UNSWEEP: Formulation and Computational Properties

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

Abstract

We introduce and formally define a new geometric modeling operation $\texttt{unsweep}(E, M)$ which, given an arbitrary n-dimensional subset of Euclidean space E and a general motion M, returns the subset of E that remains inside E under M. This new operation is dual to the usual sweeping operation and has important applications in mechanical design. When M is a translation, $\texttt{unsweep}(E, M)$ naturally reduces to the usual Minkowski difference of E and the trajectory generated by the inverted motion $\hat{M}$. We show that unsweep has attractive computational properties and give a practical point membership test for arbitrary general motions. By duality, the established properties of unsweep can be used to develop a practical point membership test for general sweeps.

1 Introduction

1.1 Sweeps in modeling

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 [17], 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 [24, 7, 2]. In graphics, allowing object S to deform as it moves along M is often used to generate complex scenes and visual effects [20, 21]. 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[25, 14]. 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, 12, 21]. But general and reliable procedures for a point membership classification (PMC) [23] 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.

In this paper, we introduce a new operation called unsweep that is dual to the general sweep and has a number of practical applications. It is important that unsweep comes with a relatively straightforward PMC

In the Proceedings of the Fourth ACM Symposium on Solid Modeling and Applcations, Atlanta, GA, May 14-16, 1997. Also available as a Technical Report SAL-1996-3, Spatial Automation Laboratory, 1513 University Avenue, University of Wisconsin-Madison, WI 53706.

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 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):

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

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. $\texttt{unsweep}(S, M)$ is the largest set of points that remains inside S under M;
2. PMC of a point p against $\texttt{unsweep}(S, M)$ reduces to classifying the trajectory of p under M (a curve) against the set S;
3. if X^c denotes the complement of a set X, then the relationship between the two dual operations is given by:

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

The first property suggests that `unsweep` has many practical applications in modeling; the second property indicates that it can be computed effectively; and the third property extends the computational advantages of `unsweep` to general sweeps.

1.3 Example application

Packaging is one of most common problems in mechanical design. Static interference of fixed parts, say within a single assembly or enclosing envelope[2], 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 envelope 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).

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 envelope. 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).

The above example is only one of many applications of the new operation. In general terms, `unsweep` gives a new 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 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, 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 lead 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`. The precise nature of this duality is identified in section 3. Section 4 deals with computational issues, including point membership classification, several strategies for implementing `unsweep` and some 3-dimensional examples. The concluding section 5 briefly discusses the significance of our findings and a number of promising extensions of this work.

[1] The concepts of motion and inverted motion are explained in section 2.1.

[2] In this paper the term 'envelope' is synonymous with 'containing set' and has no relationship to the theory of envelopes.

