$, the filtered functions $LPF^{(m)}[x(t)]$, $m = 1, 2 \ldots$ should be the same. However, the practical gains of the filter at frequencies lower than f_{c} could be smaller than unity, and, from (9), they could make the final $d(t)$ excessively smoothened and unnecessarily distorted from the original $d_0(t)$. To alleviate this excessive attenuation, we should avoid the repeated filtering. Hence, we approximate (7) as follows:

$$\begin{aligned} d_n(t) &= LPF\left[d_{n-1}^{\mathrm{clip}}(t)\right] \\ &= LPF\left[d_{n-1}(t) - e_{n-1}(t)\right] \\ &\approx d_{n-1}(t) - LPF\left[e_{n-1}(t)\right]. \end{aligned} \tag{10}$$

The function $d_n^{\mathrm{clip}}(t)$ can then be represented as

$$d_n^{\mathrm{clip}}(t) = LPF\left[d_0(t)\right] - LPF\left[e_1(t)\right] - LPF\left[e_2(t)\right] \cdots - e_n(t). \tag{11}$$

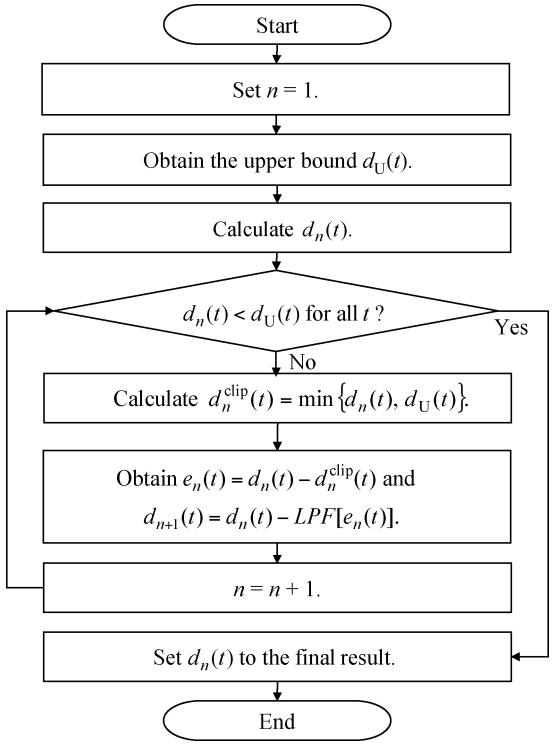

Fig. 12. Flowchart of the algorithm for smoothening $d(t)$.

Contrary to the situation in (9), we can obtain the $d_n^{\mathrm{clip}}(t)$ in (11) without repeated filtering. Fig. 12 shows the smoothening procedure. The low-pass filter in the procedure has a cutoff frequency that can control the degree of smoothness. Fig. 13 gives two examples of smoothening. For the synthesized function $d(t)$ without an upper-bound constraint, Fig. 13(b) shows the smoothening results with the user-defined cutoff frequency depicted in Fig. 13(a). Fig. 13(c) shows the corresponding results for the synthesized function $d(t)$ with an upper-bound constraint. Fig. 13(d) shows the convergence of the algorithm. To guarantee the convergence of $e_n(t)$, we may adopt a finite-impulse-response low-pass-filter having nonnegative coefficients, whose performance is still good enough for smoothening $d(t)$. Then, the filtering can be regarded as averaging of neighboring samples with positive weights. Hence, $LPF[e_n(t)] > 0$ for any n before converging, because $e_n(t) > 0$. Thus, from (10), $d_n(t)$ always decreases as n increases. From (8), $e_n(t) - e_{n-1}(t)$ becomes negative since $d_n^{\mathrm{clip}}(t) = d_{n-1}^{\mathrm{clip}}(t) = d_U(t)$ for a positive $e_n(t)$. Thus $e_n(t)$ monotonically decreases and becomes zero if $d_n(t)$ is less than $d_{\mathrm{U}}(t)$.

D. Determination of Up-Vectors

Fig. 4 demonstrates a rendering process for a virtual camera with a limited FOV. As shown in the figure, the selection of up-vector $\mathbf{u}_i$ is unrelated to visibility. This up-vector only affects the comfortableness of navigation because it determines the degree of rotation between rendered images. Hence, we choose $\mathbf{u}_i$ to minimize the rotation between consecutive images. To minimize the rotation, $\hat{\mathbf{u}}_i$ minimizing $\|\hat{\mathbf{u}}_i - \hat{\mathbf{u}}_{i-1}\|^2$ is to be determined under the condition of $\mathbf{u}_i \perp \mathbf{d}_i$. Here, $\|\cdot\|$

