A new MLC segmentation algorithm/software for step-and-shoot IMRT delivery

Shuang Luan, Chao Wang, Danny Z. Chen, and Xiaobo S. Hu
Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, Indiana 46556
Shahid A. Naqvi and Cedric X. Yu
Department of Radiation Oncology, University of Maryland School of Medicine, Baltimore, Maryland 21201
Chad L. Lee
Department of Radiation Oncology, Sinai Hospital, Baltimore, Maryland 21215

(Received 23 July 2003; revised 17 December 2003; accepted for publication 17 December 2003; published 10 March 2004)

We present a new MLC segmentation algorithm/software for step-and-shoot IMRT delivery. Our aim in this work is to shorten the treatment time by minimizing the number of segments. Our new segmentation algorithm, called SLS (an abbreviation for static leaf sequencing), is based on graph algorithmic techniques in computer science. It takes advantage of the geometry of intensity maps. In our SLS approach, intensity maps are viewed as three-dimensional (3-D) “mountains” made of unit-sized “cubes.” Such a 3-D “mountain” is first partitioned into special-structured submountains using a new mixed partitioning scheme. Then the optimal leaf sequences for each submountain are computed by either a shortest-path algorithm or a maximum-flow algorithm based on graph models. The computations of SLS take only a few minutes. Our comparison studies of SLS with CORVUS (both the 4.0 and 5.0 versions) and with the Xia and Verhey segmentation methods on Elekta Linac systems showed substantial improvements. For instance, for a pancreatic case, SLS used only one-fifth of the number of segments required by CORVUS 4.0 to create the same intensity maps, and the SLS sequences took only 25 min to deliver on an Elekta SL 20 Linac system in contrast to the 72 min for the CORVUS 4.0 sequences (a three-fold improvement). To verify the accuracy of our new leaf sequences, we conducted film and ion-chamber measurements on phantom. The results showed that both the intensity distributions as well as dose distributions of the SLS delivery match well with those of CORVUS delivery. SLS can also be extended to other types of Linac systems.

Key words: conformal therapy, IMRT, leaf sequencing, MLC segmentation algorithm

I. INTRODUCTION

Intensity Modulated Radiation Therapy (IMRT) is a new technique for cancer treatments and aims to deliver a highly conformal dose to the target volume while keeping the dose to other surrounding organs at risk (OAR) below tolerance levels.1–6,10,15,17,25,26,30,34 The implementation of IMRT requires an ability to deliver two-dimensional nonuniform fluence distributions, called intensity maps or intensity modulated beams (IMBs) [Fig. 1(a)].

Currently, IMRT can be delivered by multiple overlapping fields shaped by a multileaf collimator (MLC) [see Fig. 1(b) for a schematic representation of a MLC]. Using a MLC, IMRT can be implemented either statically or dynamically. In dynamic approaches, the MLC leaves keep moving across a treatment field while the radiation remains on. In static approaches, also referred to as “step-and-shoot” IMRT, the MLC leaves stay stationary during irradiation, and move to reshape the beam while the radiation is turned off. A number of dynamic and static MLC techniques have been described in Webb’s books.26,27

Due to the complexity of dynamic intensity modulated beams, the verification of all aspects of the planning and delivery procedures becomes difficult. Thus, “step-and-shoot” is the currently preferred method for delivering IMRT in clinical practices. Advantages of the “step-and-shoot” technique include precise delivery, easy verification, and general availability. A key disadvantage, however, is that it may require a prolonged treatment time, since radiation has to be constantly switched on and off to allow MLC leaves to reshape. In some machines, such as Elekta and Siemens, the intersegment delay dominates the total treatment time in a step-and-shoot delivery. Thus, in such cases, it is highly desirable to use a leaf sequence with the fewest segments possible to deliver the desired intensity map.32

This paper is one of a series of papers on new segmentation algorithms for step-and-shoot IMRT delivery.7,8,18,19,31 The new MLC segmentation algorithm, called SLS (an abbreviation for static leaf sequencing) seeks to shorten the treatment times of IMRT plans by significantly reducing the number of aperture shapes used. The input to SLS is intensity maps computed by current planning systems, and its output is optimized leaf sequences.

In contrast to the known segmentation algorithms, SLS is based on graph algorithmic techniques in computer science and is designed to take advantage of the geometric structures.
of intensity maps. Intuitively, in SLS, intensity maps (or IMBs) are viewed as 3-D “mountains” made of unit-sized “cubes.” Such a 3-D “mountain” (possibly with a complex shape) is first partitioned into a set of special-structured submountains using a new mixed partitioning scheme. Then the optimal leaf sequences for each resulting submountain are computed by a shortest path algorithm or a maximum flow algorithm based on a graph model. Treating an intensity map as a three-dimensional solid has also been considered by Siochi; however, his “rod pushing” algorithm is not based on graph algorithmic techniques.

We have also developed a SLS software package using C programming language on Linux workstations. We conducted comparison studies of SLS with CORVUS versions 4.0 and 5.0, a popular commercial planning system developed by NOMOS Corp., and with our implementation of the Xia and Verhey algorithm. Our comparisons between SLS and CORVUS indicated that for the same intensity maps, the numbers of aperture shapes computed by SLS are up to 80% less than CORVUS 4.0 and 40% less than CORVUS 5.0 (31% less than CORVUS 5.0 with the leakage correction turned off) on Elekta Linac systems. Analysis based on software simulations, delivered films, and ion chamber measurements showed that the 3-D doses produced by modified IMRT plans of SLS match those of CORVUS, and thus confirmed the clinical feasibility of SLS. The comparisons between SLS and the Xia and Verhey algorithm also showed significant improvements. For the same set of intensity maps under the Elekta constraints, SLS outperformed the Xia and Verhey algorithm by about 33% in terms of the numbers of aperture shapes.

