The Dual of Sweep

Horea T. Ilies Vadim Shapiro
Spatial Automation Laboratory
1513 University Avenue
University of Wisconsin-Madison, 53706 USA
Email: ilies@cae.wisc.edu, vshapiro@engr.wisc.edu

May 11, 1998

Abstract

As an infinite union operation, `sweep` of a moving object through space is a powerful and natural addition to the Boolean set operations that incorporates motion-related information for the purposes of shaping, collision detection, and simulation of moving objects. Use of `sweep` has been hindered by limited computational support and by the fact that it is a 'material growing' operation, whereas many applications, such as interference elimination and mechanism design, appear to be better modeled by a 'material removal' operation.

This paper formally defines a new geometric modeling operation of `unsweep`. Given an arbitrary subset E of Euclidean space and a general motion M, $\texttt{unsweep}(E, M)$ returns the largest subset of E that remains inside the original E under M. When M is a translation, $\texttt{unsweep}(E, M)$ naturally reduces to the standard Minkowski difference of E and the trajectory generated by the inverted motion $\hat{M}$. In this sense, the operation of `unsweep` is a generalization of Minkowski difference that corresponds to a 'material removal' operation, it can be also defined as an infinite intersection operation, and is the dual of `sweep` in a precise set-theoretic sense.

We show that `unsweep` has attractive theoretical and computational properties, including a practical point membership test for arbitrary general motions. By duality, the established properties of `unsweep` extend to the general `sweep` operation and can be used to improve the computational support for general sweeps.

Keywords: `unsweep`, `sweep`, shaping, geometric and solid modeling, motions, point membership classification, set operations.

1 Introduction

1.1 Shaping with motions

Sweeping a set of points along some trajectory is one of the fundamental operations in geometric and solid modeling. If M is a path of configurations for a moving set of points S, then the sweep of S along M is the set of points swept (or occupied) by S at some time during the motion. Formally,

$$\texttt{sweep}(S, M) = \bigcup_{q \in M} S^q \tag{1}$$

where S^q denotes set S positioned according to q. Sweeps are considered to be one of the basic representation schemes in [20], and have numerous applications in graphics, geometric modeling, mechanical design and manufacturing, and motion planning. Sweeps are used extensively to construct and model surfaces and solids in both academic and commercial systems [28, 7, 2]. In graphics, allowing object S to deform as it moves along M is often used to generate complex scenes and visual effects [23, 24]. In mechanical design, sweeps of moving parts can be used for collision detection [6] in assemblies. Sweeping a solid (cutter) along the specified trajectory (tool path) is the preferred method of NC machining simulation[29, 17]. Finally, sweeps arise naturally in most situations involving moving bodies, e.g. in studies of robot workspace [3].

Despite their usefulness, properties of general sweeps (notably, their validity and computational properties) are not well understood. Several methods for generating candidate surfaces bounding the sweep are known [1, 15, 24]. But general and reliable procedures for a point membership classification (PMC) [26] to determine if a given point is in, on, or out of the sweep defined by expression (1) appear to be identified only in special and restricted situations [7]. Numerous approaches to computing sweeps have been published, including:

- restricting the type of the moving object S, for example to a ball, a convex polyhedron, or a planar cross-section that remains orthogonal to the trajectory;
- allowing only simple motions M that prevent self intersections in the sweep, or limiting them to simple translations and rotations;
- formulating PMC procedures in terms of heuristic numerical sampling and searching algorithms;
- approximating expression (1) by a discrete union of S^q computed at a finite number of locations q;
- using rendering methods to compute the image of the sweep without computing the complete representation of the sweep;
- combinations of some or all of the above.

By definition, sweeping a moving object S through a motion M requires that both the object S and motion M are given. The result of sweeping is always a set of points containing the original object S; in other words, `sweep` is a 'material growing' operation. By contrast, many engineering design and planning applications require that the shapes of the moving objects be determined based on some criteria for fit, containment, non-interference, and so on. As we explain in section 1.3, using `sweep` in such applications may force inefficient and expensive trial-and-error solutions. Perhaps, a more appropriate 'material removal' operation would start with a sufficiently large space and carve away the unwanted portions based on some well-defined criteria.

In this paper, we introduce a new operation called `unsweep` that is dual to the general `sweep`; it corresponds to a material removal operation and has many applications such as collision and interference detection of arbitrary objects in general motion, design of objects in relative motion, design of moving mechanical parts, and shape synthesis. It is important that `unsweep` comes with a relatively straightforward PMC procedure. By duality, the same PMC procedure extends to general sweeps, while all known results and methods for dealing with sweeps also apply to the dual `unsweep`.

1.2 The dual of `sweep`

If set S and motion M are defined as before, and $\hat{M}$ is the inverted motion[1], then the dual of the `sweep` is obtained by changing union to intersection and replacing motion M by the inverted motion $\hat{M}$ in expression (1):

$$\texttt{unsweep}(S, M) = \bigcap_{q \in \hat{M}} S^q \tag{2}$$

The precise nature of this duality is explained in section 3.2. Definition (2) is not particularly revealing: it may not be obvious why this new operation is useful, why it may possess computational advantages over `sweep`, and what is the precise relationship between the two dual operations. We study these and other related issues below and show that:

1. `unsweep`(S, M) is the largest set of points that remains inside S under M;
2. PMC of a point p against `unsweep`(S, M) reduces to classifying the trajectory of p under M (a curve) against the set S;

[1]The concepts of motion and inverted motion are discussed in section 2.

3. if X^c denotes the complement of a set X, then the relationship between the two dual operations is given by:

$$[\texttt{unsweep}(S^c, \hat{M})]^c = \texttt{sweep}(S, M) \tag{3}$$

The first property suggests that `unsweep` has many practical applications in modeling. It implies that `unsweep` corresponds to a material removal operation and that the shape of the moving object is generated by `unsweep` and is not required *a priori*. The second property indicates that `unsweep` can be computed effectively, either exactly (within the machine precision) or approximately (by using standard approximation methods like octrees, marching cubes etc). The third property extends the theoretical properties and computational advantages of `unsweep` to general sweeps and vice versa.

1.3 Example applications

Packaging is one of most common problems in mechanical design. Static interference of fixed parts, say within a single assembly or containing set, can be determined in a straightforward fashion by computing intersection of the corresponding solid models. Packaging of moving parts is more difficult but can be formulated using `sweep` and now `unsweep` operations. A typical situation is shown in Figure 1(a). Given a completely designed part S and the containing set E, S has to fit inside E while it is moving according to the motion M. A common way to approach this problem has been to test if $E \cap \texttt{sweep}(S, M) = \emptyset$, as illustrated in Figure 1(b).

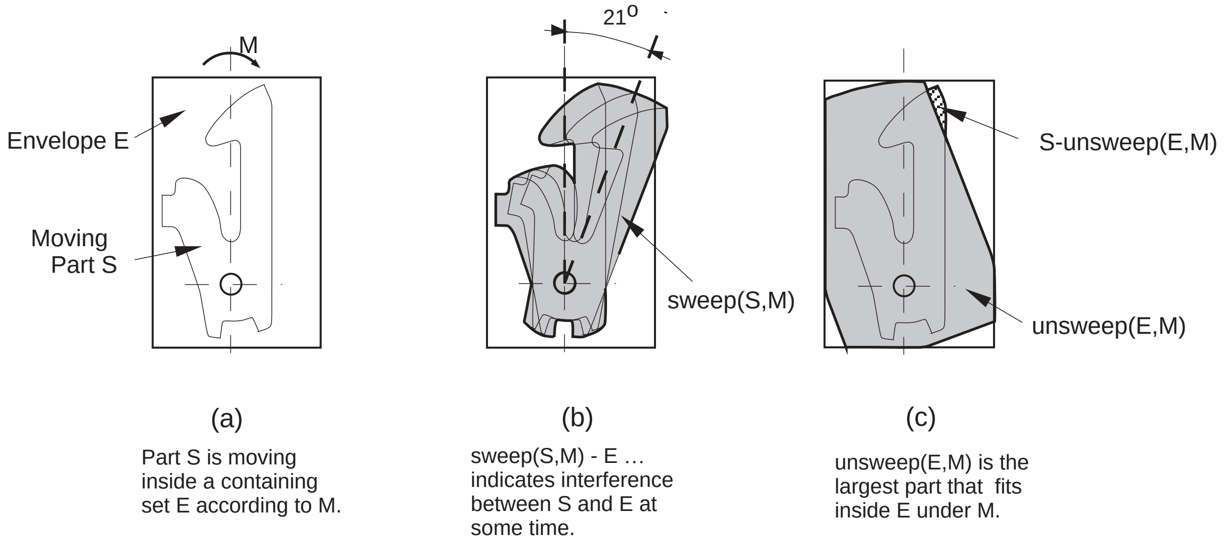

M
Envelope E
Moving Part S
21°
sweep(S,M)
S-unsweep(E,M)
unsweep(E,M)
(a)
Part S is moving inside a containing set E according to M.
(b)
sweep(S,M) - E ... indicates interference between S and E at some time.
(c)
unsweep(E,M) is the largest part that fits inside E under M.

Figure 1: The moving part S has to fit within the given containing set E

In the general case, this test may be difficult to implement due to computational limitations of sweeps mentioned above. Even when fully implemented, the result of the test is binary: 'yes' or 'no', and, when the answer is 'no', it does not suggest how the moving part should be modified to fit inside the containing set. For this task, the best tools available today to human designers are their experience and intuition. This approach to design almost always forces inefficient and costly iterations in the design process [8].

Perhaps a more efficient way to approach this design problem is to ask what is the largest part S that would fit inside E under the specified motion M — and use this information in deciding how to shape the final product. As is often the case, the complexity of this design problem usually exceeds the human intuition and/or experience. The solution to this problem is readily found by computing $\texttt{unsweep}(E, M)$ and is shown in Figure 1(c).

In computer graphics and geometric modeling `unsweep` can be used to synthesize shapes with desired aesthetic and mathematical properties. Starting with some already constructed geometric object S, the user specifies some hypothetical (virtual) motion M that may or may not have physical significance. The motions do not have to be rigid, but may include scaling, deformation, perspective, visibility and other properties. Then computing `unsweep`(S, M) yields a new shape that inherits and combines the properties of S and M. For example, when S is the complement of a linear halfspace and M is a 360 degrees rotation around an axis parallel to the plane, `unsweep`(S, M) produces an infinite solid cylinder; unsweep of this cylinder by another rotation can produce a solid sphere, and so on. Figure 2(a) shows the `unsweep` of a cylinder by

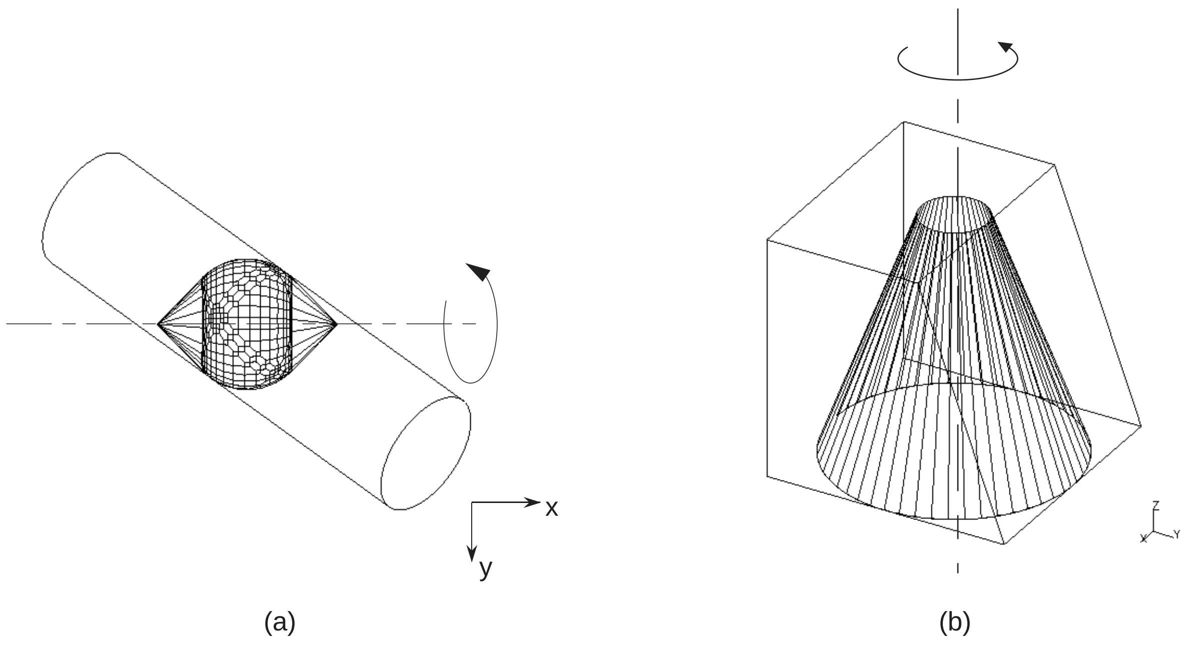


Figure 2: `Unsweep`(S, M) shapes solids that inherit and combine properties of generator S and motion M.

a 360 degrees rotation around the axis shown; unsweeping a prismatic solid S by a 360 degrees rotation, as illustrated in Figure 2(b), produces a trimmed cone. More complex objects and motions will produce increasingly complicated and interesting results.