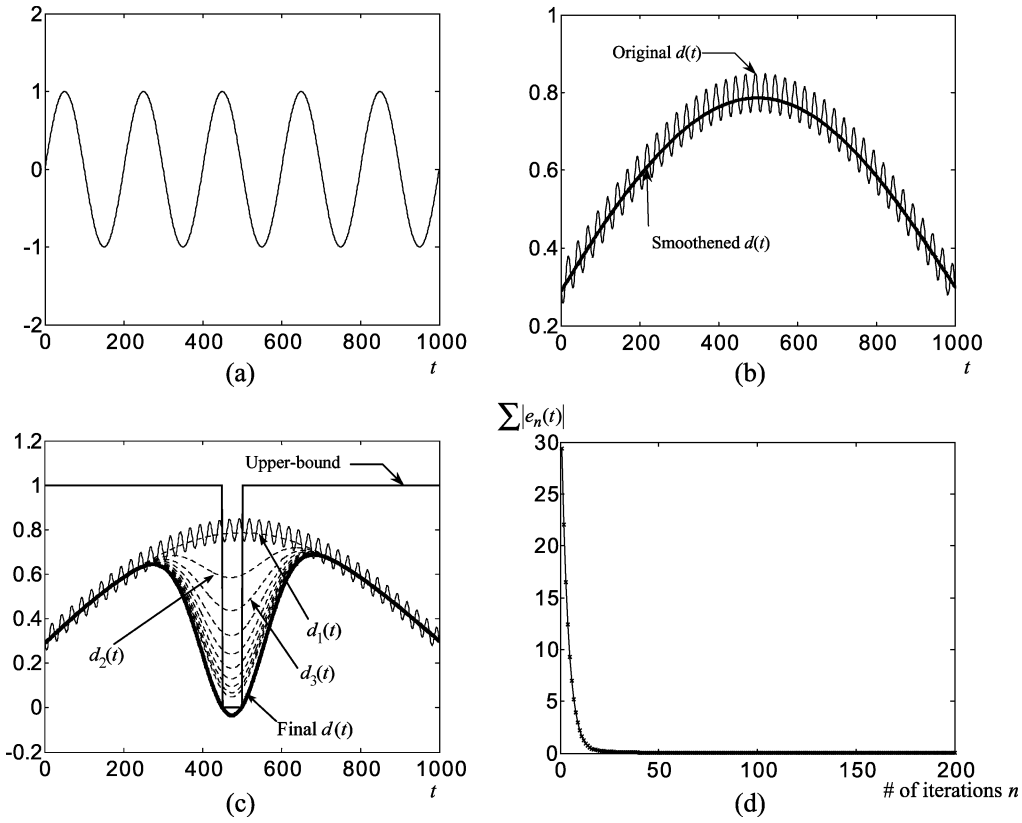

Fig. 13. Examples of $d(t)$-smoothening. (a) A sinusoidal function with the same frequency as the cutoff frequency of the adopted low-pass filter. (b) The smoothened result of the synthesized function $d(t)$ *without* an upper-bound constraint. (c) The smoothened result *with* an upper-bound constraint, and (d) the corresponding convergence curve of $\sum |e_n(t)|$.

denotes the 2-norm operation and $\hat{\mathbf{x}}$ denotes the unit vector of $\mathbf{x}$. Meanwhile, $||\hat{\mathbf{u}}_i - \hat{\mathbf{u}}_{i-1}||^2$ can be rewritten as

$$\begin{aligned} ||\hat{\mathbf{u}}_i - \hat{\mathbf{u}}_{i-1}||^2 &= ||\hat{\mathbf{u}}_i||^2 - 2\hat{\mathbf{u}}_{i-1} \cdot \hat{\mathbf{u}}_i + ||\hat{\mathbf{u}}_{i-1}||^2 \\ &= 2 - 2 \cdot \hat{\mathbf{u}}_{i-1} \cdot \hat{\mathbf{u}}_i \\ &= 2 - 2 \cdot [(\hat{\mathbf{u}}_{i-1} \cdot \mathbf{d}_i)\mathbf{d}_i + \hat{\mathbf{u}}_{i-1} \\ &\qquad - (\hat{\mathbf{u}}_{i-1} \cdot \mathbf{d}_i)\mathbf{d}_i] \cdot \hat{\mathbf{u}}_i \\ &= 2 - 2 \cdot [\hat{\mathbf{u}}_{i-1} - (\hat{\mathbf{u}}_{i-1} \cdot \mathbf{d}_i)\mathbf{d}_i] \cdot \hat{\mathbf{u}}_i. \quad (12) \end{aligned}$$