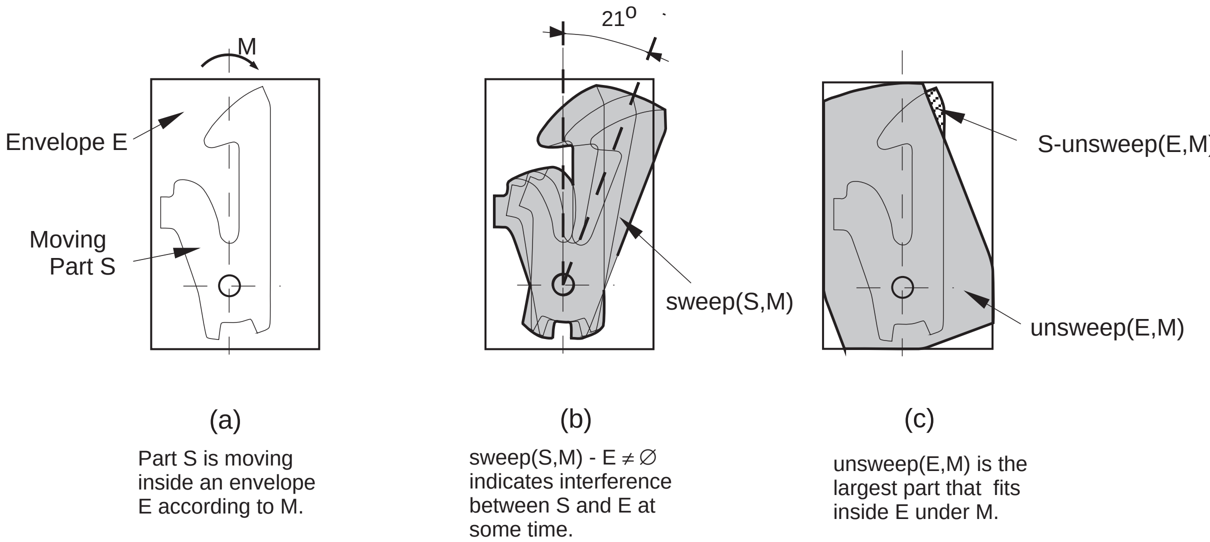


Figure 1: The moving part S has to fit within the given envelope E

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 [10], we define the motion $M(t), t \in [0,1]$ as a one-parameter 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 4.2. We mostly use rigid body motions for illustration purposes but, except when noted, all discussion 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 $\mathcal{F}_\mathcal{W}$; 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 [10].

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 2(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

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

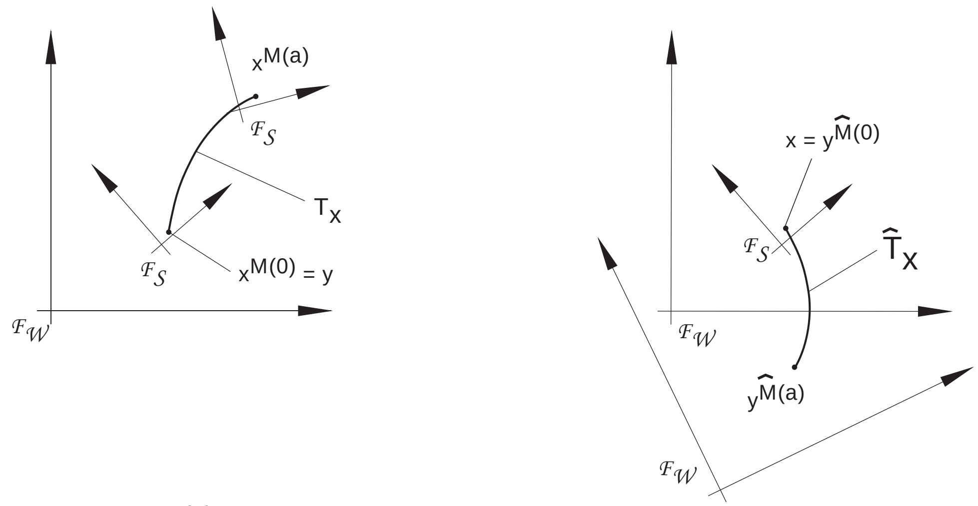
Each instantaneous transformation $M(a)$ in equation (4) has a unique inverse $\hat{M}(a)$ such that $x = \hat{M}(a)[M(a)x]$. Given a transformation $M(t)$ for a range of values of $t \in [0,1]$, we will call transformation $\hat{M}(t)$ *inverted* if it is the inverse of $M(t)$ for every instance of 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 $\hat{M}(t)$.

In other words, the moving observer will see the inverted trajectory (Figure 2(b))

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

[3] We use two-dimensional examples for clarity, but the same arguments hold in any d-dimensional space.



(a)

Trajectory of point x generated by M(t) as seen from the fixed coordinate system $\mathcal{F}_\mathcal{W}$. Point y remains fixed in $\mathcal{F}_\mathcal{W}$.

(b)

Trajectory of point x generated by M(t), as seen from the moving coordinate system $\mathcal{F}_\mathcal{S}$, can be observed by watching the 'trace' of point y fixed in $\mathcal{F}_\mathcal{W}$.