The above examples are only some of many applications of the new operation. Although unsweeping S with a given motion has a "carving" effect on S, we shall see in section 3.2 that the resulting set does not have to be dimensionally homogeneous. In general terms, `unsweep` gives a method for creating and modifying geometric shapes and provides a new computational utility for many motion-intensive applications.

1.4 Scope and outline

Our main goals are the rigorous definition of `unsweep`, exploring its theoretical and computational properties in a general setting, and understanding the relationship between the operations of `sweep` and `unsweep`. For the large part, we will avoid putting restrictions on the generator set S or motion M. Additional properties, such as regularity, and representational issues may be required for specific applications; they are outside of the scope of this paper. We chose to present our results in the set-theoretic terms; in this sense, they are independent of a particular choice of representation schemes for S and M and are applicable to all valid and unambiguous representations. Representational issues are important for effective implementation of `unsweep`, but the range of possibilities is too great for discussion in this paper. We only mention some of them.

The paper is organized as follows. In section 2 we explain that every moving set of points can be viewed and modeled in two distinct ways; this leads to the notions of motions and inverted motions, and the corresponding trajectories of the moving points. The introduced notions are then used to reexamine the usual concept of `sweep` and to define the dual operation of `unsweep`. Set theoretic properties of `unsweep` are discussed in section 3 and the precise nature of the duality relationship between `sweep` and `unsweep` is identified in section 3.2. Computational issues, including point membership classification, several strategies for implementing `unsweep` and some 3-dimensional examples are discussed in section 4. The concluding section 5 briefly discusses the significance of our findings and a number of promising extensions of this work.

2 Formulation

2.1 Motions and trajectories

Consider a set of points S with its own coordinate system $\mathcal{F}_S$ moving in a d-dimensional Euclidean space $\mathcal{W}$ with respect to some global fixed coordinate system $\mathcal{F}_\mathcal{W}$. Following the notation in [12], we define the motion $M(t), t \in [0,1]$ as a one-parameter[2] family of transformations in the higher-dimensional configuration space $\mathcal{C}$. For the purposes of this paper, "motions" and "transformations" are interchangeable and are commonly represented by matrices, as discussed in section 2.2. We mostly use rigid body motions for illustration purposes but, except when noted, all discussions and results apply to general non-singular affine transformations in E^d.

At every instant $t = a$, the original point x of S moves to a new location that is determined by the transformation $M(a)$. We will use the superscript notation to define the transformed (set of) points as

$$x^{M(a)} = M(a)x; \qquad S^{M(a)} = M(a)S \tag{4}$$

The transformation $q \in M(t)$ for some instantaneous value t determines the position and orientation of $\mathcal{F}_S$ with respect to $\mathcal{F}_\mathcal{W}$ at that instance and therefore determines the coordinates of every point x^q of the moving set S with respect to $\mathcal{F}_\mathcal{W}$. The identity transformation is a member of the family $M(t), t \in [0,1]$ and $M(0)S = S$. By definition, a point $x \in S$ is located at $x^{M(0)}$ with respect to $\mathcal{F}_\mathcal{W}$.

In the special, but common, case of a rigid body motion in the three-dimensional Euclidean space, each transformation $M(a)$ specifies rotation and translation of S at time a with respect to $\mathcal{F}_\mathcal{W}$. A rigid body motion in a d-dimensional space is determined by $\frac{d(d+1)}{2}$ independent degrees of freedom, as a path in the configuration space $\mathcal{C}$. Mathematical properties of such a configuration space are well understood [12].

For a *range* of values of t, $M(t)$ is a *subset* of the configuration space $\mathcal{C}$. For brevity we may denote the set $M(t)$, $t \in [0,1]$ simply as M. A motion M specifies how the moving coordinate system $\mathcal{F}_S$ moves with respect to the fixed coordinate system $\mathcal{F}_\mathcal{W}$. Every point x of set S moves with respect to $\mathcal{F}_\mathcal{W}$ according to M, as is illustrated in Figure 3(a)[3]. As t goes from 0 to a, the moving point $x^{M(t)}$ sweeps, with respect to the fixed frame $\mathcal{F}_\mathcal{W}$, a set of points T_x called the *trajectory* of x and defined as

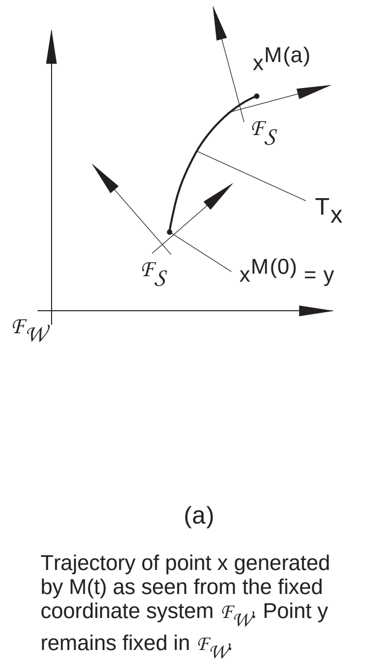
$$T_x \equiv M(t)x = \bigcup_{q \in M} x^q \tag{5}$$

Each instantaneous transformation $M(a)$ has a unique inverse $\hat{M}(a)$ such that $x = \hat{M}(a)[M(a)x]$. Given a transformation $M(t)$ for a range of values of $t \in [0,1]$, we will call transformation $\hat{M}(t)$ *inverted* if it is the inverse of $M(t)$ for every instance of t.

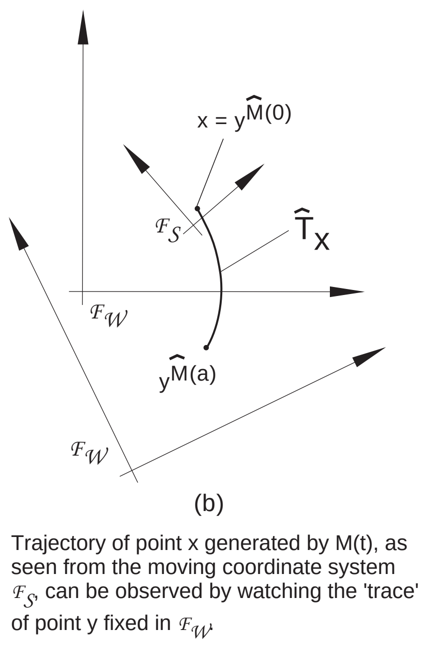
Consider the point y as being the copy of point $x \in S$ in $\mathcal{F}_\mathcal{W}$ at the initial configuration. Point y is fixed in $\mathcal{F}_\mathcal{W}$ and does not move with the object S, but rather it moves *relative* to S. To an observer placed at the origin of $\mathcal{F}_S$, the moving point $x \in S$ will appear to be fixed while the fixed coordinate system $\mathcal{F}_\mathcal{W}$ and point y fixed in $\mathcal{F}_\mathcal{W}$ will appear to be moving according to the inverted motion $\hat{M}(t)$. When point x moves to point $x^{M(a)}$, the moving observer sees point y moved to $y^{\hat{M}(a)}$. And since $x = y$, the trace of moving point x as seen from the moving coordinate system (Figure 3(b)) is the inverted trajectory

$$\hat{T}_x = \hat{M}(t)x = \bigcup_{q \in \hat{M}} x^q \tag{6}$$

[2]We normalize all intervals $[a,b]$ to $[0,1]$, so that the results presented here remain valid when $t \in [a,b]$, $0 < a < b$.
[3]Two-dimensional examples are used for clarity, but the same arguments hold in higher dimensional spaces.



$x^{M(a)}$
$\mathcal{F}_S$
T_x
$\mathcal{F}_S$
$x^{M(0)} = y$
$\mathcal{F}_W$
(a)
Trajectory of point x generated by M(t) as seen from the fixed coordinate system $\mathcal{F}_W$. Point y remains fixed in $\mathcal{F}_W$.



$x = y^{\hat{M}(0)}$
$\mathcal{F}_S$
$\hat{T}_x$
$\mathcal{F}_W$
$y^{\hat{M}(a)}$
$\mathcal{F}_W$
(b)
Trajectory of point x generated by M(t), as seen from the moving coordinate system $\mathcal{F}_S$, can be observed by watching the 'trace' of point y fixed in $\mathcal{F}_W$.

Figure 3: Motions and trajectories

To paraphrase, the trajectory of the moving point x, observed from $\mathcal{F}_\mathcal{W}$, is generated by the motion M, while the trajectory of the same point x, observed from $\mathcal{F}_S$, is generated by the inverted motion $\hat{M}$. Intuitively, T_x represents the trace left by moving point x as seen from $\mathcal{F}_\mathcal{W}$, while $\hat{T}_x$ is the trace of x as seen from $\mathcal{F}_S$. Therefore, observed from the fixed coordinate system $\mathcal{F}_\mathcal{W}$, the points x of S are moving according to M while, observed from the moving coordinate system $\mathcal{F}_S$, the 'world' appears to be moving according to $\hat{M}$. We emphasize that, in general, the two trajectories T_x and $\hat{T}_x$ are quite different sets of points from one another and the relationship between T_x and $\hat{T}_x$ is not elementary, except maybe in some very special cases. For example, when M is a pure translation, the two trajectories T_x and $\hat{T}_x$ are simply reflections of one another with respect to point x.

In the above discussion, motion M can be considered as a special case of a more general *relative* motion with the coordinate system $\mathcal{F}_\mathcal{W}$ fixed in $\mathcal{W}$. If $\mathcal{F}_\mathcal{W}$ were not fixed, then motion M would represent the relative motion between $\mathcal{F}_S$ and $\mathcal{F}_\mathcal{W}$. Let M_S and M_W be the *absolute* motions of $\mathcal{F}_S$ and $\mathcal{F}_\mathcal{W}$ relative to a fixed coordinate system in $\mathcal{W}$. The relative motion between $\mathcal{F}_S$ and $\mathcal{F}_\mathcal{W}$ can be expressed in any coordinate system defined in the same space, but usually it is convenient to have it expressed in one of the two moving coordinate systems. Then, the relative motion is expressed in $\mathcal{F}_S$ by

$$M_{S/W} = {M_W}^{-1} M_S \tag{7}$$

and, consequently, in $\mathcal{F}_\mathcal{W}$ by

$$M_{W/S} = {M_S}^{-1} M_W. \tag{8}$$

Equation (7) expresses the motion of $\mathcal{F}_S$ observed from $\mathcal{F}_\mathcal{W}$, while equation (8) expresses the motion of $\mathcal{F}_\mathcal{W}$ as observed from $\mathcal{F}_S$. Note that both equations (7) and (8) express the *same* physical relative motion between $\mathcal{F}_S$ and $\mathcal{F}_\mathcal{W}$, but in different coordinate systems.