Hence, $\hat{\mathbf{u}}_i$ minimizing $||\hat{\mathbf{u}}_i - \hat{\mathbf{u}}_{i-1}||^2$ is $(\hat{\mathbf{u}}_{i-1} - (\hat{\mathbf{u}}_{i-1} \cdot \mathbf{d}_i)\mathbf{d}_i)/(||\hat{\mathbf{u}}_{i-1} - (\hat{\mathbf{u}}_{i-1} \cdot \mathbf{d}_i)\mathbf{d}_i||)$ and it also satisfies $\hat{\mathbf{u}}_i \perp \mathbf{d}_i$. It is interesting to note that $\mathbf{u}_i$ can be obtained by projecting $\mathbf{u}_{i-1}$ onto the plane perpendicular to $\mathbf{d}_i$.

IV. Experiments

A. Conditions

To demonstrate the performance of our algorithm, we compare it to a conventional algorithm in which the view positions are determined from the centerline and the view directions are set to the tangential direction of the centerline [18]. To achieve better coverage at a region near a haustral fold, both algorithms use a combined antegrade navigation and retrograde navigation [24], [25]. To fairly compare the coverage, we should extract the same number of view positions (or images) in each algorithm so that the time for the flythrough would be the same.

As mentioned in Section III-A, the interval of the sampled point on the centerline determines the degree of overlap among the slice surfaces. And the degree of overlap directly affects two important parameters: the observing time for shape interpretation and the total navigation time. Hence, we determine the degree of overlap or the navigation speed after deciding the values of those two parameters properly. For the combined navigation of a human colon with a length of 1.5 m and an interval between neighboring center positions of about 1 mm, the total time required is approximately 300 s. We obtain this time by assuming an appropriate frame rate for display of about 10 fps. Meanwhile, in the proposed algorithm, we select the main parameter k so that it would be greater than or equal to $k_0 (= 1/\tan(\text{FOV}))$: for an FOV of 60°, we set k to 1.5 for $k_0 \approx 0.58$; and, for an FOV of 45°, we set k to 2.0 for $k_0 = 1$.

TABLE I
Dimension and Resolution of Human Colon CT Data Sets for Experiments

Dataset	Resolution (mm^3)	Dimension
A	0.676 x 0.676 x 1.5	512 x 512 x 301
B	0.582 x 0.582 x 1.5	512 x 512 x 285
C	0.578 x 0.578 x 1.5	512 x 512 x 315

B. Results

By using the three human colon CT data sets described in Table I, we compare the visibility coverage with the data in Table II for the two different FOV's and two N_S values. According to the results in Table II, if we assume for satisfactory

TABLE II
COMPARISON OF VISIBILITY COVERAGE ($N_S = 1$ CORRESPONDS TO THE CONVENTIONAL VISIBILITY MEASURE)

FOV	Direction of flythrough	Method	Coverage rate (%)					
			$N_S = 1$ ($T = 0.1$ sec)			$N_S = 10$ ($T = 1$ sec)		
			Dataset A	Dataset B	Dataset C	Dataset A	Dataset B	Dataset C
45°	Antegrade	Conventional	63.54	65.99	70.38	55.41	57.25	64.58
		Proposed	77.58	79.15	88.52	67.52	68.27	81.49
		Difference	**14.04**	**13.16**	**18.14**	**12.11**	**11.02**	**16.91**
	Retrograde	Conventional	62.81	66.36	72.85	55.13	56.96	66.77
		Proposed	77.08	77.92	89.90	67.50	67.40	82.89
		Difference	**14.27**	**11.56**	**17.05**	**12.37**	**10.44**	**16.12**
	Combined	Conventional	84.11	86.21	91.83	78.22	78.94	88.12
		Proposed	94.50	94.50	98.61	88.44	87.16	95.48
		Difference	**10.39**	**8.29**	**6.78**	**10.22**	**8.22**	**7.36**
60°	Antegrade	Conventional	79.64	80.92	84.64	70.04	70.29	79.09
		Proposed	86.94	88.20	93.76	76.84	76.89	88.73
		Difference	**7.30**	**7.28**	**9.12**	**6.80**	**6.60**	**9.64**
	Retrograde	Conventional	78.63	79.85	86.05	69.43	69.04	80.62
		Proposed	86.55	86.65	95.56	76.40	75.16	90.51
		Difference	**7.92**	**6.80**	**9.51**	**6.97**	**6.12**	**9.89**
	Combined	Conventional	94.74	95.03	98.07	90.70	89.46	96.25
		Proposed	99.40	99.08	99.95	96.31	94.82	99.50
		Difference	**4.66**	**4.05**	**1.88**	**5.61**	**5.36**	**3.25**