Figure 2: Motions and trajectories

Note here that we introduced point y only to illustrate the difference between what the observer sees from the two coordinate systems $\mathcal{F}_\mathcal{W}$ and $\mathcal{F}_\mathcal{S}$.

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 $\mathcal{F}_\mathcal{S}$, 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 $\mathcal{F}_\mathcal{S}$. 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 $\mathcal{F}_\mathcal{S}$, the 'world' appears to be moving according to $\hat{M}$.

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 the origin, i.e. $T_x = -\hat{T}_x$ [18].

2.2 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

$$M(t)S = \bigcup_{q \in M} S^q \tag{7}$$

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 3(a). This formulation assumes that the sweep is computed from the fixed coordinate system $\mathcal{F}_\mathcal{W}$, 'observing' the moving coordinate system $\mathcal{F}_\mathcal{S}$.

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

$$\mathtt{sweep}(S, \hat{M}) = \bigcup_{q \in \hat{M}} S^q \tag{8}$$

The set of points defined by equation (8) is quite distinct from that defined in equation (7). Figure 3(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 (7) in terms of the inverted trajectories of moving points (i.e. as observed from the moving system $\mathcal{F}_\mathcal{S}$). 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.

2.3 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 $M(t),\ t \in$

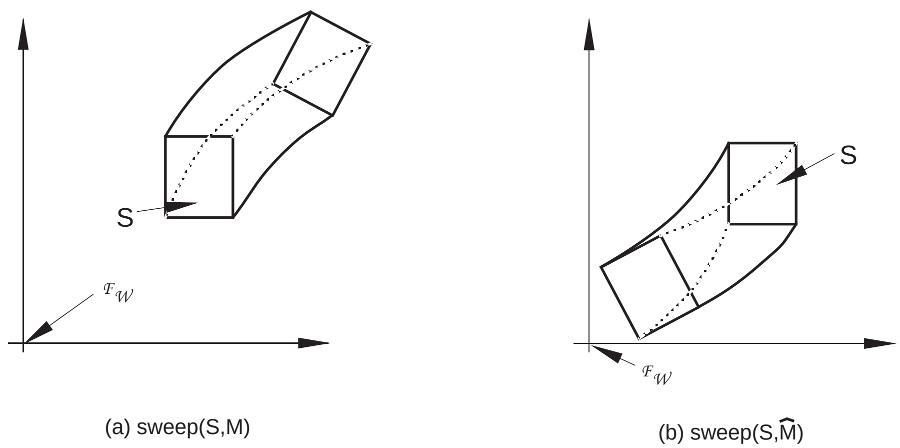


Figure 3: Sweeping a set according to motions M and $\hat{M}$

$[0,1]$. Figure 4(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 4, some of the points $x \in F$ will travel outside of E, while others will never leave E. We now define the unsweep(E, M) as the set of all those points x that remain inside E for all values of t. Formally,

$$\texttt{unsweep}(E, M) \equiv \{x \mid x^{M(t)} \in E, \ \forall t \in [0,1]\} \qquad (9)$$

By this definition, unsweep(E, M) is well-defined and is the largest subset of F that stays inside E while F is moving according to $M(t)$. The characterization of unsweep in definition (9) 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,

$$\texttt{unsweep}(E, M) = \{x \mid T_x \subset E\} \qquad (10)$$

The condition in equation (10) is illustrated in Figure 4(a). As long as the trajectory T_x of point x stays inside of E, $x \in$ unsweep(E, M), while if T_x intersects the complement set E^c then point $x \notin$ unsweep(E, M). In section 4.1 we will use this observation to develop a rigorous point membership classification (PMC) procedure for 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 4(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 that is identical to the dual of sweep defined by expression (2) in section 1.2.