Based on these encouraging studies, we have started to apply SLS clinically in the Department of Radiation Oncology at the University of Maryland Medical Center. For example, for one of the clinical cases that we have treated on an Elekta SL20 Linac system, SLS shortened the treatment time of the CORVUS 4.0 IMRT plans from 72 to 25 min.

The rest of the paper is organized as follows. In Sec. II (Methods and Materials) we present our new SLS algorithm/software. In Sec. III (Results) we give some comparison results between SLS and CORVUS or the Xia and Verhey algorithm. In Sec. IV we conclude the paper and discuss some of the possible extensions of our new segmentation approach.

II. METHODS AND MATERIALS

A. The SLS segmentation algorithm

In this section, we present our new segmentation algorithm SLS. Recall that the goal of a MLC segmentation algorithm is to generate the minimum number of aperture shapes (also called segments) for creating a desired intensity map. While most known approaches formulate the problem as a matrix partition problem (i.e., partitioning the intensity map matrix into a set of special matrices, each defining an aperture shape), our approach is very different in the sense that it exploits the geometric nature of the problem.

1. A geometric viewpoint of the segmentation problem

Geometrically, one may view a MLC aperture shape as a polygon on the xy plane that also has a uniform “height” in the z axis; the “height” of the polygon represents the radiation intensity to be delivered for that aperture shape. In this view, each aperture shape can be treated as a “plateau” in the 3-D space with a uniform height h for some integer h > 0.
2. Main ideas of the SLS algorithm

In this paper, we solve three versions of the segmentation problem. These problems are as follows.

1. The 2-D segmentation problem: Given an IMB with entries of either 0 or 1 (called zero–one IMB), find the minimum number of segments for creating the IMB, such that each segment has a height of 1. See Fig. 5(a) for an example of a zero–one IMB.

2. The Bortfeld–Boyer segmentation problem: Given an arbitrary IMB, find the minimum number of segments for the IMB, such that each segment has one unit of intensity. See Fig. 2 for an example of the Bortfeld–Boyer segmentation problem.

3. The 3-D segmentation problem: Given an arbitrary IMB, find the minimum number of segments for the IMB, such that each segment has an arbitrary integral intensity level.

Observe that both the 2-D segmentation problem and Bortfeld–Boyer segmentation problem are special cases of the 3-D segmentation problem.

In our solution for the 3-D segmentation problem, we first optimally solve the 2-D segmentation problem (Sec. II A 3) and the Bortfeld–Boyer segmentation problem (Sec. II A 4). Here, the word “optimal” means that the optimal output quality of the solutions for these special cases can be proved mathematically. Then, based on our optimal solutions for the special cases, we present an effective segmentation algorithm for the 3-D segmentation problem (Sec. II A 5).

3. Special case I: The 2-D segmentation problem

The first special case of the 3-D segmentation problem that we consider is the 2-D segmentation problem (e.g., see Fig. 5): Given a zero–one IMB (with entries of either 0 or 1), find the minimum number of segments for creating the zero–one IMB, such that each segment has a height of 1. Observe that this 2-D version can be viewed as a polygon partition problem that seeks to partition a given planar domain into the minimum number of desired polygons; here, each resulting polygon corresponds to a MLC aperture shape.
a. Why 2-D segmentation?

The 2-D segmentation problem is important in that an optimal solution for this problem can lead to potential improvements to the segmentation algorithm by Xia and Verhey. To illustrate the importance of this 2-D problem, we first review the main ideas of the Xia and Verhey algorithm. The following is the major steps of their algorithm based on the “reducing level” technique used in the paper of Galvin et al. 14

1. Choose an intensity level $d = 2^i$, where $i = \text{Int}(\log_2(I_{\text{max}})) - 1$ (for any value $I > 1$, $\text{Int}(\log_2(I))$ denotes the power of 2 closest to $I$) and $I_{\text{max}}$ is the maximum intensity level of the IMB under consideration.

2. From the IMB, find an aperture shape with the largest area on the $xy$ plane and with an intensity level $d$; remove this aperture shape from the IMB. (Note that the algorithm by Xia and Verhey was mainly designed to address the Siemens constraints; hence, to adapt their algorithm to satisfying the Elekta constraints, we need to find the largest aperture shape subject to the Elekta constraints in this step.)

3. Repeat steps (1) and (2) on the remaining IMB, until the remaining IMB is empty.

For example, on the IMB in Fig. 5(b), the maximum intensity level $I_{\text{max}} = 7$, $i = \text{Int}(\log_2(I_{\text{max}})) - 1 = 2$, and $d = 2^i = 4$. If one keeps applying the above algorithm to segment the IMB in Fig. 5(b) until $I_{\text{max}}$ is no longer 7, then this segmentation process produces the segments in Fig. 5(c), all having a height of $d = 4$. (We assume in this paper that the MLC leaves always move along the $y$ axis.)

Observe that in the above segmentation process, the Xia and Verhey algorithm essentially performs a partition of the shaded area in Fig. 5(b) (i.e., containing all entries whose values are at least 4). In Fig. 5(b), if one views the shaded area as containing entries that are all marked by 1 and the unshaded areas as containing entries marked by zero, then the Xia and Verhey algorithm is in fact dealing with the 2-D segmentation problem [partitioning the shaded area in Fig. 5(a)]. Thus, the 2-D segmentation problem arises as a key subproblem in the Xia and Verhey algorithm.