observation that a surface point should be displayed for at least 1 s (that is, $N_S = 10$ with a frame rate of 10 fps), the proposed algorithm enables us to observe between 96 percent to 99 percent of the surface points for a camera FOV of 60°. These results are between 3 percent to 6 percent better than the results of a conventional algorithm. In Table II, the coverage rate is more degraded for a smaller FOV. The first reason for the degradation is that a small FOV increases the invisible area generated between the haustral folds, as illustrated in Fig. 2. The second reason is that, the value of k_0 becomes larger for a smaller FOV. As a result, the original $d(t)$ violates the upper-bound, and more distortions are consequently produced in the smoothening process of $d(t)$. However, the graphs in Fig. 14 show that our proposed path always outperforms the conventional centerline path for various N_S values.

In Fig. 15, the blind areas are displayed to show the superiority of our proposed algorithm in a different way. The results in the figure show that our proposed path improves the coverage rate by reducing the blind areas, especially at highly curved parts. However, as we predicted in Section I, some areas between the folds are still not visible. Hence, we can separately display the remaining blind area after the flythrough for complete examination [27]. But not all blind areas are equally important. Because the range of diameters of clinically important polyps is limited to between 5 mm and 30 mm in virtual colonoscopy, we do not consider the blind areas that have a diameter of less than 5 mm. The graphs in Fig. 16 compare the distribution of

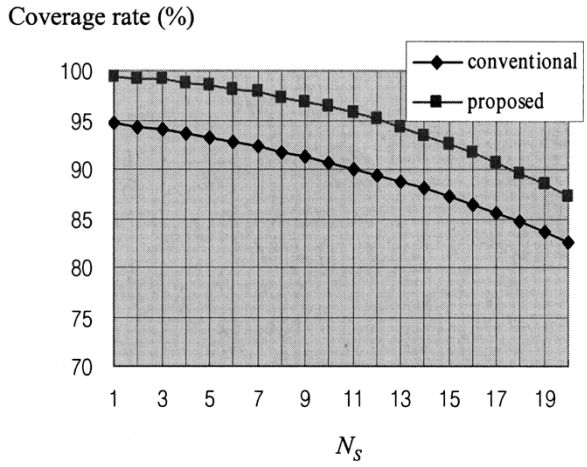

Fig. 14. Visibility coverage corresponding to various N_S values with combined navigation. Data set A is used, and the FOV of 60° is assumed.

the sizes of the blind patches. In the figure, a patch denotes a group of connected blind points on the surface; the size of the patch is defined as a diameter that is twice the number of layers peeled off in the thinning process. The graphs clearly show that our proposed algorithm not only reduces the number of blind patches, but also concentrates the diameter of the blind polyps to below 5 mm. Thus, the visibility increase due to our algorithm further improves the ability of clinical diagnosis.