2.2 Representing Motions and Trajectories

General affine transformations in E^d can be represented as linear transformations in projective space using homogeneous coordinates and $(d+1) \times (d+1)$ matrices [5]. Thus, if motion $M(t)$ is given by a matrix $A(t)$, then the inverted motion $\hat{M}$ is given by the inverse of this matrix $A^{-1}(t)$. In the case of a rigid body motion in E^3, we have

$$A(t) = \begin{bmatrix} & \Theta(t) & & \mathcal{T}(t) \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad A^{-1}(t) = \begin{bmatrix} & \Theta^T(t) & & -\Theta^T(t)\mathcal{T}(t) \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{9}$$

where $\Theta(t)$ and $\mathcal{T}(t)$ represent the rotational and translational components of the motion M(t). In the case of pure rotation, $\mathcal{T}(t) = 0$, and $A^{-1}(t)$ is obtained from $A(t)$ by replacing the orthonormal sub-matrix $\Theta(t)$ with its transpose $\Theta^T(t)$. When $M(t)$ is a pure translation, $\Theta(t)$ is the identity, and $A^{-1}(t)$ is obtained from $A(t)$ by replacing $\mathcal{T}(t)$ with its reflection $-\mathcal{T}(t)$.

If a point x is represented by a vector $\vec{v}$, and a motion $M(t)$ is represented by a matrix $A(t)$, the trajectory T_x can be written in parametric form simply as $A(t) \cdot \vec{v}$. The parametric form of the curve T_x is suitable for computing the intersection of T_x with the boundary of a given set (typically solid) S, as needed for the PMC procedure developed above. Such a representation of T_x may use trigonometric functions, or it may be useful to consider under what conditions T_x can be re-parameterized in a computationally more convenient form, for example as a rational parametric curve [10].

2.3 Sweep

Following the above notation and definitions from equations (4) and (5), the 'trace' left by a set of points S that is moving according to $M(t)$, $t \in [0, 1]$, is given by

$$M(t)S = \bigcup_{q \in M} S^q \tag{10}$$

which is immediately recognized as definition (1) of the general sweep given earlier. A rectangular 2-dimensional set S in general motion M with respect to a fixed coordinate system is shown in Figure 4(a). This formulation assumes that the sweep is computed from the fixed coordinate system $\mathcal{F}_{\mathcal{W}}$, 'observing' the moving coordinate system $\mathcal{F}_S$.

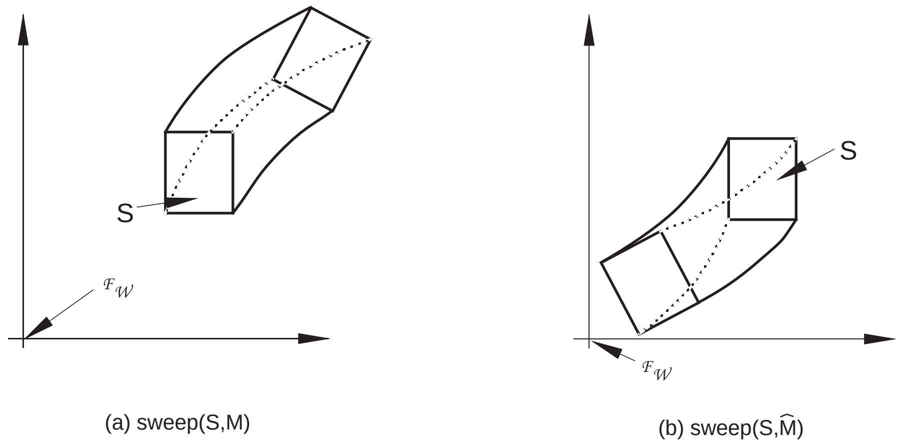


Figure 4: Sweeping a set according to motions M and $\hat{M}$

We could also define sweep, as observed from the moving coordinate system $\mathcal{F}_S$, by

$$\texttt{sweep}(S, \hat{M}) = \bigcup_{q \in \hat{M}} S^q \tag{11}$$

The set of points defined by equation (11) is quite distinct from that defined in equation (10). Figure 4(b) shows the sweep of set S that moves according to the inverted motion $\hat{M}$. We will see in section 3.2 that there is also a computationally convenient interpretation of the usual sweep defined by equation (10) in terms of the inverted trajectories of moving points (i.e. as observed from the moving system $\mathcal{F}_S$). Notice that, in both definitions of `sweep`, the trajectories of distinct points of S generated by the same transformation $M(t)$ need not be congruent, and, in general, the relationship between these trajectories is not simple. In section 3.1, we will consider the case when M is a pure translational motion, and all points of S do move on the trajectories that are translation invariant.

2.4 Unsweep

Consider now a set of points E fixed in the global coordinate system $\mathcal{F}_\mathcal{W}$ and a different set F of points that are moving according to some transformation $M(t),\ t \in [0,1]$. Figure 5(a) shows a simple two-dimensional example where sets F and E are initially the same set of points. Initially, at $t = 0$, the two sets coincide, i.e. $F^{M(0)} = E$. As illustrated in Figure 5, some of the points $x \in F$ will travel outside of E, while others will never leave E. We now define the $\texttt{unsweep}(E, M)$ as the set of all those points x that remain inside E for all values of t. Formally,

$$\texttt{unsweep}(E, M) \equiv \{x \mid x^{M(t)} \in E,\ \forall t \in [0,1]\} \tag{12}$$

By this definition, $\texttt{unsweep}(E, M)$ is the largest subset of F that stays inside E while F is moving according to $M(t)$. The characterization of `unsweep` in definition (12) may not provide an insight in the nature of `unsweep` or possible methods for computing the results of this operation. An equivalent, but more convenient, definition is obtained by observing that the set $x^{M(t)},\ t \in [0,1]$, is simply the trajectory T_x of point x. Therefore,

$$\texttt{unsweep}(E, M) = \{x \mid T_x \subset E\} \tag{13}$$

The condition in equation (13) is illustrated in Figure 5(a). As long as the trajectory T_x of point x stays inside of E, $x \in \texttt{unsweep}(E, M)$, while if T_x intersects the complement set E^c then point $x \notin \texttt{unsweep}(E, M)$. In section 4.1 we will use this observation to develop a rigorous point membership classification (PMC) procedure for $\texttt{unsweep}(E, M)$.

We can also characterize `unsweep` by reversing the roles of the stationary set E and moving points $x \in F$ as illustrated in Figure 5(b). It should be clear from the discussion in section 2.1 that keeping *all* points $x \in F$ fixed, the same *relative* motion between F and E will be described by a set E moving according to the inverted motion $\hat{M}(t)$. At any instant $t = a$, the moving set will occupy points $E^{\hat{M}(a)}$. From these points, only the set $F \cap E^{\hat{M}(a)}$ may remain inside the set F. Since this intersection condition must hold for all values of t, we obtain the third equivalent characterization of `unsweep` as an intersection $\bigcap_{q \in \hat{M}} S^q$ that is identical to the dual of sweep defined by expression (2) in section 1.2.

Finally note that the set of points

$$\texttt{unsweep}(E, \hat{M}) = \bigcap_{q \in M} E^q = \{x \mid \hat{T}_x \subset E\} \tag{14}$$

is *not* the same set as $\texttt{unsweep}(E, M)$. Figure 6(a) shows the $\texttt{unsweep}(E, M)$, i.e. the points of set F are moving in pure translation according to M *or* set E moves according to $\hat{M}$, while Figure 6(b) shows $\texttt{unsweep}(E, \hat{M})$ when the points of set F are moving in pure translation according to $\hat{M}$ *or* set E moves according to M.

3 Set-theoretic Properties

3.1 Duality in the special case of translational motion

The dual relationship between `sweep` and `unsweep` is easier to see in the restricted case when $M(t)$ is a pure translation, that is $\Theta(t)$ in equation (9) is the identity. In this case, each point x of the moving set S sweeps

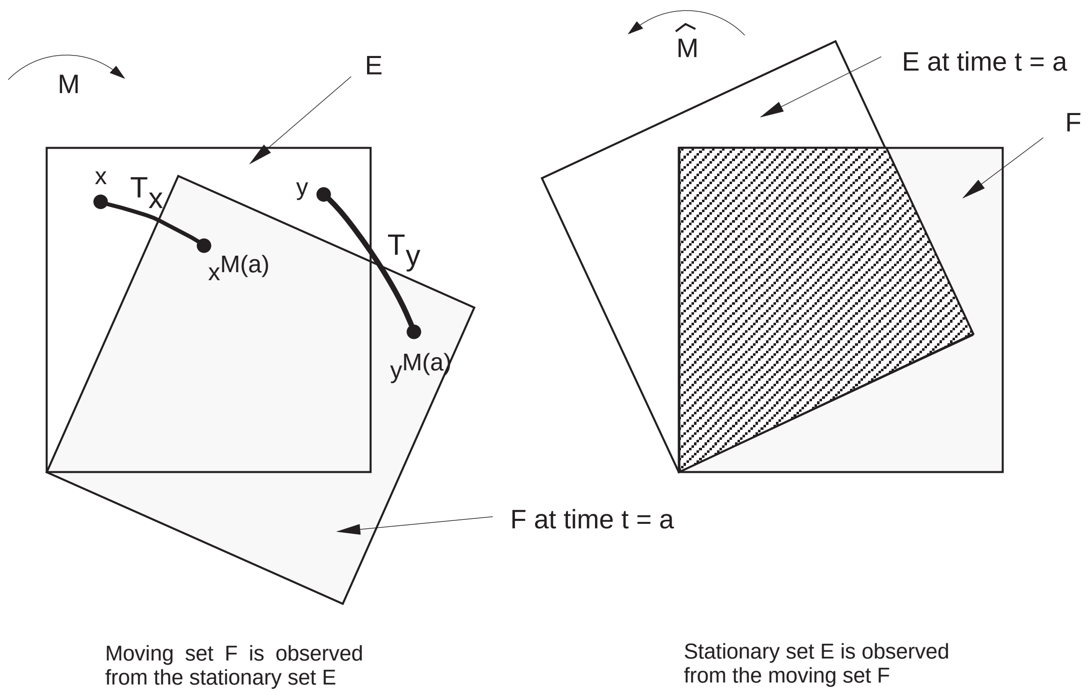

M
E
x
T_x
y
$x^{M(a)}$
T_y
$y^{M(a)}$
F at time t = a
Moving set F is observed from the stationary set E
$\hat{M}$
E at time t = a
F
Stationary set E is observed from the moving set F

Figure 5: Definition of the `unsweep` operation: two sets E and F in relative motion

the trajectory

$$T_x = x + \mathcal{T}(t),\ t \in [0,1]$$

where both x and translational component $\mathcal{T}(t)$ of $M(t)$ have vector values represented with respect to the fixed coordinate system $\mathcal{F}_{\mathcal{W}}$. If $x, y \in S$ are two distinct points with trajectories T_x and T_y, respectively, then it should be clear that

$$T_x = T_y \oplus (x - y)$$

where $\oplus$ is Minkowski (vector set) addition with the usual properties [21]. In other words, all points of S move on congruent trajectories that are determined by $\mathcal{T}(t)$ and differ only by the relative position between the points. This implies that set S moves without changing its orientation and if T is the trajectory of any point of S, but located at the origin of $\mathcal{F}_{\mathcal{W}}$,

$$\texttt{sweep}(S, M) = \bigcup_{q \in M} S^q = S \oplus T \tag{15}$$

is also translation invariant. Similarly, using definition (2) of `unsweep` for the same set S and motion M, we have

$$\texttt{unsweep}(S, M) = \bigcap_{p \in \hat{M}} S^p = S \ominus \hat{T} \tag{16}$$

where $\ominus$ is the usual Minkowski difference operation [21], $\hat{T}$ is the inverted trajectory and in this case it is also the reflection of trajectory T with respect to the origin of the frame $\mathcal{F}_{\mathcal{W}}$. The rightmost expression is sometimes also called *erosion* and is the set of points $\{y \in S \mid T_y \subset S\}$, which is consistent with definition (12) of `unsweep`.