Finally, note that the set of points

$$\texttt{unsweep}(E, \hat{M}) = \bigcap_{q \in M} E^q = \{x \mid \hat{T}_x \subset E\} \qquad (11)$$

is *not* the same set as unsweep(E, M). Figure 5(a) shows the 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 5(b) shows 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 Duality

3.1 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. In this case, each point x of the moving set S sweeps the trajectory

$$T_x = x + M(t), \ t \in [0,1]$$

where both x and $M(t)$ 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 [18]. In other words, every point of S moves on the same trajectory T that is translation invariant and is in fact equivalent to the set of translations

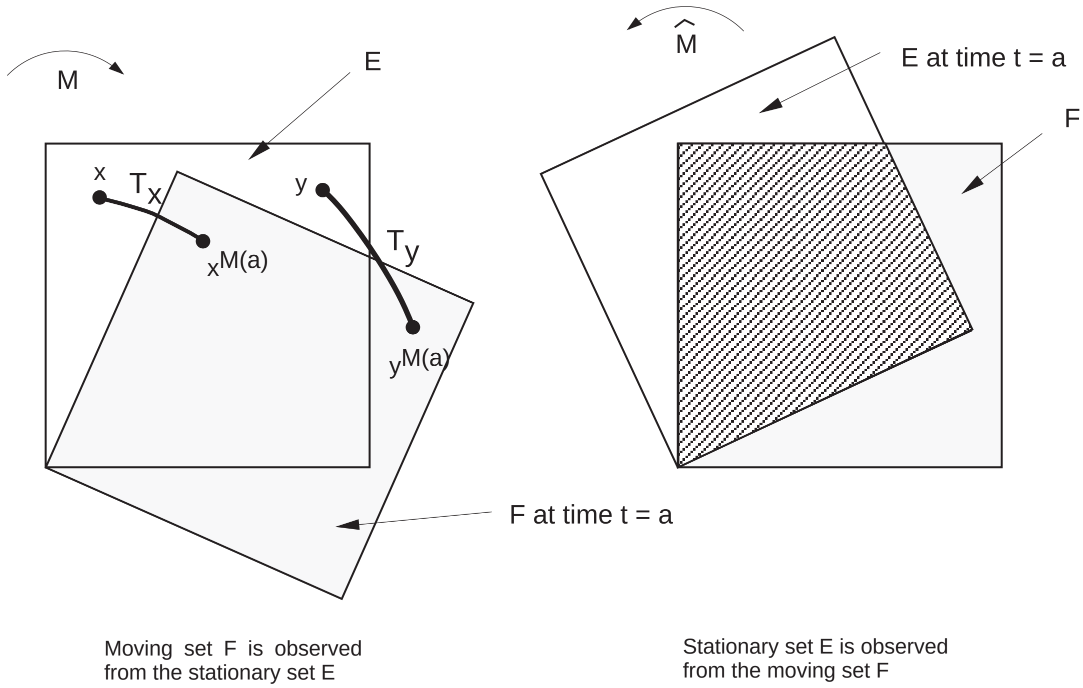


Figure 4: Definition of the `unsweep` operation: two sets E and F in relative motion

M. This implies that set S moves without changing its orientation and that

$$\texttt{sweep}(S, M) = \bigcup_{q \in T} S^q = S \oplus T \tag{12}$$

is also translation invariant. Similarly, using definition (2) of `unsweep` in terms of the moving set S on inverted trajectory, we have

$$\texttt{unsweep}(S, M) = \bigcap_{p \in \hat{T}} S^p = S \ominus \hat{T} \tag{13}$$

where $\ominus$ is the usual Minkowski difference operation [18]. 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 (9) of `unsweep`. Notice that $\hat{T} = -T$ is simply a *reflection* of T with respect to the origin of the frame $\mathcal{F}_{\mathcal{W}}$ and is again translation invariant.