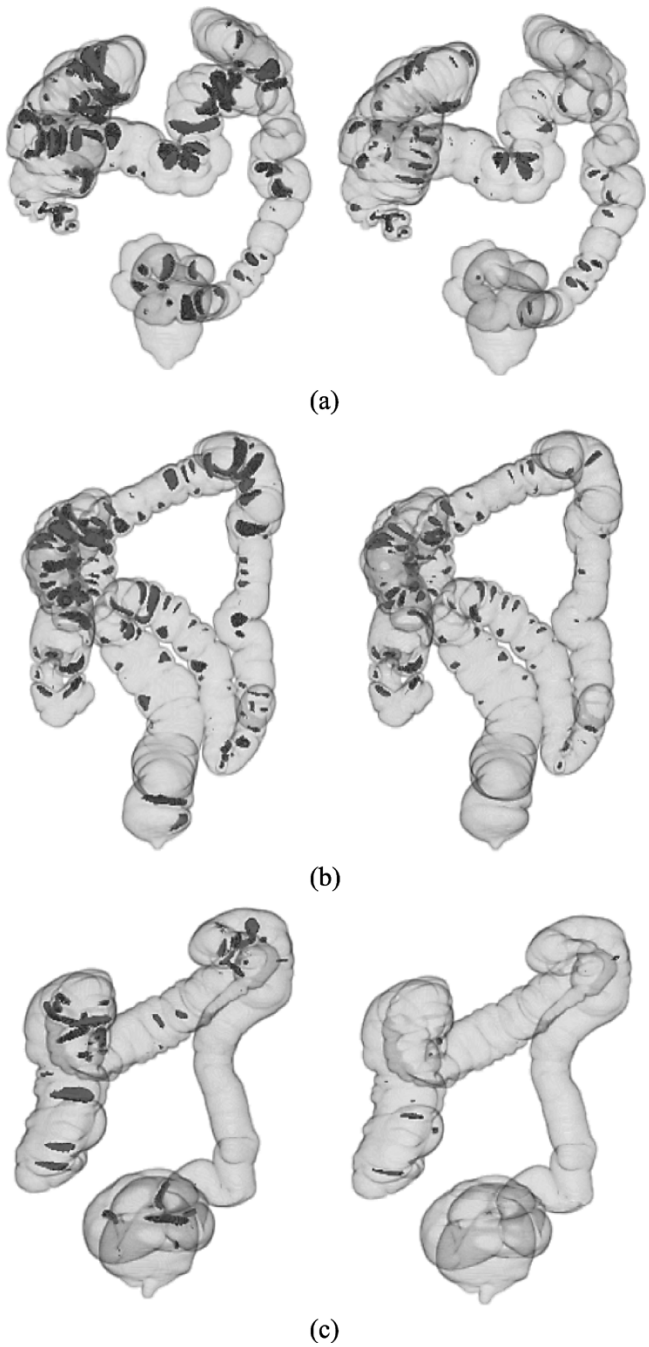

Fig. 15. Blind areas obtained with an N_S value of 10 and an FOV of 60°, by using (a) data set A, (b) data set B, and (c) data set C. Images in the left column are obtained with a conventional path and the images in the right column are obtained with the proposed path.

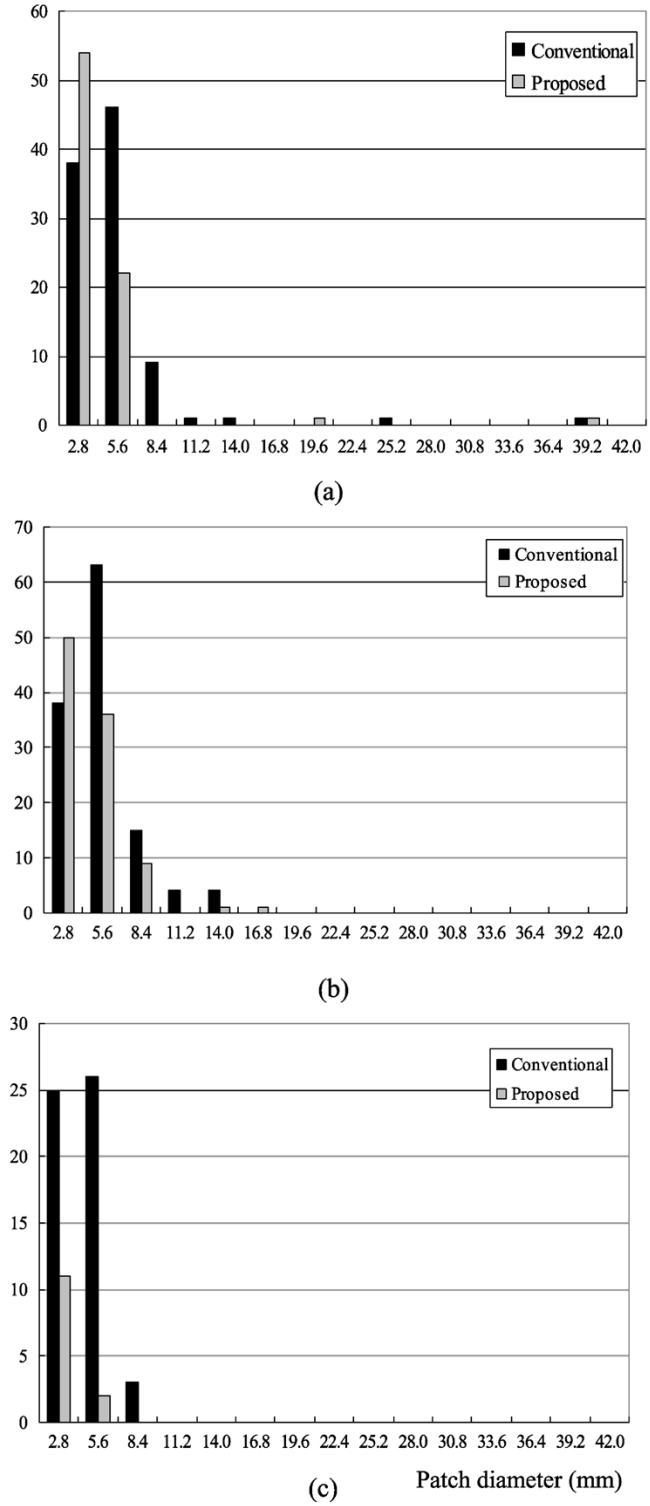

Fig. 16. Distribution of the sizes of the blind patches for (a) data set A, (b) data set B, and (c) data set C.