Minkowski operations have many useful properties and are used extensively in geometric modeling, motion planning, and image processing [4, 12, 21, 14, 25]. In particular, definitions of Minkowski operations imply

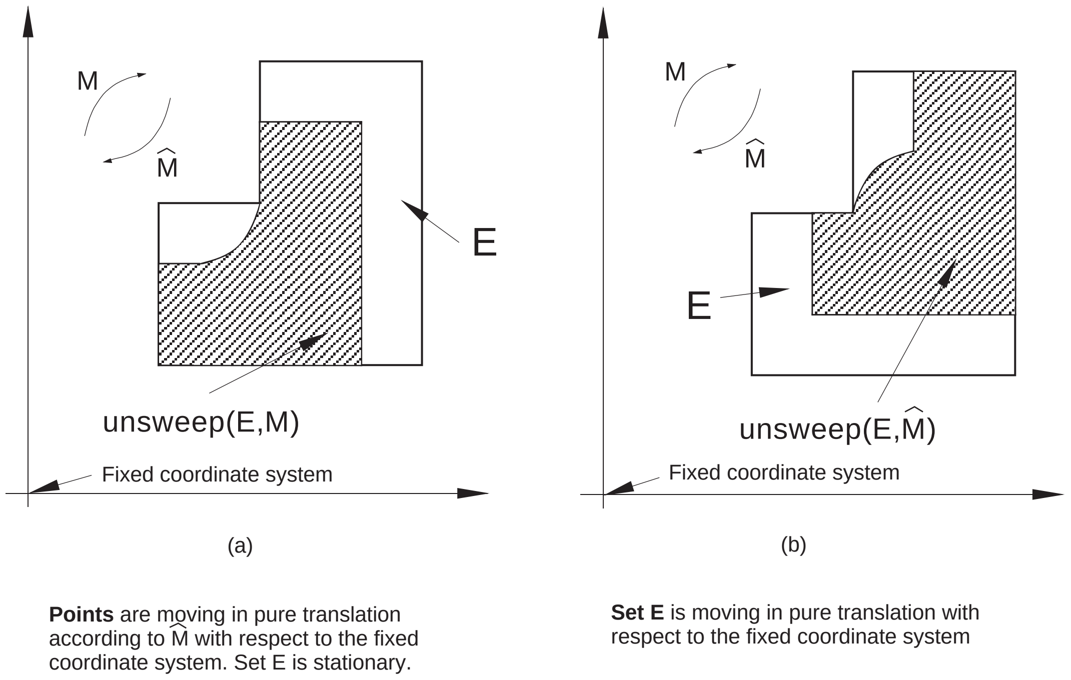

M
$\widehat{M}$
E
unsweep(E,M)
Fixed coordinate system
(a)
Points are moving in pure translation according to $\widehat{M}$ with respect to the fixed coordinate system. Set E is stationary.
M
$\widehat{M}$
E
unsweep(E,$\widehat{M}$)
Fixed coordinate system
(b)
Set E is moving in pure translation with respect to the fixed coordinate system

Figure 6: Difference between the sets $\texttt{unsweep}(E, M)$ and $\texttt{unsweep}(E, \hat{M})$

that $\oplus$ and $\ominus$ are dual to each other via

$$S \oplus T = (S^c \ominus T)^c \tag{17}$$

with X^c denoting the usual complement of set X.

3.2 Duality in the general case

Many properties of Minkowski operations depend on the translation invariance and do not generalize to sweeps and unsweeps under motions other than translations. But a number of Minkowski properties follow strictly from the set-theoretic considerations and definitions. One would expect that all such properties should extend to more general motions. Indeed, the duality of operations $\oplus$ and $\ominus$ is one such property that generalizes to the duality of the operations `sweep` and `unsweep` as defined in this paper.

Let us rewrite the duality (17) using the set theoretic definitions of Minkowski addition and subtraction as

$$\begin{array}{c} \bigcup_{q\in M} S^q = [\bigcap_{q\in M} (S^c)^q]^c, \\ \text{or} \\ (\bigcup_{q\in M} S^q)^c = \bigcap_{q\in M} (S^q)^c \end{array} \tag{18}$$

since $(S^c)^q = (S^q)^c$. Note that the last equality, and hence the duality of $\oplus$ and $\ominus$, is simply a restatement of the usual DeMorgan's law for operations of union and intersection generalized for arbitrary number or families of sets [11]. But this law does not place restrictions on the type of motion and hence it remains valid for general motions M. In this case, the last equality in equation (18) can be rewritten as

$$[\texttt{sweep}(S, M)]^c = \texttt{unsweep}(S^c, \hat{M}) \tag{19}$$

which is equivalent to the relationship (3). The established duality characterizes $\texttt{sweep}(S, M)$ in terms of points moving according to the inverted motion $\hat{M}$ as observed from S^c (or equivalently from S). In other words, sweeping (unsweeping) set S with motion M is equivalent to unsweeping (sweeping) the complement of S with the inverted motion $\hat{M}$ and complementing the result.

The duality between `sweep` and `unsweep` is particularly significant because the two operations are complementary in terms of their set-theoretic properties:

- `unsweep`(S, M) is always a smaller set than S while `sweep`(S, M) is always a larger set, i.e.,

$$\texttt{unsweep}(S, M) \subset S \subset \texttt{sweep}(S, M) \tag{20}$$

- `sweep` preserves connectedness and homogeneity, while `unsweep` does not, as shown in Figure 7;

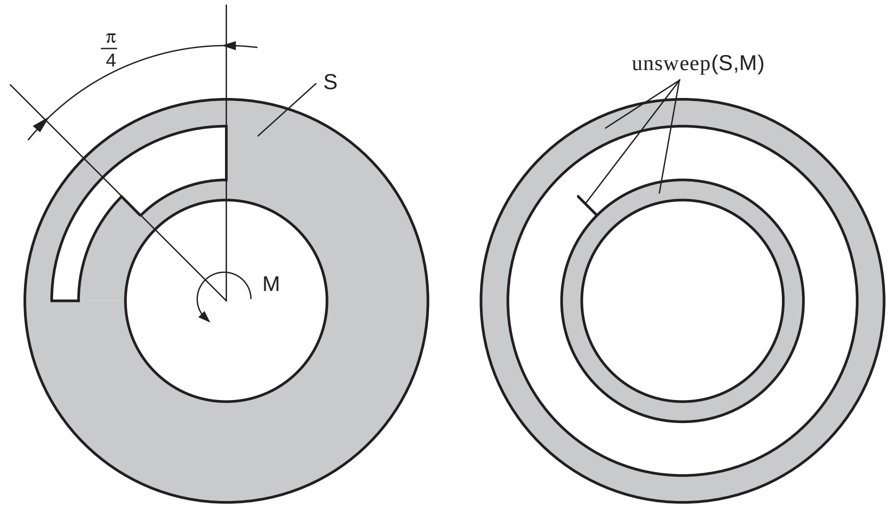


Figure 7: Dimensionally inhomogeneous and disconnected result. Motion M is a $\frac{7\pi}{4}$ rotation around the "center" of the inner circular hole.

- in contrast to `sweep`, `unsweep` (as an infinite intersection) of a convex set remains convex for all motions.

Thus it should be clear that both operations have important practical applications and therefore the relationship between the two operations warrants further investigation.

If x is a point, then

$$x \in \texttt{unsweep}(S, M) \Longleftrightarrow \texttt{sweep}(x, M) \subset S \tag{21}$$

Equation (21) follows directly from the definition (13) of `unsweep` and allows one to decide if a point x belongs to the `unsweep` or not. As discussed later in section 4.1, this test can be extended to include the distinction between points that are "in" or "on" the `unsweep`. A slight generalization of equation (21), which may be useful for deriving a more general Set Membership Classification test[4] for `unsweep`, can be written as

$$Q \subset \texttt{unsweep}(S, M) \Longleftrightarrow \texttt{sweep}(Q, M) \subset S \tag{22}$$

Equation (22) has already been applied in the packaging example shown in Figure 1, and it is a direct consequence of equation (13).

A further consequence of the duality related to equation (22), can be stated as follows: set Q intersects the set swept by S during M if and only if sweeping set Q with the inverted motion $\hat{M}$ "penetrates" S, or

$$Q \bigcap \texttt{sweep}(S, M) \neq \emptyset \Longleftrightarrow \texttt{sweep}(Q, \hat{M}) \bigcap S \neq \emptyset \tag{23}$$

[4] Given a candidate set X and S, Set Membership Classification computes (regularized) portions of X on S, X in S, and X outside of S [26].

Equation (23) may offer additional computational advantages in special situations. For example, if Q is a line, instead of computing the `sweep` of S and then test if the line intersects its boundary, one can test the ruled surface generated by the line Q moving according to $\hat{M}$ against the representation of set S. More generally, if set Q is "simpler" than S, then the equation (23) may eliminate the need to compute the `sweep` of a more complex S.

One way of detecting the collision between two moving objects S and Q is to compute the `sweep` of one of the objects, say S, with the relative motion between them, say M, and to test if $Q \cap \texttt{sweep}(S, M)$ [6]. But equation (23) can also be interpreted as follows: set S collides with set Q during M if and only if Q collides with S during $\hat{M}$. Hence, if one uses the `sweep` to detect the collision between the two moving objects, it may be more efficient to compute the `sweep` of the "simpler" object among Q and S and use equation (23) to produce the result of the collision test.

It may be tempting to think of `unsweep` as the inverse of `sweep`. This is certainly not true, in the sense that $\texttt{unsweep}[\texttt{sweep}(S, M), M]$ is equal to S only in very special situations — just like $(A \oplus B) \ominus B$ is usually not equal to A for Minkowski operations [21]. Formally,

$$\texttt{sweep}[\texttt{unsweep}(S, M), M] \subset S \subset \texttt{unsweep}[\texttt{sweep}(S, M), M] \tag{24}$$

which can be easily proven by using equation (22). In words, it is more severe to `unsweep` first and then `sweep` than to do the reverse. We illustrate this result in Figure 8 in which a square S is moving according

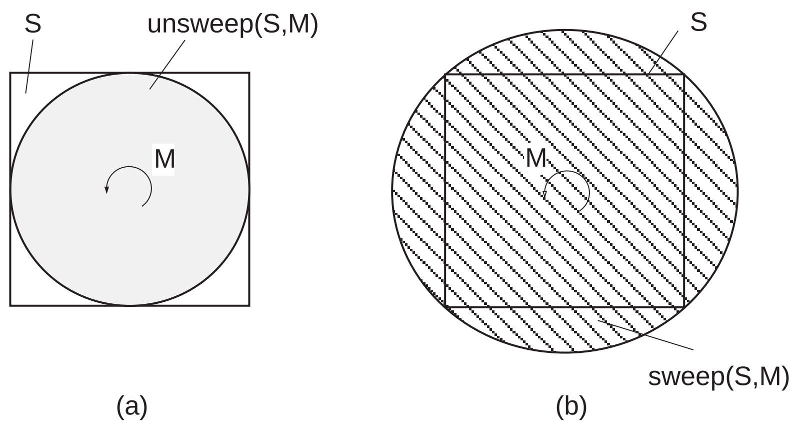


Figure 8: The operation of `unsweep` is not the inverse of `sweep`. Motion M is a 2π counter-clockwise rotation about the "center" of S.

to a motion M which is a 2π counter-clockwise rotation about the "center" of S. If we first `unsweep` S we obtain the set shown in Figure 8(a). Observe that sweeping this set with the same motion M leaves the set $\texttt{unsweep}(S, M)$ unchanged. Similarly, unsweeping the set $\texttt{sweep}(S, M)$ (shown in Figure 8(b)) with M will not modify it. Therefore equation (24) holds even for simple sets and motions.

Properties of Minkowski operations are often used to round corners and edges of objects[5] [4]. Similarly, depending on the type of motion M, combinations of `unsweep` and `sweep` may be used to simplify objects and eliminate details that are not needed in specific applications.