In the Xia and Verhey algorithm, a “sliding window”-like method is used to solve the 2-D segmentation problem, which repeatedly finds the largest-area deliverable aperture shapes. Applying the sliding window method to the shaded area in Fig. 5(a) generates six aperture shapes, each with a height of 4 [Fig. 5(c)], while the mathematically optimal solution produced by our algorithm has only three aperture shapes [Fig. 5(d)], cutting down the number of aperture shapes by 50% for this example. This implies that solving the 2-D segmentation problem optimally can lead to improvements to the Xia and Verhey algorithm.

b. The 2-D segmentation algorithm.

Now we present our optimal 2-D segmentation algorithm, called 2-D SLS. The key idea of the 2-D SLS is based on an observation called peak eliminations. For convenience, hereafter in this section, we refer to any zero–one IMB as a polygon $P$ on the $xy$ plane. Note that such a polygon $P$ may contain holes and the boundary edges of $P$ are either vertical or horizontal. We say that a vertical boundary edge $e$ of a polygon $P$ is a left peak if the two horizontal edges of $P$ adjacent to $e$ are both to the left of $e$ [e.g., see Fig. 6(a)]. Similarly, a right peak is a vertical edge of $P$ whose two adjacent horizontal edges are both to its right.

Observe that the existence of left and right peaks is the reason for which the polygon $P$ cannot be delivered by using a single MLC aperture shape. This is because when delivering a MLC aperture shape, each pair of MLC leaves defines an opening of the shape as a continuous vertical strip for the exposure of radiation. Left and right peaks separate the involved vertical strips of the IMB into disjoint pieces, and hence prevent the IMB from being delivered as a single aperture shape. Thus, the key to segmenting $P$ into MLC aperture shapes is to “eliminate” all left and right peaks by par-
tioning $P$ into multiple “nice” polygons (i.e., MLC aperture shapes).

Observe that, to eliminate a left (resp., right) peak $e$, one must divide $P$ into two smaller polygons using a polygonal curve that connects the peak $e$ to some point $p$ on the boundary of $P$, such that $p$ is not to the left (resp., right) of $e$ [Fig. 6(b)]. We call such polygonal curves used in partitioning $P$ the dividing curves. Each dividing curve should be deliverable by MLC leaves, that is, it should consist of edges that are only vertical or horizontal, and the widths of the horizontal edges should fit the width of the MLC leaves. Further, a dividing curve should not introduce new left and right peaks; hence, starting at the left end of the curve and walking along the curve, one should always go rightward and never go leftward (i.e., the curve is from left to right, with possible ups and downs).

Introducing a dividing curve to eliminate a peak of $P$ may increase the number of resulting polygons (MLC aperture shapes). Hence, to partition $P$ into the minimum number of aperture shapes, we need to introduce the minimum number of dividing curves. Since each left or right peak can be eliminated by one dividing curve, and one dividing curve can at best eliminate two peaks (i.e., connecting a left peak to a right peak), it follows that we need to find a maximum set of mutually disjoint dividing curves such that each curve connects a left peak and a right peak [e.g., see Fig. 6(b)]. Note that the dividing curves must be mutually disjoint to avoid introducing redundant aperture shapes. In fact, finding a maximum set of such dividing curves is the key to solving the 2-D segmentation problem. For the remaining unconnected peaks, it is relatively easy to eliminate them by connecting each of them to some nonpeak boundary point of $P$.

To compute a maximum set of connections between the left and right peaks, we formulate the task as a maximum flow problem on a directed grid graph induced by the IMB grid [Fig. 6(c)]. (For discussions on the maximum flow problem and its solutions, see the book of Cormen et al.12) Specifically, we assign a unit flow capacity to all vertices and edges of the grid graph (to be defined below). It can be proved that a maximum flow in this directed grid graph yields a maximum set of sought connections that specifies an optimal solution for the 2-D segmentation problem.

The main steps of our 2-D SLS algorithm are as follows.

1. Construct a directed graph $G$ induced by the underlying grid of the given zero–one IMB. The vertices of $G$ include all intersection points of the IMB grid that are either part of a peak or in the interior of the polygon $P$ corresponding to the IMB. The edges of $G$ include those edges of the IMB grid whose end points are vertices in $G$. Directions are assigned to the edges of $G$ based on the following rules:
   (i) For an edge involving a left peak vertex, its direction is away from the left peak.
   (ii) For an edge involving a right peak vertex, its direction is toward the right peak.
   (iii) The direction of a horizontal edge that does not involve any peak vertex is always from left to right.
   (iv) The direction of a vertical edge that does not involve any peak vertex is always bidirectional.

2. For all vertices corresponding to the same left (resp., right) peak, introduce a source (resp., sink) vertex to $G$, and edges from the source vertex (resp., peak vertices) to those peak vertices (resp., the sink vertex) [e.g., see Fig. 6(c)].

3. Assign a unit flow capacity to each vertex and edge of $G$. Note that this flow capacity assignment ensures that any directed path of a unit flow from a source vertex to a sink vertex in $G$ always connects a left peak to a right peak of $P$.

4. Compute a maximum flow in the directed grid graph $G$ from the source vertices to the sink vertices, and based on the resulting maximum flow, extract the connections between the left and right peaks of $P$.

5. Eliminate the remaining unconnected peaks, and extract the resulting segments.

Observe that a unit flow capacity on each vertex and edge of $G$ guarantees that a maximum flow in $G$ yields a maximum set of mutually disjoint paths (from the sources to the sinks), which specifies precisely a maximum set of sought connections between the left and right peaks of $P$. It can be proved that the above 2-D SLS algorithm takes a polynomial time to solve the 2-D segmentation problem.