Fig. 17 shows the volume-rendered images and obtained paths for a colon data set, as well as some typical frames that are generated along the obtained path. In contrast to a conventional path, the proposed path clearly provides a more stable and centered view in the rendered images. In addition, Fig. 17 shows that the proposed camera path is generally more winding and longer than a conventional centerline path. This phenomenon occurs because (6) includes a derivative term of the centerline. Hence, if we extract the same number of positions on each path, the position-to-position interval of the proposed path may be longer than that of a conventional path.

Finally, Table III shows the execution time for both algorithms. The execution time includes the computation time for distance map generation, centerline extraction, and path planning and smoothing. We obtain the data by implementing the algorithm on a PC equipped with a 2-GHz Pentium CPU. As shown in the table, our proposed algorithm takes about 20 s and, compared to a conventional algorithm, its additional computational burden is insignificant.

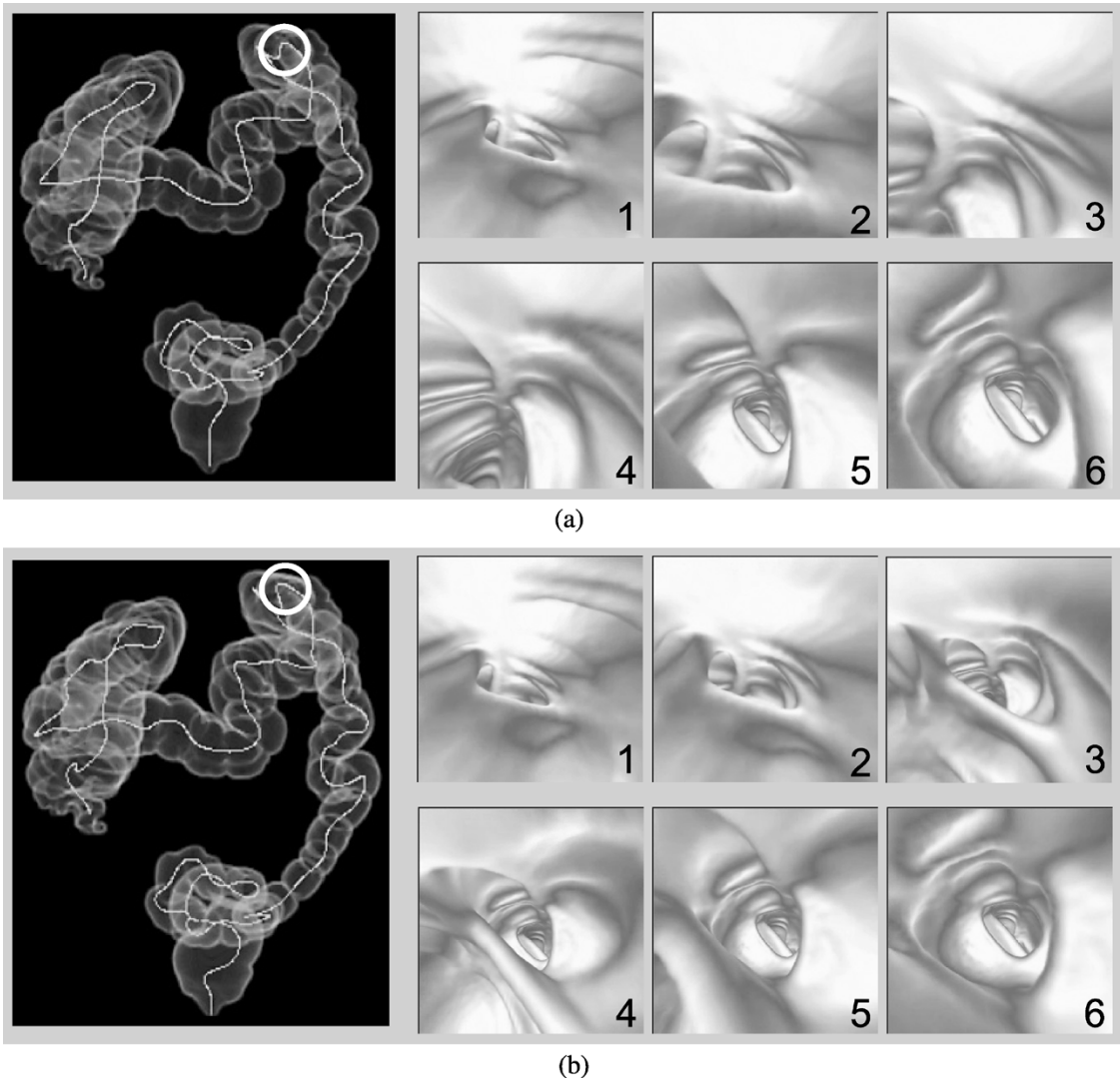

Fig. 17. Generated paths (the camera positions are illustrated) and the corresponding rendered images at a region of high curvature by using (a) a conventional algorithm and (b) the proposed algorithm.