3.3 Multiple Sets

In many cases an object S is given as a Boolean combination of "primitive" sets S_i. For example, to efficiently compute the set $\texttt{unsweep}(S, M)$ when set S is given by its CSG representation with primitives S_i, one needs to relate $\texttt{unsweep}(S, M)$ to the sets $\texttt{unsweep}(S_i, M)$ for various boolean combinations of sets S_i.

[5] $(S \oplus B) \ominus B$ and $(S \ominus B) \oplus B$ – where B is a spherical ball – have the effect of rounding the edges of S.

It is well known that **sweep** distributes over the union, but not over the intersection. It is not surprising that **unsweep** comes with dual properties. If S can be described by an intersection of i arbitrary "primitive" objects S_i that move according to the same motion M, then **unsweep** distributes over intersection and

$$\texttt{unsweep}(\bigcap_i S_i, M) = \bigcap_i \texttt{unsweep}(S_i, M) \tag{25}$$

which is equivalent to

$$\bigcap_{q \in \hat{M}} (\bigcap_i S_i^q) = \bigcap_i (\bigcap_{q \in \hat{M}} S_i^q) \tag{26}$$

and follows from the properties of generalized set operations [11]. For example, Figure 9(b) shows a set S being the intersection between two cylinders S_1 and S_2 shown in Figure 9(a). Motion M (not shown) is a 1

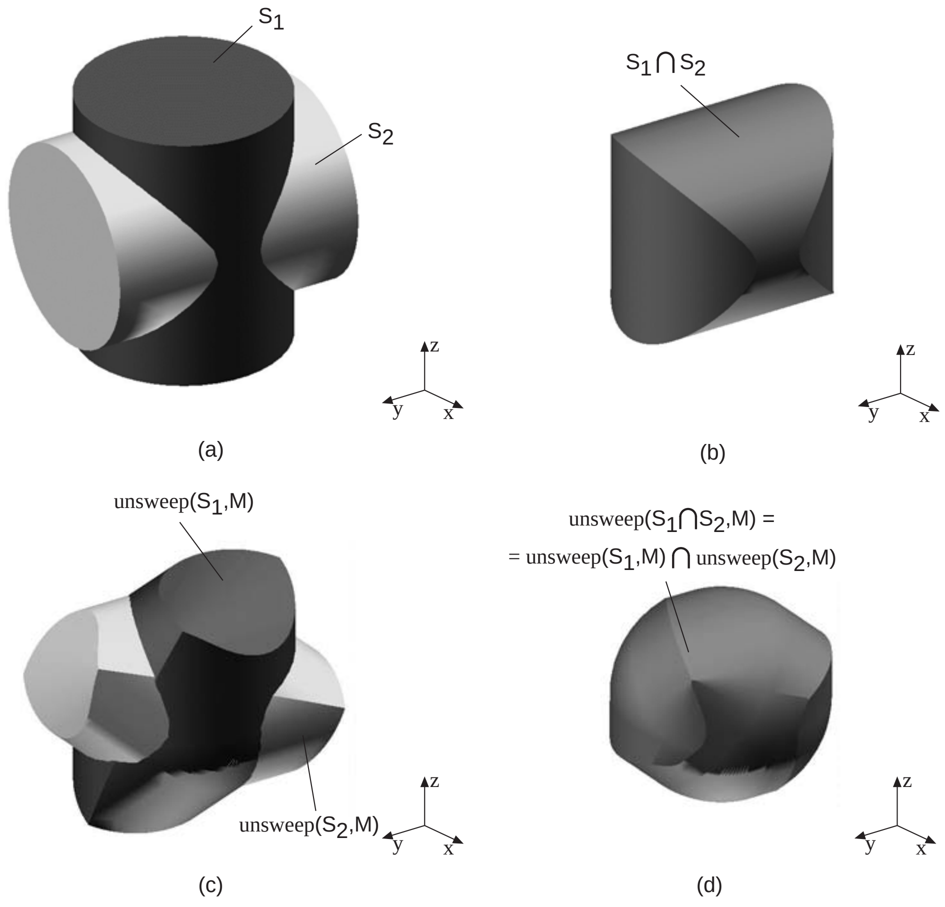


Figure 9: The set S, obtained by intersecting two cylinders whose axes intersect (a), may have a complex shape (b). The set $\texttt{unsweep}(S, M)$ (d) could be obtained by either computing $\texttt{unsweep}(S_1, M) \bigcap \texttt{unsweep}(S_1, M)$ (c) or $\texttt{unsweep}(S_1 \bigcap S_2, M)$ directly

radian counter-clockwise rotation around an axis aligned with the direction (1 1 1) which passes through the point of intersection of the axes of the two cylinders. Instead of computing $\texttt{unsweep}(S, M)$, one can compute **unsweep** for the two "primitive" cylinders and the given motion, as shown in Figure 9(c), and then simply take the boolean intersection of the results. Figure 9(d) shows the result $\texttt{unsweep}(S_1, M) \bigcap \texttt{unsweep}(S_1, M)$ which, according to equation (9), could also be obtained by computing $\texttt{unsweep}(S_1 \bigcap S_2, M)$ directly. Depending on the representation of S, computing $\bigcap_i \texttt{unsweep}(S_i, M)$ may be more convenient or efficient than the direct computation of $\texttt{unsweep}(S, M)$.

Unfortunately, `unsweep` does not distribute over the union, except in some very special situations. When set S can be described by a boolean union of "primitive" sets as $S = \bigcup_i S_i$, we have

$$S \supset \texttt{unsweep}(S, M) \supset \bigcup_i \texttt{unsweep}(S_i, M) \tag{27}$$

since $S_i \subset S$ implies that $\texttt{unsweep}(S_i, M) \subset \texttt{unsweep}(S, M)$ for every i. Equation (27) is illustrated in Figure

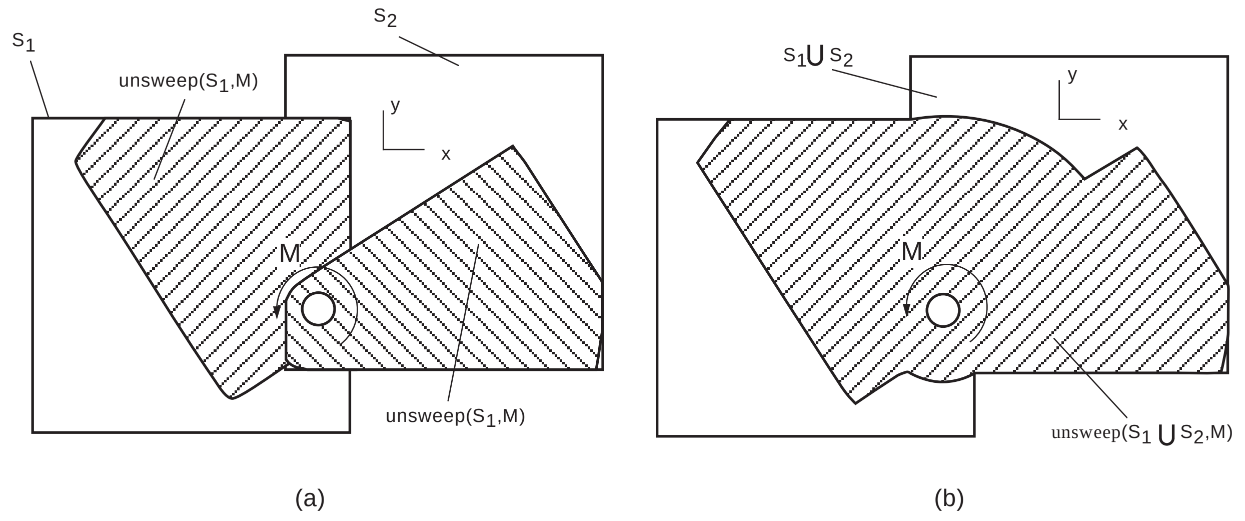


Figure 10: When set S is given as a union of i primitive sets S_i, the set $\texttt{unsweep}(S, M)$ includes $\bigcup_i \texttt{unsweep}(S_i, M)$.

10: two moving blocks S_1 and S_2, shown in Figure 10(a), are both moving according to a motion M which is a 1-radian counter-clockwise rotation around the z - axis that passes through the center of the indicated hole. The computed sets $\texttt{unsweep}(S_1, M)$ and $\texttt{unsweep}(S_2, M)$ remain inside S_1 and S_2 respectively. If we compute the set $\texttt{unsweep}(S_1 \bigcup S_2, M)$, we obtain the set shown in Figure 10(b), and the relationship given in equation (27) becomes obvious even in this simple case. Note that in equation (27) sets S_i do not have to be disjoint; if the sets S_i are disjoint *connected components* of S, then it is equivalent either to `unsweep` S considered as a whole, or to `unsweep` each S_i separately, and take the union of the results. Thus, in this restricted case equation (27) becomes:

$$\texttt{unsweep}(S, M) = \bigcup_i (\texttt{unsweep}(S_i, M)) \tag{28}$$

To see this, consider a point $x \in \texttt{unsweep}(S, M)$ which implies that $T_x \subset S$. But S contains disjoint connected components S_i, hence $T_x \subset S_i$ and by definition $x \in \texttt{unsweep}(S_i, M)$. Therefore $x \in \bigcup_i \texttt{unsweep}(S_i, M)$ and $\texttt{unsweep}(S, M) \subset \bigcup_i \texttt{unsweep}(S_i, M)$. The rest of the proof follows from equation (27). Of course, if equation (28) is satisfied, it *does not* imply that the sets S_i are disconnected.

3.4 Composite Transformations

In many practical situations, set S can move according to a motion M that is best described by a sequence of motions M_i, $i = 1, ..., n$. We say that motions M_i are applied in *sequence*, or equivalently M is a *sequential* motion and denote it by M^s. The sequential motion M^s is a composition of the primitive motions in the sequence:

$$M^s(t) = M_n(t)[M_{n-1}(t)[...[M_1(t)]]] \tag{29}$$

Observe that the order in which the motions M_i are applied[6] to S is important and that, informally, the motions M_i are applied "one after another". If all motions are represented as discussed in section 2.2, the composition becomes a sequence of matrix multiplication operations.

Consider a simple example in Figure 11 where S has to move according to a simple sequence of two motions. Then the sets $\texttt{unsweep}(S, M_1)$ and $\texttt{unsweep}(S, M_2)$ remain inside S during M_1 only and M_2 only, as shown in Figure 11(a) and (b) respectively. Now, if one simply intersects these two sets that correspond to each of the two motions, one would obtain the set shown in Figure 11(c) which will remain, by definition, inside S during M_1 and M_2. But this set $\texttt{unsweep}(S, M_1) \cap \texttt{unsweep}(S, M_2)$ is neither the *largest* set that remains inside S during M^s, and hence it does not equal $\texttt{unsweep}(S, M^s)$, nor a subset of $\texttt{unsweep}(S, M^s)$ in general. To obtain $\texttt{unsweep}(S, M^s)$ one has to intersect the set $\texttt{unsweep}(S, M_1)$ at its final position during M_1, with $\texttt{unsweep}(S, M_2)$, and the result of this intersection has to be "moved" back to its initial position. Therefore, for a sequential motion $M^s = M_2[M_1]$ and $[0,1] = [0, a_1] \cup [a_1, 1]$, with a_1 given, we have

$$\texttt{unsweep}(S, M^s) = \hat{M}_1(a_1)\{\texttt{unsweep}[M_1(a_1)\texttt{unsweep}(S, M_1), M_2]\} \tag{30}$$

which is the largest set that will remain inside S during the sequential motion M^s. Equation (30) can be generalized for a sequence of n motions as

$$\begin{aligned}
\texttt{unsweep}(S, M^s) = \hat{M}_1(a_1)\ldots\hat{M}_{n-1}(a_{n-1})[\\
\quad \texttt{unsweep}(M_{n-1}(a_{n-1}) \\
\quad\quad \texttt{unsweep}(M_{n-2}(a_{n-2}) \\
\vdots \\
\quad\quad\quad \texttt{unsweep}(M_1(a_1) \\
\quad\quad\quad\quad \texttt{unsweep}(S, M_1), M_2), \ldots, M_{n-1}), M_n)]
\end{aligned} \tag{31}$$

Note that equations (30) and (31) contain terms of the form $T[\texttt{unsweep}(S, M)]$, where T is an invertible affine transformation. Recall that $\texttt{unsweep}(S, M) = \cap_{q \in \hat{M}} S^q$ and look at the transformation T applied to one instance of S^q. We obviously have

$$T[q(S)] = T\{q[T^{-1}T(S)]\} = (TqT^{-1})[T(S)]$$

Applying T to every instance of q yields

$$T[\texttt{unsweep}(S, M)] = \texttt{unsweep}(T(S), TMT^{-1}) \tag{32}$$

and

$$\texttt{unsweep}(T(S), M) = T[\texttt{unsweep}(S, T^{-1}MT)] \tag{33}$$

Equations (32) and (33) show how to apply affine transformations to sets and their unsweeps. They are useful in many situations where transformations are inherently part of the representation scheme, e.g, in CSG trees or sequential motions we described above.