Another key special case of the 3-D MLC segmentation problem is the Bortfeld–Boyer version of the segmentation problem.1–6 Given an arbitrary IMB, find a minimum set of segments for creating that IMB, such that each segment has exactly one unit of intensity. Geometrically, the Bortfeld–
FIG. 7. (a) A simple IMB. (b) The IMB in (a) can be partitioned into two segments of a unit height (assuming the MLC leaves move along the y axis). (c) The intensity profiles (IP) of the three columns of the IMB in (a). (d) The delivery options (DO) of the three intensity profiles in (c) and the stitching of their field openings (indicated by arrows); each path following the arrows corresponds to a segment in (b).

Boyler version aims to partition a given 3-D IMB “mountain” into “plateaus” (segments) of a unit height (e.g., see Fig. 2). Note that for the general 3-D segmentation problem, the uniform height of each segment can be any integer \( \geq 1 \) (e.g., in the Xia and Verhey algorithm 32). Unless otherwise specified, in this section (Sec. II A 4), all segments are of a unit height.

a. Why Bortfeld–Boyer segmentation?

Studying the Bortfeld–Boyer segmentation problem is important because the maximum heights of the majority of IMBs used in current clinical treatments are around 5 to 10 and because a mathematically optimal solution for this special case on such IMBs is often very close to an optimal solution for the general 3-D segmentation problem.

b. Main ideas.

Geometrically, one may view a 3-D IMB mountain as consisting of a series of IMB “mountain slices,” each of which is a 2-D object defined on a column of the IMB grid [see Fig. 7(c) for examples of IMB “mountain slices”]. Actually, such an IMB “mountain slice” corresponds to the concept of intensity profile 18,28 or 1-D IMB 18,28.

Let \( S=\{S|j|e| \} \) denote a set of segments that “builds” a 3-D IMB mountain, where \( I \) is an index set. For the Bortfeld–Boyer segmentation problem, since the height of each segment \( S_j \) is 1, \( S_j \), in fact, builds a continuous block of a unit height on every IMB column \( C_j \) that intersects the projection of \( S_j \) on the IMB grid. We call the continuous block on an IMB column \( C_j \) created by a segment \( S_j \) a field opening and denote it by \( B_{ij} \). Note that when delivering a segment \( S_j \), each of its field openings \( B_{ij} \) is delivered by a pair of MLC leaves that is aligned to its corresponding IMB column \( C_j \).

Suppose that one “collects” the field openings created by all segments in \( \{S|j|e| \} \) on an IMB column \( C_j \). Then this “collection” \( \{B_{ij}|i|e| \} \) of field openings actually “builds” the IMB mountain slice (intensity profile) defined on \( C_j \). We call a collection of field openings that builds an IMB mountain slice defined on an IMB column a delivery option of the corresponding intensity profile. For example, the boxes of the delivery options in Fig. 7(d) represent the field openings.

From the above analysis, we can solve the Bortfeld–Boyer segmentation problem by first selecting a “good” delivery option for building each IMB mountain slice, and then somehow “stitching” together the field openings along the consecutive IMB columns to form segments. Here, “stitching” means which of the selected field openings for column \( C_j \) should follow which field opening for column \( C_{j+1} \) in a segment [see Fig. 7(d) for an illustration]. Observe that, if no “stitching” occurs, then a given 3-D IMB mountain can be “built” by letting each selected field opening be a segment. For example, the IMB in Fig. 7(a) can be delivered using six segments, with each segment being one of the field openings in Fig. 7(d). Observe that each time we “stitch” two field openings together (without violating the MLC constraints), the number of segments used is reduced by 1. In Fig. 7(d), four “stitchings” can be done, thus reducing the number of segments to 2 [see Fig. 7(b)].

It can be shown 8 that it is sufficient to locally “stitch” together the field openings of the two selected delivery options for any two consecutive IMB columns, by computing a maximum cardinality matching between the two corresponding sets of intervals, with each interval representing exactly one selected field opening—“the number of segments thus resulted is mathematically minimum with respect to the selected delivery options (with one delivery option for each IMB column). Every segment consists of a sequence of field openings thus stitched together, with one field opening from each of the selected delivery options for a consecutive list of IMB columns.

The above approach for solving the Bortfeld–Boyer segmentation problem, however, faces a major difficulty: The intensity profile of an IMB column often can be created by using any one of a large number of different delivery options, and any such delivery option may possibly be used by an optimal set of segments for creating that IMB. In fact, in the worst case, there can be up to \( N! \) different delivery options for an intensity profile, where \( N \) is the minimum number of field openings needed for creating the intensity profile. 28,33 For instance, the intensity profile in Fig. 8(a) has \( 3! = 6 \) different delivery options in Fig. 8(b).

Our key idea for resolving this difficulty is hinged on a geometric observation. We find out that to compute an optimal solution for the Bortfeld–Boyer segmentation problem, it is sufficient to use only a few special delivery options out of all possible delivery options. Being able to use dramatically fewer delivery options enables our algorithms/software to run very fast.