V. Conclusion

We propose a path planning algorithm that minimizes the blind area and provides a comfortable flythrough in virtual colonoscopy by appropriately determining the camera positions, view directions, and camera up-vectors. The main contribution of our work is that it incorporates surface information into the path planning procedure. By relying on an extracted centerline and its distance-transform values, we roughly approximate the colon shape with consecutive overlapping segments. To maximize the temporally visible area from a camera with a limited FOV, the algorithm then finds the camera positions and directions appropriate for sequentially viewing the approximated segments. The determination of up-vectors is also important for comfortable navigation. To maximize comfortableness, we therefore choose up-vectors that minimize the camera rotation and smoothen the camera positions while keeping the path inside the colon object. For evaluation of our proposed algorithm, we quantify the blind area by using a new visibility coverage measure that reflects the temporal visibility for a camera path. The results of the simulation show that our proposed algorithm improves visibility at the highly curved regions of a human colon. Our algorithm therefore outperforms a conventional algorithm in terms of the new visibility coverage measure. Furthermore, the additional computational burden of our algorithm is insignificant. Moreover, our algorithm is clinically useful because it reduces the number and size of blind areas, thereby significantly reducing the possibility of overlooking polyps of meaningful size. Finding a way to minimize the blind area between haustral folds may be the next challenging topic in the virtual flythrough method.

TABLE III
Comparison of the Calculation Time Required for Path Generation

	Dataset A	Dataset B	Dataset C
Conventional	14 sec	13 sec	17 sec
Proposed	18 sec	17 sec	21 sec

Acknowledgment

The authors wish to thank Prof. W. J. Lee, Yonsei University College of Medicine, for providing them CT data.

REFERENCES

[1] M. J. O'Brien, S. J. Winawer, and A. G. Zauber *et al.*, "The national polyp study: patient and polyp characteristics associated with high-grade dysplasia in colorectal adenomas," *Gastoenterol.*, vol. 98, pp. 371–379, 1990.

[2] H. M. Fenlon and J. T. Ferrucci, "Virtual colonoscopy: What will the issues be?," *AJR Am. J. Roentgenol.*, vol. 169, no. 2, pp. 453–458, Aug. 1997.

[3] L. Hong, A. Kaufman, Y. Wei, A. Viswambharan, M. Wax, and Z. Liang, "3D virtual colonoscopy," presented at the *Symp. Biomedical Visualization*, Los Alamitos, CA, Oct. 1995, pp. 26–32.

[4] L. Hong, S. Muraki, A. Kaufman, D. Bartz, and T. He, "Virtual voyage: interactive navigation in the human colon," in *Proc. ACM SIGGRAPH*, Los Angeles, CA, Aug. 1997, pp. 27–34.

[5] T.-Y. Lee, P.-H. Lin, C.-H. Lin, Y.-N. Sun, and X.-Z. Lin, "Interactive 3-D virtual colonoscopy system," *IEEE Trans. Inform. Technol. Biomed.*, vol. 3, no. 2, pp. 139–150, Jun. 1999.

[6] M. Wan, W. Li, A. Kaufman, Z. Liang, D. Chen, and M. Wax, "3-D virtual colonoscopy with real-time volume rendering," in *Proc. SPIE Med. Imag.*, vol. 3978, San Diego, CA, Feb. 2000, pp. 165–170.

[7] Y. Hayashi, K. Mori, J. Hasegawa, Y. Suenaga, and J. Toriwaki, "A method for detecting undisplayed regions in virtual colonoscopy and its application to quantitative evaluation of fly-through methods," *Acad. Radiol.*, vol. 10, pp. 1380–1391, 2003.

[8] S. Haker, S. Angenent, A. Tannenbaum, and R. Kikinis, "Nondistorting flattening maps and the 3-D visualization of colon CT images," *IEEE Trans. Med. Imag.*, vol. 19, no. 7, pp. 665–670, Jul. 2000.

[9] A. V. Bartroli, R. Wegenkittl, A. König, and E. Gröller, "Nonlinear virtual colon unfolding," in *Proc. IEEE Visualization 2001*, San Diego, CA, Oct. 2001, pp. 411–418.