Finally, for arbitrary motions M_i we have

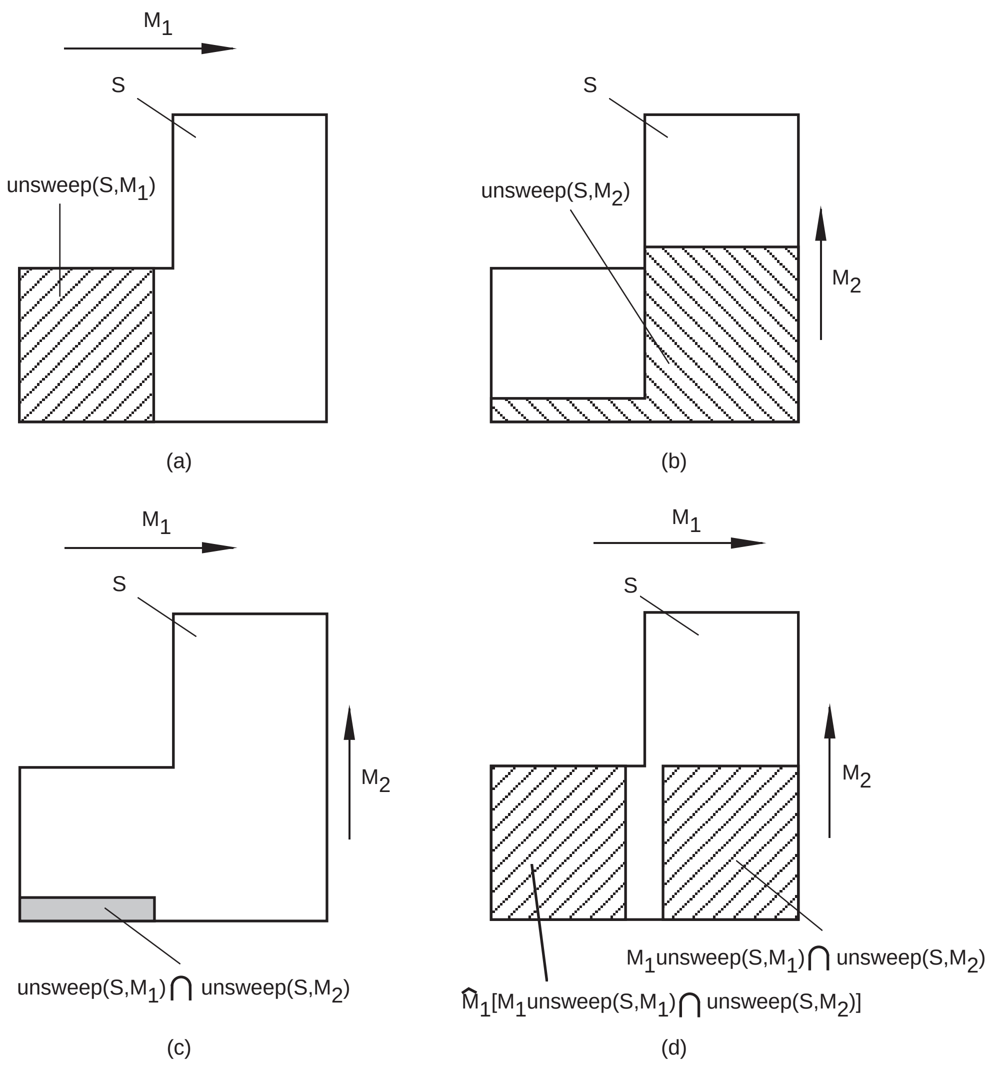
$$S \supset \bigcup_i \texttt{unsweep}(S, M_i) \supset \bigcap_i \texttt{unsweep}(S, M_i) \tag{34}$$

which follows directly from the definitions of the $\texttt{unsweep}$ operation since $\texttt{unsweep}(S, M)$ is defined as the largest subset of S that remains inside the original S during M. In other words, for every motion M_i, each $\texttt{unsweep}(S, M_i)$ corresponds to a material removal operation.

[6] To be more precise, the motion M^s defined by equation (29) is a sequence if motions M_i are defined by

$$M_i(t) = \begin{cases} I, & t < a_{i-1} \\ M_i(t), & t \in [a_{i-1}, a_i] \\ M_i(a_i), & t > a_i \end{cases}$$

where $t \in [0,1]$, the interval $[0,1]$ is partitioned in n sub-intervals by $[0,1] = [0, a_1] \cup [a_1, a_2] \cup \ldots \cup [a_{n-1}, 1]$ with a_i fixed, and I is the identity matrix.



M1
S
unsweep(S,M1)
(a)
S
unsweep(S,M2)
M2
(b)
M1
S
M2
unsweep(S,M1) ⋂ unsweep(S,M2)
(c)
M1
S
M2
M1unsweep(S,M1) ⋂ unsweep(S,M2)
M̂1[M1unsweep(S,M1) ⋂ unsweep(S,M2)]
(d)

Figure 11: A sequential motion $M = M_2[M_1]$ applied to set S. Simply intersecting the unsweeps that correspond to each of the two motions (c) would not result in the largest set that remains inside S during that sequence of motions (d).

4 Computational Issues

4.1 PMC for `unsweep` and `sweep`

Point Membership Classification (PMC) is a procedure for deciding whether a given point x is inside, outside, or on the boundary of a set S [26]. It has its roots in solid modeling, and set S is usually a d-dimensional solid ($d = 1, 2, 3$). The PMC procedure is a sign of 'informational completeness' of a representation scheme [20], indicating that any geometric property can be computed at least in principle. As a matter of practical importance, PMC is used in almost all geometric modeling algorithms, including boundary evaluation, discretization, and rendering. PMC is a special case of the Set Membership Classification function that, given a candidate set X and S computes (regularized) portions of X on S, X in S, and X outside of S [26].

We make no assumptions on whether S is regular or not. The operations of `sweep` and `unsweep`, as defined in this paper, may result in sets that are not regular, i.e. dimensionally inhomogeneous sets with "dangling" faces, edges or isolated points. Closed regular sets are not closed under Minkowski difference (see example in [18]), and in section 3 is shown an example showing that `unsweep` of a regular set S is not regular. Regularity, other topological, and set-theoretic properties are outside of the scope of this paper. Instead we show how to use the definitions of `unsweep` to obtain a PMC procedure for the general `unsweep` (and then general `sweep`). Accordingly, we generalize the standard PMC notions in a manner consistent with definitions in [26]. We will say that point x is 'in' set S when there is an open neighborhood of x contained in S; point x is 'out' of S when some open neighborhood of x is contained in the complement set S^c; and x is 'on' S if every neighborhood of x intersects both S and S^c. The last condition implies that $x \in \partial S$ is a boundary point; but x may or may not belong to S, because set S could be open, closed, or neither.

The basis for a sound PMC procedure is supplied by equations (12)-(13), and an appropriate interpretation of the moving point trajectory T_x. The PMC definitions imply that we need to consider not only a moving point x, but also its open neighborhood ball $B(x^{M(t)})$ of points that is transformed with x. To perform PMC on $\texttt{unsweep}(E, M)$ we need to consider three situations:

1. If trajectory T_x remains in the interior $\mathbf{i}S$ of S, then point $x^{M(t)}$ remains inside S for all values of t during the motion $M(t)$ that generated T_x. Furthermore, there is an open neighborhood $B(x^{M(t)})$ of points that also remain in the interior $\mathbf{i}S$ of S for all t (see Figure 12 (a)). It follows that in this case x is 'in' $\texttt{unsweep}(S, M)$.

2. Whenever T_x intersects the boundary ∂S, the neighborhood $B(x^{M(t)})$ gets 'trimmed' by ∂S, as illustrated in Figure 12 (b). This implies that no open neighborhood of x is contained in $\texttt{unsweep}(S, M)$, and therefore x cannot be 'in'; thus, x must be either 'out' or 'on' $\texttt{unsweep}(S, M)$.

3. Additional neighborhood analysis to distinguish between 'on' and 'out' cases depends on properties of set S (open, closed, dimension) and trajectory T_x (whether it intersects ∂S transversally, whether it remains inside S, etc.) In all cases, the analysis amounts to determining whether the neighborhood $B(x^{M(t)})$ of the moving point gets 'trimmed' down to the empty set $\emptyset$. If so, the point x must be 'out' of $\texttt{unsweep}(S, M)$. Otherwise, x remains 'on.'

Let us consider a common case when S is any closed set, for example a solid. As long as the trajectory T_x does not 'go' outside of S, point x cannot classify 'out'. Therefore, the PMC procedure reduces to the straightforward classification of a curve T_x against a representation for set S:

$$PMC[x, \texttt{unsweep}(S, M)] = \begin{cases} in, & T_x \subset \mathbf{i}S \\ out, & T_x \cap S^c \neq \emptyset \\ on, & \text{otherwise} \end{cases} \tag{35}$$

and it remains correct irrespective of homogeneity or dimension of S — assuming that boundary, interior, and complement are all defined relative to the same universal set.

By duality, we can use the same technique to construct a PMC procedure for any $\texttt{sweep}(S, M)$. Since

$$\partial[\texttt{sweep}(S, M)] = \partial[\texttt{unsweep}(S^c, \hat{M})] \tag{36}$$

it is sufficient to classify point x against $\texttt{unsweep}(S^c, \hat{M})$ and simply exchange 'in' and 'out' results. Notice however that whenever S and $\texttt{sweep}(S, M)$ are closed sets, set S^c and $\texttt{unsweep}(S^c, \hat{M})$ are open, and the PMC conditions are slightly different from (35) above. Since the boundary ∂Y of an open set Y is a subset of Y^c, the condition $T_x \bigcap Y^c \neq \emptyset$ may imply that point x is either 'on' or 'out' of Y. In this situation, additional neighborhood analysis may be required to distinguish between the two cases. This complication does not arise in solid modeling applications when all sets are regularized.

Finally note that, when S is a planar cross-section moving in E^3, this approach for PMC on $\texttt{sweep}(S, M)$ yields the procedure proposed in [7].