We need to introduce the notion of left and right MLC leaf positions of an intensity profile as in Webb’s paper. 28 Geometrically, an intensity profile is a functional curve \( f \) that is defined on an interval and consists of only vertical and horizontal edges [e.g., the thick solid curve in Fig. 8(a)]. As we “walk” along such a curve \( f \) from left to right, each unit-length upward vertical edge segment (in our walking direction) is called a left leaf position \( (L \) position), and each unit-length downward vertical edge segment is called a right leaf position \( (R \) position). Note that a vertical edge of length \( L \) on \( f \) consists of \( L \) leaf positions (here, \( L \) is an integer). For convenience, we label the left leaf positions of an intensity profile as \( L_i \)'s, and the right leaf positions as \( R_i \)'s, in increas-
ing order as we traverse its curve from left to right [e.g., see Fig. 8(a)]. Clearly, the numbers of the \( L \) positions and \( R \) positions of each intensity profile are always the same.

It is easy to see that every \( L \) or \( R \) position corresponds to exactly one end point of a field opening in any delivery option for an intensity profile [e.g., see Fig. 8(b)]; that is, all \( L \) and \( R \) positions of the intensity profile correspond to some endpoints of the field openings of each of its delivery options. However, not every end point of the field openings of a delivery option necessarily corresponds to an \( L \) or \( R \) position of the intensity profile; that is, a field opening of a delivery option can have end points that need not be an \( L \) or \( R \) position of the intensity profile.

We call the left (resp., right) end point of a field opening that does not correspond to an \( L \) position (resp., \( R \) position) of its intensity profile a left (resp., right) Steiner point. Steiner points can be somehow specified on an intensity profile (say, randomly) for defining field openings, with the numbers of specified left and right Steiner points being the same. A delivery option with Steiner points has more field openings than a delivery option without Steiner points for an intensity profile. Note that the concept of Steiner points has also been described in Webb’s paper, where they are also called “splits.”

The advantage of using delivery options with Steiner points is that the number of segments may be reduced. For example, the optimal solution in Figs. 9(c)–9(d) is obtained by “breaking” one of the field openings for the intensity profile of the middle IMB column of Fig. 9(a) at some Steiner points [Fig. 9(c)], thus enabling us to connect the bottom two segments in Fig. 9(b) into one segment and reducing the number of segments used.

We call the union of all left (resp., right) Steiner points (if any specified) and \( L \) positions (resp., \( R \) positions) on an intensity profile the set of left (resp., right) field opening end points (FO end points) of the intensity profile, and still label them as \( L_i \)'s (resp., \( R_j \)'s) in the left-to-right order. As shown in the papers, a delivery option for an intensity profile can be formed by a pairing between its left and right FO end points, such that each left FO end point is paired with exactly one right FO end point that is to its right.

**Definition 1 (Canonical delivery option)** Let \[\{\left(L_{i1},R_{i1}\right)\}_{1\leq i\leq k}\] be a delivery option based on a set \( S \) of FO end points for an intensity profile, with \( k=|S|/2 \), such that each field opening \((L_{i1},R_{i1})\) has a unit height. Then the delivery option is said to be canonical if and only if for any two field openings \((L_{i1},R_{i1})\) and \((L_{j1},R_{j1})\), \( i\neq j \), neither the closed interval \([x(L_{i1}),x(R_{i1})]\) is contained in the open interval \([x(L_{j1}),x(R_{j1})]\), nor the closed interval \([x(L_{j1}),x(R_{j1})]\) is contained in the open interval \([x(L_{i1}),x(R_{i1})]\) [e.g., see Fig. 8(b)], where \(x(\cdot)\) denotes the \(x\)-coordinate of an FO end point.

The following facts have been shown in the paper: (1) For any set of left and right FO end points on an intensity profile, the canonical delivery option is unique and is \(\{\left(L_{i1},R_{i1}\right)\}_{1\leq i\leq k}\); (2) it is sufficient to use only the canonical delivery option (among all possible delivery options) for each set of left and right FO end points of the intensity profile of every IMB column to compute a mathematically optimal set of segments for the Bortfeld–Boyer segmentation problem.

**c. The algorithm.**

To highlight some of the key aspects, below we present first a preliminary version of our algorithm for solving the Bortfeld–Boyer segmentation problem. The assumption for this preliminary algorithm is that for the intensity profile of each IMB column, only one set of left and right FO end points is used (this assumption will be removed later, of course).

(1) For the intensity profile of every IMB column, generate the unique canonical delivery option for the given set of FO end points.

**Fig. 9.** (a) An IMB. (b) An optimal solution without using Steiner points. (c)–(d) An optimal solution with Steiner points.
(2) For every two consecutive IMB columns, compute a maximum matching between their canonical delivery options. Here, two field openings can be matched if they satisfy the no interdigitation constraint. (Note that the minimum leaf separation constraint has already been taken into account when a delivery option is generated.) A maximum matching here corresponds to the “stitching” process discussed previously.

(3) Extract the segments for the input IMB based on the matched field openings of the canonical delivery options in the increasing order of consecutive IMB columns.

Now we present our solution to the Bortfeld–Boyer segmentation problem.

Note that, for an IMB column of $n$ entries (pixels), the number of possible locations for Steiner points is $n - 1$. Since multiple Steiner points may be inserted at the same location, the number of possible sets for inserting $m$ Steiner points onto the intensity profile is bounded by $O(n^m)$. Our experience based on clinical datasets actually indicated that for IMRT planning, the number $n$ is around 10 and the number $m$ of Steiner points needed for producing an optimal set of segments is usually no bigger than 5. Hence, to compute an optimal set of segments for an IMB, we only need to use a few tens of thousands of different sets of Steiner points to form the sets of FO end points. For each such set of FO end points, we only use its unique canonical delivery option. In this way, using Steiner points in our algorithm does not create a severe problem to the running time. In fact, our program runs in only a few minutes using the sets of FO-endpoints thus generated. Yet, the usage of Steiner points reduces the number of segments by 30% compared to not using Steiner points.