[10] F. M. Vos, I. W. O. Serlie, R. E. van Gelder, F. H. Post, R. Truyen, F. A. Gerritsen, J. Stoker, and A. M. Vossepoel, "A new visualization method for virtual colonoscopy," in *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2001 (Proc. 4th Int. Conf., Utrecht, NL, Oct. 14–17)*. ser. Lecture Notes in Computer Science, W. J. Niessen and M. A. Viergever, Eds. Berlin, Germany: Springer Verlag, Oct. 2001, vol. 2208, pp. 645–654.

[11] R. Chiou, A. Kaufman, Z. Liang, L. Hong, and M. Achniotou, "An interactive fly-path planning using potential fields and cell decomposition for virtual endoscopy," *IEEE Trans. Nucl. Sci.*, vol. 46, no. 4, pp. 1045–1049, Aug. 1999.

[12] D. Paik, C. Beaulieu, R. Jeffery, G. Rubin, and S. Napel, "Automated flight path planning for virtual endoscopy," *Med. Phys.*, vol. 25, no. 5, pp. 629–637, May 1998.

[13] Y. Zhou and A. W. Toga, "Efficient skeletonization of volumetric objects," *IEEE Trans. Visual. Comput. Graphics*, vol. 5, no. 3, pp. 196–209, Jul.–Sep. 1999.

[14] I. Bitter, A. E. Kaufman, and M. Sato, "Penalized-distance volumetric skeleton algorithm," *IEEE Trans. Vis. Comput. Graphics*, vol. 7, no. 3, pp. 195–206, Jul.–Sep. 2001.

[15] T. He, L. Hong, D. Chen, and Z. Liang, "Reliable path for virtual endoscopy: ensuring complete examination of human organs," *IEEE Trans. Vis. Comput. Graph.*, vol. 7, no. 4, pp. 333–342, Oct.–Dec. 2001.

[16] Y. Samara, M. Fiebrich, A. H. Dachman, J. K. Kuniyoshi, K. Doi, and K. R. Hoffman, "Automated calculation of the centerline of the human colon on CT images," *Acad. Radiol.*, vol. 6, no. 6, pp. 352–359, 1999.

[17] G. Iordanescu and R. M. Summers, "Automated centerline for computed tomography colonography," *Acad. Radiol.*, vol. 10, no. 11, pp. 1291–1301, 2003.

[18] M. Wan, Z. Liang, Q. Ke, L. Hong, I. Bitter, and A. Kaufman, "Automatic centerline extraction for virtual colonoscopy," *IEEE Trans. Med. Imag.*, vol. 21, no. 12, pp. 1450–1460, Dec. 2002.

[19] D.-G. Kang, S. M. Kwon, and J. B. Ra, "A virtual colonoscopy system based on automatic path generation and fast volume rendering," in *Proc. Virtual Systems and Multimedia 2002*, Gyeongju, Korea, Sep. 2002, pp. 878–887.

[20] I. Bitter, M. Sato, M. Bender, K. McDonnel, A. Kaufman, and M. Wan, "CEASAR: a smooth, accurate and robust centerline-extraction algorithm," in *Proc. Visualization 2000*, Salt Lake City, UT, 2000, pp. 45–52.

[21] Y. Ge, D. R. Stelts, J. Wang, and D. J. Vining, "Computing the centerline of a colon: a robust and efficient method based on 3-D skeleton," *J. Comput. Assist. Tomogr.*, vol. 23, no. 5, pp. 786–794, 1999.

[22] T. Deschamps and L. D. Cohen, "Fast extraction of minimal paths in 3D images and applications to virtual endoscopy," *Med. Image Anal.*, vol. 5, no. 4, pp. 281–299, 2001.

[23] Y. Zhou, A. Kaufman, and A. W. Toga, "Three-dimensional skeleton and centerline generation based on an approximate minimum distance field," *Vis. Comput.*, vol. 14, no. 7, pp. 303–314, 1998.

[24] H. M. Fenlon, D. P. Nunes, P. D. Clarke, and J. T. Ferrucci, "Colorectal neoplasm detection using virtual colonoscopy: A feasibility study," *Gut*, vol. 43, pp. 806–811, 1998.

[25] J. T. Ferrucci, "Colon cancer screening with virtual colonoscopy: promise, polyps, politics," *Am. J. Roentgenol.*, vol. 177, pp. 975–988, 2001.

[26] G. Borgefors, "Distance transformations in digital images," *Comput. Vis. Graph. Image Process.*, vol. 34, pp. 344–371, 1986.

[27] K. Kreeger, F. Dachille, M. Wax, and A. E. Kaufman, "Covering all clinically significant areas of the colon surface in virtual colonoscopy," in *Proc. SPIE Med. Imag.*, vol. 4683, San Diego, CA, Feb. 2002, pp. 198–206.