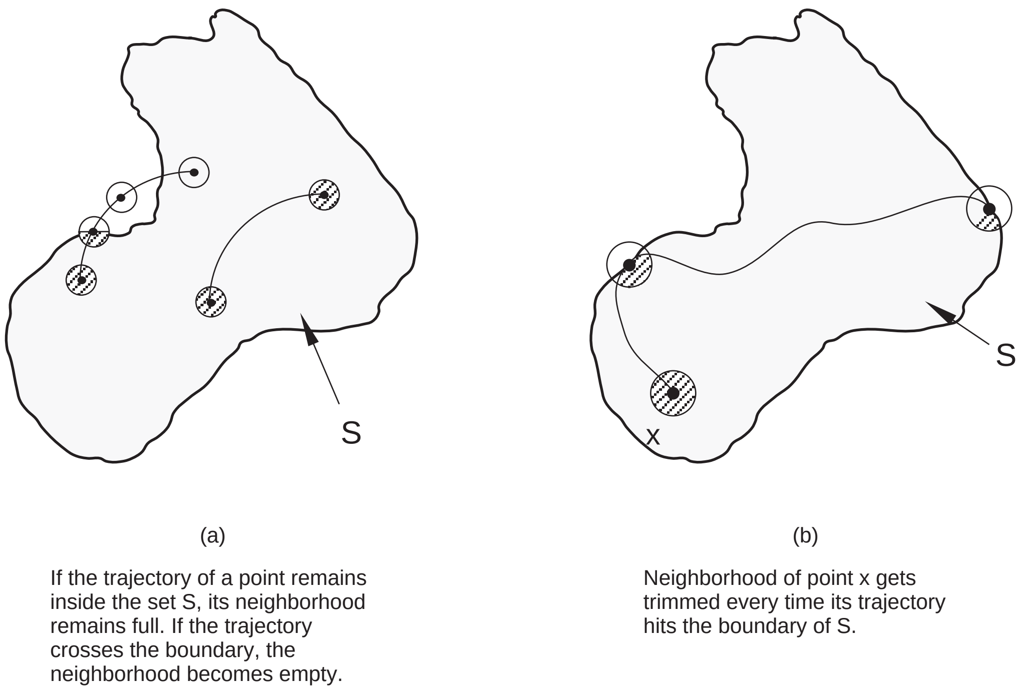

S
(a)
If the trajectory of a point remains inside the set S, its neighborhood remains full. If the trajectory crosses the boundary, the neighborhood becomes empty.
S
x
(b)
Neighborhood of point x gets trimmed every time its trajectory hits the boundary of S.

Figure 12: Neighborhoods of points moving on trajectories generated by motion M

4.2 Implementation using the Point Membership Classification test

The formulation defining $\texttt{unsweep}(S, M)$ in terms of trajectories of moving points (equation (13)) leads to a well-defined PMC procedure as described above. Ability to perform PMC can be used for computing $\texttt{unsweep}(S, M)$ either exactly (within the machine precision) or approximately. For example, the steps in computing the exact boundary representation of `unsweep` are similar to the usual procedure for boundary evaluation [19]:

1. generating surfaces bounding the `unsweep`;
2. intersecting the generated surfaces to produce a set of potential candidate faces;
3. testing which of the candidate faces lie on the boundary of `unsweep`.

The shared boundary of `sweep` and its dual `unsweep` (equation (36)) implies that all the methods used in generating bounding surfaces for computing `sweep` ([15, 1, 24]) are also applicable to computing `unsweep`. The degree of difficulty of the second step clearly depends on the types of surfaces generated in the first step. The third step amounts to selecting a representative point in each candidate face and testing it against $\texttt{unsweep}(S, M)$ using the PMC procedure, as described above in section 4.1.

The PMC procedure also opens the doors to the standard approximation methods based on various cell decompositions, such as piecewise linear tesselations, octrees, and marching-cube algorithms [5]. As an example, a recursive algorithm that computes `unsweep` by using an octree decomposition of space was implemented. The trajectory of every corner of every cell was tested against the set S according to the PMC test described in section 4.1, and the results of these tests were used to decide whether the cell was *in*, *out* or *partially in* the `unsweep`. Depending on the geometry of S, it may not be enough to test only the corners of every cell; other representative points from every cell may have to be tested. Figure 13(b) shows the set `unsweep`(S, M) approximated by a computed octree decomposition. The moving object S, is the union of a cylinder and a cube (Figure 13(a)), and motion M is a translation along a trajectory whose projections (on the three coordinate planes) are shown in (c), (d) and (e).

4.3 Implementation using discrete intersection

Definition (2) gives a method for approximating `unsweep`(S, M) by a finite intersection of sets $S^{\hat{M}(t)}$ positioned at discrete time intervals according to the inverted motion $\hat{M}$. Intuitively, at every time step $t = a$, the "unwanted" portion of $S^{\hat{M}(a)}$ that protrudes outside of $S^{\hat{M}(0)}$ is eliminated through the intersection operation. This method is easy to implement in any system that supports the desired transformations (e.g. rigid body motions) and Boolean set operations. When S is a solid, good performance and quality of the approximations can be obtained by using raycasting software and hardware techniques, as described in [17, 16]. All examples of `unsweep`(E, M) presented in the previous sections of this paper were created by approximating `unsweep` as a discrete intersection following a straightforward algorithm. These restricted situations, which involve only "simple" containing sets and translations or rotations illustrate that the results of `unsweep` are not always intuitive and would be difficult to obtain manually.

Other three-dimensional examples of `unsweep` are shown in Figure 14. In Figure 14(a) the containing set E is a cylinder and the motion M is a rotation around the axis aligned with the center of the shown hole (parallel to the z-axis and perpendicular to the axis of the cylinder). The computed set `unsweep`(E, M) is shown inside the cylinder and represents the largest subset of E which remains inside E while rotating around the rotation axis in the counterclockwise direction. Figure 14(b) shows another example where the containing set E is the union of a cylinder and a sphere and M is inverted from a 'helical' motion $\hat{M}$ that is given by:

$$\theta(t) = t, \quad x(t) = R\cos t, \quad y(t) = R\sin t, \quad z = 10t,$$

where θ specifies the rotation around z-axis. In Figure 14(c), E is a cylinder and M is a sequence of two rotations: first 1 radian rotation around the x-axis (1 0 0), followed by another 1 radian rotation around the axis aligned with (0 1 1). The two axes are positioned so that they intersect the axis of the cylinder at the same point (not shown). It is easy to see that, as the containing set and motions become more complex, predicting the shape of `unsweep`(E, M) becomes very difficult (if not impossible) without proper computational support. This may explain why `unsweep` has not been formulated or used until now. Recall that the `unsweep` operation preserves convexity of the containing set; thus, if E is convex, then `unsweep`(E, M) will also be convex as in Figures 14(a) and 14(c). Figure 14(b) shows that `unsweep`(E, M) does not have to be convex, but, intuitively, it will never be "any less convex" than the generator set E.

The final example in Figure 14(d) shows a translational sweep of a simple solid S. The solid is constructed as the union of a cube and a cylinder, and the motion is a translation in the direction of vector (1 1 1). The shown sweep was actually computed as `unsweep` of the complemented set, using the duality equation (19). For practical purposes, the role of the universal set is played by a bounding box that is large enough to contain the sweep. Then the (relative) complement of S is simply the set difference of the bounding box and S.

By duality, all of the above methods are also applicable to `sweep`. Specific representational choices and computational strategies will depend on properties of S and M. For example, it makes little sense to approximate `sweep`(S, M) by discrete union or intersection when S is a planar cross-section moving in E^3, and it may not be feasible to compute the exact boundary representation when S is a solid bounded by parametric surfaces of high degree.

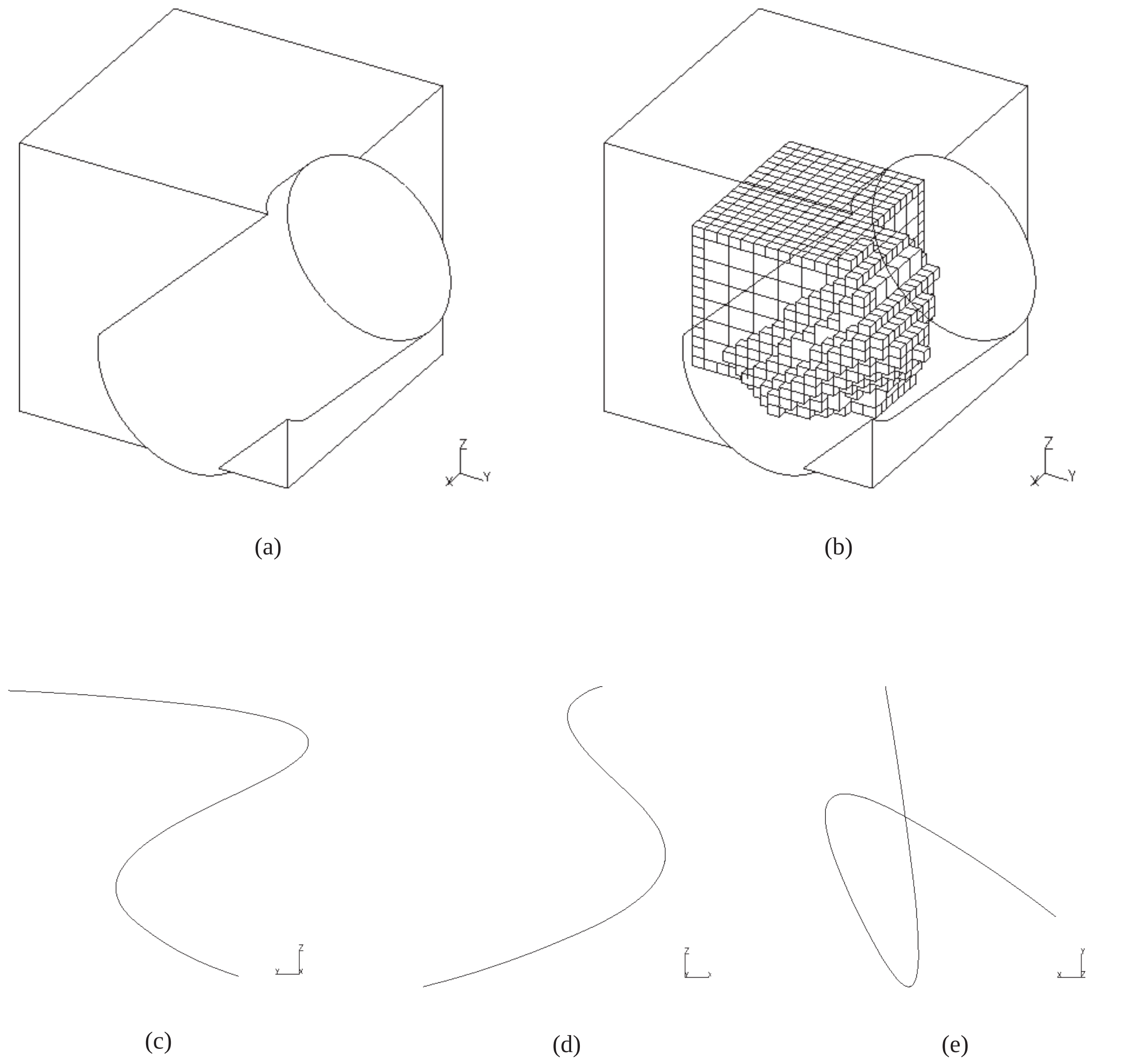


Figure 13: Using an octree decomposition of space coupled with the PMC test to compute the set **unsweep**(S, M). Motion M is a translation along the trajectory whose projections along the coordinate axes are shown.