To distinguish from our algorithm for the general 3-D segmentation problem, we call the following algorithm for the Bortfeld–Boyer segmentation problem the basic 3-D SLS.

(1) Choose the number $m$ of Steiner points for forming sets of FO end points.

(2) Generate the canonical delivery option for each set of FO end points containing at most $m$ Steiner points for the intensity profile of every IMB column.

(3) Construct a directed graph $G$, such that each vertex of $G$ corresponds to exactly one delivery option generated in step (2).

(4) For every two vertices of $G$ corresponding to delivery options for the intensity profiles of any two consecutive IMB columns $C_j$ and $C_{j+1}$, put a directed edge from the vertex for column $C_j$ to the vertex for column $C_{j+1}$, and assign a weight to that edge based on a maximum matching between the two sets of field openings of the two corresponding delivery options. (Note that $G$ thus constructed is acyclic.)

(5) A shortest path in $G$ from a vertex of the first IMB column to a vertex of the last IMB column then specifies an optimal set of segments for the input IMB. (For discussions on the shortest path problem and its algorithms, see the book of Cormen et al.13) Note that when Steiner points are used, the intensity profile of each IMB column has many sets of FO end points and thus many canonical delivery options (one canonical delivery option for each set of FO end points). Then finding an optimal set of segments becomes a shortest path problem in $G$.

The above basic 3-D SLS algorithm solves the Bortfeld–Boyer segmentation problem in a polynomial time of $n$ if $m$ is a constant (e.g., $m \approx 5$).

5. The general 3-D segmentation algorithm

Our algorithm for the general 3-D segmentation problem is crucially based on our two algorithms in Secs. II A 3 and II A 4. To be able to make use of these two algorithms, we need to partition the input 3-D IMB mountain into a set of “nice” sub-IMBs, such that each such sub-IMB can be handled optimally by one of these two algorithms (i.e., to use our 2-D SLS algorithm for the 2-D segmentation problem, a resulting sub-IMB must have a uniform height; to use our basic 3-D SLS algorithm for the Bortfeld–Boyer segmentation problem, the maximum height of such a sub-IMB should be reasonably small, say, around 5).

A beamlet (i.e., an entry) of the input IMB with a value $i$ is called a $z$ post of height $i$ ($z$ stands for the $z$ axis corresponding to intensity values). We use the following scheme to solve the general 3-D segmentation problem.

(1) Let $I_{\text{max}}$ be the maximum intensity level of the IMB under consideration. If $I_{\text{max}}$ is small (e.g., no more than 5), then go to step (5) and perform option (a) on the IMB.

(2) Choose an intensity level $d$ with $1 \leq d \leq I_{\text{max}}$.

(3) Consider only all $z$ posts of the IMB whose heights are at least $d$ (i.e., for the time being, all $z$ posts lower than $d$ are ignored). “Cut” each “tall” $z$ post (whose height is $\geq d$) at the level $d$, i.e., the part of the $z$ post at levels $> d$ is ignored. [See Fig. 5(b) for an illustration, in which the chosen intensity level is $d = 4$; the shaded portions of the IMB are the “tall” $z$ posts.] Observe that what is now left of the 3-D IMB mountain is a “plateau” consisting of $z$ posts of height exactly $d$.

(4) Treat this “plateau” (of a uniform height $d$) as the input of a 2-D segmentation problem, and partition it into a minimum set of aperture shapes of height $d$ using the 2-D SLS algorithm in Sec. II A 3.

(5) Remove the “plateau” from the 3-D IMB mountain, and recursively process the remaining IMB by using exactly one of the following two options, depending on the maximum height of the remaining IMB.

(a) If the maximum height of the remaining IMB is small, then apply our basic 3-D SLS algorithm (Sec. II A 4) to obtain the segments for the IMB.

(b) Otherwise repeat steps (2)–(5) on the remaining IMB.

A key to the above scheme is how to choose an appropriate level $d$ to form the “plateaus” (for producing a small set of FO end points).
of segments for the input 3-D IMB mountain). There are several possible choices for \( d \): a function of the maximum intensity level \( I_{\text{max}} \) (e.g., \( d = \lfloor I_{\text{max}}/2 \rfloor \)), a function of the volume of the resulting "plateau," a random choice, etc. Our experiments on these choices of \( d \) using various real medical datasets showed that different choices of \( d \) give good results for different types of datasets. To obtain the "best" quality output, our general 3-D SLS algorithm combines various ways of choosing \( d \) in a "mixed" fashion, and we call it the mixed partitioning method. Specifically, in the mixed partitioning method, every time when step (2) of the previous scheme is performed, we use each of the "good" ways to choose \( d \), and the algorithm, based on every chosen value of \( d \), is carried out recursively on the remaining IMB. At the end, our 3-D SLS algorithm selects the best overall result. The number of different ways of choosing \( d \) that we actually use is four. (Of course, based on the characteristics of the intensity maps in hand, one may develop and include/exclude any partitioning methods into/from this scheme.)