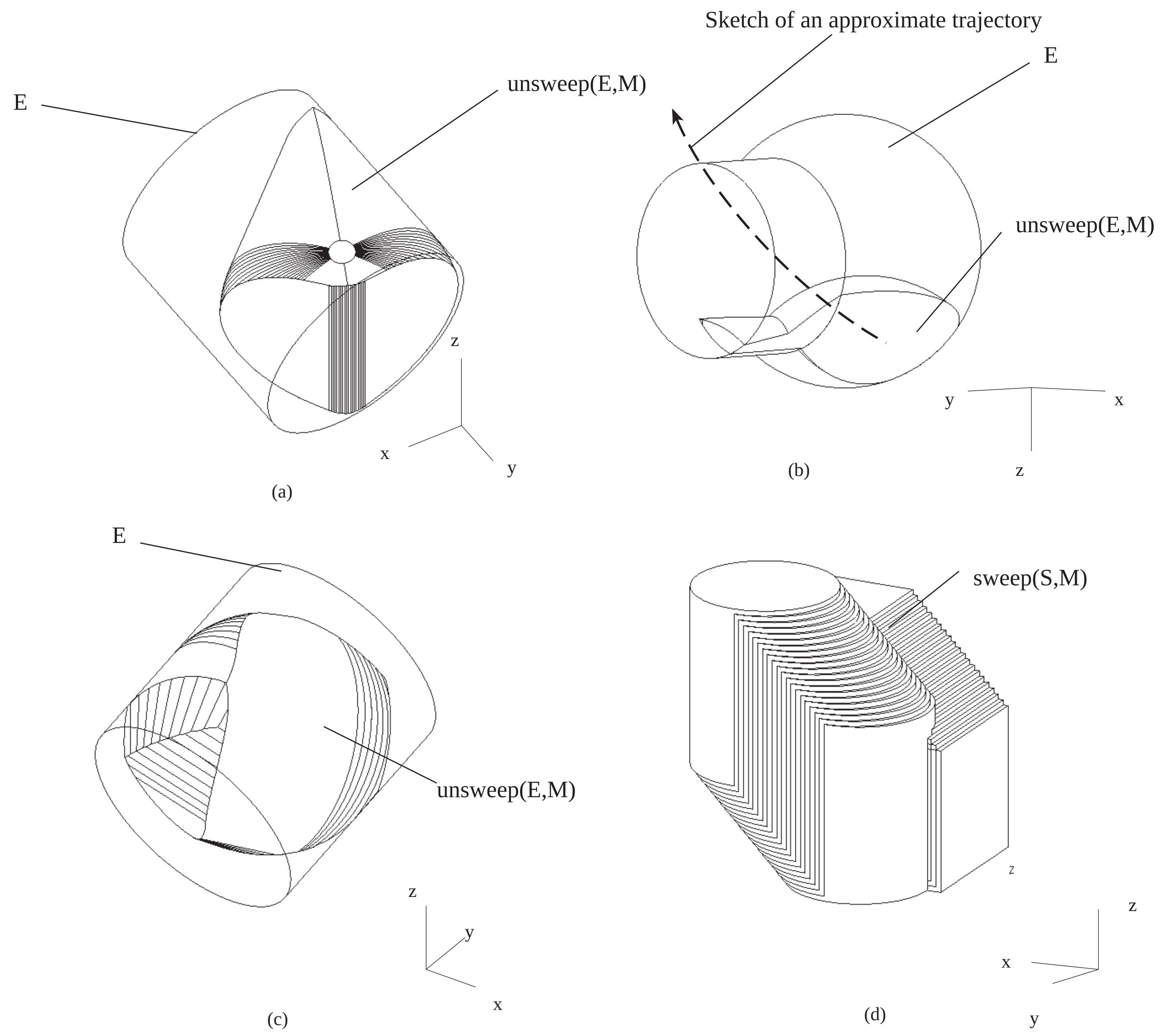


Figure 14: Examples of `unsweep` applications: (a) E is a cylinder and M is a rotation around the axis aligned with the center of the shown hole; (b) E is the union of a cylinder and a sphere and M is inverted from a 'helical' motion $\hat{M}$; (c) E is a cylinder and M is a sequence of two rotations; (d) a translational sweep of a simple solid S: the solid is constructed as the union of a cube and a cylinder.

5 Conclusions

5.1 Sweep or unsweep?

While the notion of 'inverse sweep' seems to arise naturally in some applications[22], it does not appear to be well-defined in general. As the *dual* of `sweep`, the operation of `unsweep` does appear to solve some 'inverse' practical problems, such as design problems described in section 1.3. It is quite unusual that the 'inverse' problem appears to be conceptually easier than the usual 'direct' problem of sweeping a moving part. Why should it be easier to perform PMC on `unsweep` than on `sweep`, even though sweeping usually appears to be more natural than unsweeping? Compare the informal descriptions of the two operations:

`sweep`: the set of points occupied by object S at *some* time during its motion; and

`unsweep`: the set of points that remain inside set S at *all* times during their motion.

The above characterization of `sweep` may be more natural in certain applications, and it suggests methods for generating the surfaces swept by boundaries of S as it moves; but it intrinsically suggests a *search*. By contrast, the description of `unsweep` naturally lends itself to an intersection or containment test, but it does not refer to a *given* moving object or its boundaries. The duality between the two operations says that the two characterizations are equivalent in the following sense: a stationary point x belongs to the `sweep` of moving S if and only if the trajectory of x as seen from S penetrates S at its initial position. Apparently, *for the same relative motion*, the choice of which object is moving and which is stationary can make a world of difference from the computational point of view.

It is well known that all Boolean set functions can be constructed without using set intersection $\cap$, since $A \cap B = (A^c \cup B^c)^c$. Yet, all three operations are needed to take full advantage of the rich Boolean algebraic structure. Both union *and* intersection are needed in most applications, and their duality is *not* viewed as an indication of redundancy. Analogously, the duality between `sweep` and `unsweep` is not so much indication of their equivalence, but a hint of a rich and unexplored algebraic structure. Without a doubt, use of `unsweep` appears to be more natural than that of `sweep` in many applications. For example, the set `unsweep`(E, M) in Figure 1(c) can obviously be rewritten, although in a very unintuitive form, as $[\texttt{sweep}(E^c, \hat{M})]^c$, but it is not likely to happen without explicit recognition of `unsweep` and its properties. We expect that the *combined* properties of `sweep` and `unsweep` should lead to new theoretical results, as well as other algorithms and applications for shape design and analysis of objects in relative motion.

5.2 Significance and Extensions

For the most part, we avoided making any assumptions about set S and motion M. This implies that our results are general and are widely applicable. The research described in this paper advances the field of geometric modeling and applications in at least three distinct ways:

- As a concept, `unsweep` is a new tool for creating and modifying geometric shapes. In addition to the example applications already mentioned in section 1.3, we anticipate many other applications in manufacturing planning, simulation, mechanical design, and analysis of moving parts and assemblies. For example `unsweep` can be applied to the design of cam-follower mechanisms, gears, and other objects in conjugate or relative motion which are traditionally modelled by using the theory of envelopes [27, 13].

- We have shown that `unsweep` corresponds to a material removal operation, and that it comes with attractive theoretical and computational properties, including a relatively straightforward PMC procedure and natural computing strategies, both exact and approximate;

- Finally, `unsweep` fills in a missing link in the theory and practice of geometric modeling. As a dual of `sweep`, it complements the theory of sweeps, provides the previously unavailable computational support, and strengthens the formal properties of sweeps as a representation scheme.

The generality of our approach also means that a number of specific issues have not been discussed and are yet to be addressed. These include other set-theoretic and topological properties of `unsweep` relevant to particular applications, detailed PMC procedures for specific sets (e.g. curves, planar, solid, open) and motions (translations, rotations, general rigid, with deformation, etc.), regularization, and others.

In the spirit of [20], we attempted to keep the discussion 'representation-free.' Representational choices for sets, motions, and trajectories are many and are clearly important; the specific choices may depend on the relative importance of simplicity, efficiency and compatibility with existing systems as well as other pragmatic considerations. This work is intended to provide the starting point for such explorations.

Acknowledgments

This work is supported in part by the National Science Foundation grants DMI-9502728 and DMI-9522806. All three-dimensional examples were created using Parasolid modeling system, courtesy of EDS Unigraphics. An extended abstract of this paper appeared in [9]. The authors are grateful to Dr. Jarek Rossignac for his insightful comments and a stimulating discussion, and to Dr. Myung-Soo Kim and anonymous reviewers for suggesting a number of improvements to the paper.

References

[1] D. Blackmore and M.C. Leu. A differential equation approach to swept volumes. *Proceedings of the 2nd International Conference on Computer Integrated Manufacturing*, pages 143–149, 1990.

[2] X. Chen and C. H. Hoffmann. Towards feature attachment. Technical Report CSD-TR-94-010, Department of Computer Science, Purdue University, West Lafayette, IN, February 1994.

[3] J.K Davidson and K.H. Hunt. Robot workspace of a tool plane: Part1 - a ruled surface and other geometry. *Journal of Mechanisms, Transmissions and Automation in Design*, 109:50–60, 1987.

[4] G. Evans, R.C. Koppelman and V.T. Rajan. Shaping geometric objects by cumulative translational sweeps. *IBM J. Res. Develop.*, 31(3):343–360, 1987.

[5] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes. *Computer Graphics: Principles and Practice.* Addison-Wesley, 1990.

[6] M. A. Ganter. *Dynamic Collision Detection Using Kinematics and Solid Modeling Techniques.* PhD thesis, University of Wisconsin-Madison, 1985.

[7] K.C. Hui. Solid modelling with sweep-CSG representation. *Proceedings of CSG 94 Set Theoretic Solid Modelling; Techniques and Applications*, pages 119–131, 1994.

[8] H. Ilies and V. Shapiro. An approach to systematic part design. In *Proceedings of the 5th IFIP WG5.2 Workshop on Geometric Modeling in CAD*, pages 383–392, may 1996.

[9] H. Ilies and V. Shapiro. `UNSWEEP`: Formulation and computational properties. In *Proceedings of the fourth ACM Symposium on Solid Modeling and applications: Solid Modeling '97*, Atlanta, Georgia, May 1997.

[10] B. Jüttler and M.G. Wagner. Computer-aided design with spatial rational B-spline motions. *Journal of Mechanical Design*, 118:193 – 201, June 1996.

[11] K. Kuratowski and A. Mostowski. *Set Theory.* North Holland, Amsterdam/New York/Oxford, 1976. Studies in Logic and Foundations of Mathematics, Vol 86.

[12] Jean-Claude Latombe. *Robot Motion Planning.* Kluwer Academic Publishers, Boston / Dordrecht / London, 1991.

[13] F. L. Litvin. *Gear Geometry and Applied Theory.* Prentice-Hall, 1994.

[14] R. R. Martin and P. C. Stephenson. Putting objects into boxes. *Computer Aided Design*, 20(9):506–514, 1988.

[15] R.R. Martin and P.C. Stephenson. Sweeping of three dimensional objects. *Computer Aided Design*, 22(4):223–233, 1990.

[16] J. Menon, R. J. Marisa, and J. Zagajac. More powerful solid modeling through ray representations. *IEEE Computer Graphics and Applications*, 14(3), May 1994.

[17] Jai P. Menon and Dean M. Robinson. Advanced NC verification via massively parallel raycasting. *Manufacturing Review*, 6(2):141–154, 1993.

[18] J.P. Menon and H. B. Voelcker. Set theoretic properties of ray representations and Minkowski operations on solids. Technical Report CPA91-9, Cornell Programmable Automation, Cornell University, Ithaca, NY, January 1992.

[19] A. A. G. Requicha and H. B. Voelcker. Boolean operations in solid modeling: Boundary evaluation and merging algorithms. *Proceedings of the IEEE*, 73(1):30–44, January 1985.

[20] A.A.G. Requicha. Representations for rigid solids: Theory, methods and systems. *Computing Surveys*, 12(4):437–463, 1980.

[21] Jean Serra. *Image Analysis and Mathematical Morphology.* Academic Press, London, 1982. Volume 1.

[22] A. Shirur and J. J. Shah. Machining algebra for mapping volumes to machining operations. In *ASME Design Automation Conference*, Irvine, CA, August 1996.

[23] John M. Snyder and James T. Kajiya. Generative modeling: A symbolic system for geometric modeling. In *Computer Graphics (SIGGRAPH '92 Proceedings)*, volume 26, pages 369–378, 1992.

[24] A. Sourin and A. Pasko. Function representation for sweeping by a moving solid. In *Third International Symposium on Solid Modeling and Applications*, pages 383–392, May 1995.

[25] A. Spyridi. *Automatic Generation of High Level Inspection Plans for Coordinate Measuring Machine.* PhD thesis, University of Southern California, 1994.

[26] R. B. Tilove. Set membership classification: A unified approach to geometric intersection problems. *IEEE Transactions on Computer*, C-29(10):874–883, October 1980.

[27] D. M. Tsay and H. M. Wei. A general approach to the determination of planar and spatial cam profiles. *Journal of Mechanical Design*, 118:259 – 265, June 1996.

[28] D. L. Vossler. Sweep-to-CSG conversion using pattern recognition techniques. *IEEE Computer Graphics and Applications*, 5(8):61–68, 1985.

[29] W.P. Wang and K.K. Wang. Geometric modeling for swept volume of moving solids. *IEEE Computer Graphics Applications*, pages 8–17, December 1987.