The four methods of choosing \( d \) used in our current version of the general 3-D SLS algorithm/software are as follows:

(a) The Xia and Verhey partitioning method. \(^{32}\) For the

<table>
<thead>
<tr>
<th>Number of IMBs</th>
<th>( I_{\text{max}} )</th>
<th>Field size (in cm)</th>
<th>Number of segments</th>
<th>Number of MUs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>CORVUS 4.0</td>
<td>SLS</td>
</tr>
<tr>
<td>Head and neck</td>
<td>5</td>
<td>5</td>
<td>6 \times 6</td>
<td>143</td>
</tr>
<tr>
<td>Thyroma</td>
<td>7</td>
<td>5</td>
<td>10 \times 9</td>
<td>235</td>
</tr>
<tr>
<td>Pancreas</td>
<td>7</td>
<td>5</td>
<td>10 \times 12</td>
<td>381</td>
</tr>
<tr>
<td>Prostate</td>
<td>7</td>
<td>5</td>
<td>9 \times 9</td>
<td>281</td>
</tr>
</tbody>
</table>

Fig. 10. Comparisons of the SLS and CORVUS 4.0 dose profiles. These profiles were plotted using the RIT113 Film Dosimetry System. The vertical axis is for absolute doses in cGy, and the horizontal axis is for locations in cm. The CORVUS dose profiles are in dark, and SLS dose profiles are in light. The profiles in (a) and (b) are orthogonal to the leaf motion direction, and the profiles in (c) and (d) are along the leaf motion direction.
IMB under consideration, choose the power of 2 that is closest to \( I_{\text{max}} / 2 \) [i.e., \( d = 2^i \), where \( i = \text{Int}(\log_2(I_{\text{max}})) - 1 \)].

(b) Que’s half-cutting method:\(^{22} \) \( d = [I_{\text{max}} / 2] \).

(c) The maximum volume method: \( d \) is chosen such that the volume of the resulting plateau is maximized.

(d) The “slice” method:\(^{21,24} \) Let \( d \) be the smallest positive entry of the IMB.

The program for our general 3-D SLS algorithm based on the above mixed partitioning method usually runs in only a few minutes. Yet, the mixed partitioning method helps produce 10%–20% fewer segments according to our experiments.

B. Computer and clinical implementation

To further study the effectiveness and performance of our new 3-D SLS algorithm, we implemented it using the C programming language on a Linux workstation. Our software package is named SLS. SLS is designed to use the CORVUS intensity maps as input, and the Elekta Precise Treatment System\(^{TM} \) (RT Desktop) as the target delivery machine.

A typical IMRT planning process with SLS is carried out using the following major steps.

1. Generate the CORVUS prescriptions and intensity maps.
2. Import the CORVUS intensity maps and compute the SLS aperture shapes (segments) for these intensity maps.
3. Import the CORVUS prescription files and compute the monitor units (MU) for the SLS aperture shapes. For the MU computation, we use two different methods: a full blown Monte Carlo based dosage calculation program\(^{20} \) and a much simpler discrete intensity map guided MU calculation program. The underlying idea of using a Monte Carlo based MU calculation is to make sure that the two leaf sequences (SLS and CORVUS) match each other in 3-D dose, by adjusting the amount of monitor units prescribed. The idea of an intensity map guided MU calculation is to guarantee that the SLS prescriptions match the CORVUS prescriptions in beamlet intensities. Using these two methods for MU calculation enhances the error tolerance of our SLS software.
4. Encode and transfer the SLS prescriptions, by using a DICOM RT protocol, to the radiation delivery machines. For this step, we developed a DICOM encoding and transferring software using the MergeCOM-3\(^{TM} \) Advanced Integrator’s Tool Kit for UNIX of Red Hat Linux systems.

III. RESULTS

After the SLS software package was developed, we conducted experiments and comparisons between SLS and two popular segmentation algorithms/software: the CORVUS inverse planning system (the 4.0 and 5.0 versions) and the Xia

---

**Fig. 11.** Illustrating how MLC leaves and the Y jaws are used for blocking radiation: (a) CORVUS 4.0, and (b) SLS.

**Fig. 12.** Comparisons of dose distributions on the coronal plane. (a) The CORVUS plan isodose lines: 90%, 75%, 60%, 40%, 20%, and 10%. (b) The SLS plan isodose lines: 90%, 75%, 60%, 40%, 20%, and 10%. (c) Overlaying the isodose lines of 75% and 40% in (a) and (b).
and Verhey segmentation algorithm. Some of our experimental results are presented in this section. In general, our results indicated the following.

1. The SLS software was implemented correctly.
2. The treatment plans produced by SLS are feasible for patient treatments.
3. The SLS plans have a significantly smaller number of segments and therefore can be delivered in a much shorter treatment time.

### A. SLS vs CORVUS 4.0 and the clinical feasibility of SLS

We conducted comparisons between our SLS software and the CORVUS 4.0 planning system used in the Department of Radiation Oncology, University of Maryland School of Medicine. A main objective of this study was to verify the clinical feasibility of the SLS software package. Hence, the comparisons were largely focused on whether the plans computed by SLS match the original CORVUS plans in a 3-D dose.

Table I gives some of our comparison results, which show that SLS significantly outperforms CORVUS 4.0 in terms of the number of segments computed. In Table I, the numbers of segments computed by SLS are up to 80% less than CORVUS 4.0. As a result, the treatment time has also been improved substantially by the SLS plans. For example, for the pancreatic case in Table I, the treatment time was shortened from 72 min for the CORVUS 4.0 plans to 25 min for the SLS plans. Note that the treatment time was measured on an Elekta Linac system, and should be accelerated on Siemens and Varian machines. We believe that even for machines without a large intersegment delay as on Elekta, the substantial reduction of the number of segments used (e.g., from 381 to 84 as the pancreas case in Table I) should significantly shorten the treatment time.

Generally speaking, the dose distributions created by SLS plans and CORVUS 4.0 plans for the same set of intensity maps are very similar. The difference between the isodose lines is within 2 mm, and the difference of doses measured by ion chamber at the isocenters is within 5%. Below we use a pancreatic case as an example to show the similarities between the SLS plans and the original CORVUS plans in terms of their dose distributions.

Figure 10 shows the dose profiles of the CORVUS 4.0 and SLS plans for a pancreatic case.

As can be seen from the figure, these dose profiles are very similar. The bigger differences exhibited that are close to the left and right ends of the dose profiles are due to the leakage of the CORVUS 4.0 plans. These differences are outside the beams, and are caused by the way that CORVUS 4.0 uses the MLC leaves and backup diaphragms to block radiation. Note that on the Elekta Linac system, MLC leaves cannot be completely closed and must maintain a minimum leaf separation distance. Hence, to achieve the effect of fully blocking the radiation through a pair of MLC leaves, we need to bring in the backup diaphragms. The backup diaphragms are also referred to as the X and Y jaws (the Y jaws have a 10% leakage). The way the Y jaws are used for blocking leaf openings in CORVUS 4.0 is shown in Fig. 11(a). Basically, CORVUS 4.0 assumes that the Y jaws and MLC leaves are “perfect” blockers. In contrast, the way the Y jaws are used for blocking leaf openings in SLS is shown in Fig. 11(b). The shaded area in Fig. 11(a) indicates the place in which most leakage occurs. As can be seen, CORVUS 4.0 has more leakage, especially around the boundaries of the beams.

Figure 12 shows the CORVUS 4.0 and SLS dose distributions and the comparisons on the coronal plane for the same pancreatic case. In this figure, the isodose lines of these two types of plans are very similar. Based on our measurement, the difference between the two 75% lines is within 2 mm.

Since the dose measured by films has a 5%–10% error, we also performed an ion chamber measurement at the isocenters. In this case, the dose measured for the SLS plan is 183 cGy, while the CORVUS 4.0 prescribed dose is 178 cGy. The difference is within 3%.

### B. SLS vs CORVUS 5.0 and the Xia and Verhey algorithm

We also conducted comparisons between SLS and CORVUS 5.0 as well as the Xia and Verhey segmentation algorithm. The focus of these comparisons was primarily on the number of segments computed by each algorithm/software.

CORVUS 5.0 is the latest version of the CORVUS planning system. The comparisons were carried out at NOMOS Corp. in Pittsburgh. We also chose to compare SLS with the Xia and Verhey algorithm because their algorithm has been coded by many people and has been used as a generally

### Table II. SLS vs CORVUS 5.0 (with and without leakage correction) and the Xia and Verhey algorithm.

<table>
<thead>
<tr>
<th>Number of IMBs</th>
<th>Maximum intensity level</th>
<th>Field size (in cm)</th>
<th>CORVUS 5.0</th>
<th>SLS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>With leakage correction</td>
<td>Without leakage correction</td>
</tr>
<tr>
<td>Brain</td>
<td>5</td>
<td>5</td>
<td>2×3</td>
<td>22</td>
</tr>
<tr>
<td>Head and neck</td>
<td>6</td>
<td>10</td>
<td>4×5</td>
<td>64</td>
</tr>
<tr>
<td>Prostate</td>
<td>5</td>
<td>5</td>
<td>16×15</td>
<td>262</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7×8</td>
<td></td>
<td>72</td>
</tr>
</tbody>
</table>

Medical Physics, Vol. 31, No. 4, April 2004
accepted benchmark program. In our study, we implemented the Xia and Verhey algorithm with the Elekta constraints based on the reducing level method.

Table II gives the comparison results. Since the current version of the SLS software does not yet include the tongue and groove correction, for a fair comparison, we include CORVUS 5.0 results, both with the leakage correction and without the leakage correction. As shown by Table II, SLS outperforms CORVUS 5.0 by a factor of about 40% (31% when the leakage correction is turned off) in terms of the number of segments computed, and outperforms the Xia and Verhey algorithm by a factor of about 33%. On all the cases, SLS performs better than CORVUS 5.0 and the Xia and Verhey algorithm in various degrees.

IV. CONCLUSION

In this paper, we presented a new segmentation algorithm, called SLS, for step-and-shoot IMRT delivery. Unlike previous segmentation approaches, SLS is based on graph algorithmic techniques in computer science, and exploits the geometric nature of the problem. Our implementation and experiments of SLS on the Elekta Linac system showed that the SLS software substantially outperforms CORVUS (versions 4.0 and 5.0) and the Xia and Verhey algorithm in terms of minimizing the number of MLC aperture shapes and shortening the treatment time. Film and ion chamber measurements proved that the SLS plans match well with the CORVUS commercial planning system in 3-D composite dose and hence paved the way for the clinical applications of SLS.

For future research, we plan to do the following.

1. We plan to extend our SLS algorithm/software to Siemens and Varian Linac systems. We believe that the treatment time should be significantly reduced due to the less strict MLC constraints and shorter intersegment delays on these two systems.

2. The SLS algorithm as in its current state does not take account of the “tongue and groove” effect.29,32 This is also evident in Fig. 12, where the 90% isodose lines are continuous for CORVUS and broken up for SLS. Therefore, for future work, we also plan to integrate the “tongue and groove” consideration into our algorithm. While the effect of adding the tongue and groove corrections to our current SLS algorithm is not completely clear at this point, we do notice that the number of segments tends to increase by a factor of 10%–20% based on the estimates by several authors23,32 on other algorithms.

ACKNOWLEDGMENTS

The authors are very grateful to the anonymous reviewers for their highly constructive comments and suggestions. This work was supported in part by the National Science Foundation under Grant No. CCR-9988468. The work of the first author was also supported in part by a fellowship from the Center for Applied Mathematics at the University of Notre Dame.

Electronic mail: sluan@cse.nd.edu