Minkowski operations have many useful properties and are used extensively in geometric modeling, motion planning, and image processing [4, 10, 18, 11, 22]. In particular, definitions of Minkowski operations imply that $\oplus$ and $\ominus$ are dual to each other via

$$S \oplus T = (S^c \ominus T)^c \tag{14}$$

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

3.2 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 equation (14) in set theoretic terms as

$$\begin{array}{c} \bigcup_{q \in T} S^q = [\bigcap_{q \in T} (S^c)^q]^c, \\ \text{or} \\ (\bigcup_{q \in T} S^q)^c = \bigcap_{q \in T} (S^q)^c \end{array} \tag{15}$$

since $(S^c)^q = (S^q)^c$. But the last equality, and hence the duality of $\oplus$ and $\ominus$, is simply a restatement of the usual DeMorgan's laws for operations of union and intersection generalized for arbitrary number or families of sets [9]. The straightforward application of the same law in the case of general motion M yields

$$\begin{array}{c} (\bigcup_{q \in M} S^q)^c = \bigcap_{q \in M} (S^q)^c, \\ \text{or} \\ [\texttt{sweep}(S, M)]^c = \texttt{unsweep}(S^c, \hat{M}) \end{array} \tag{16}$$

which is equivalent to the relationship (3). The es-

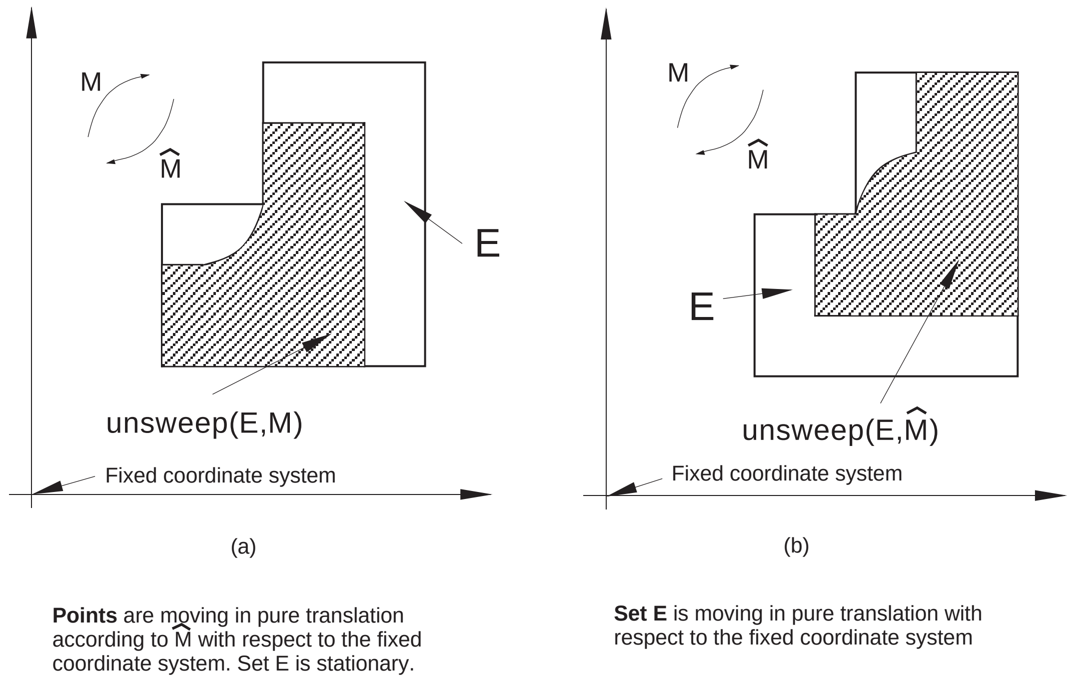


Figure 5: Difference between the sets $\texttt{unsweep}(E, M)$ and $\texttt{unsweep}(E, \hat{M})$

tablished 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.

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 [23]. 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 [17], 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 [23].

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 [15]), and it is easy to construct similar examples showing that `unsweep` of a regular set S may not be 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 [23]. 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 (9)-(10), 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 6 (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 6 (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.'

For example, let us consider a common case when S is any closed set. 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:

$$PMC[x, \texttt{unsweep}(S, M)] = \begin{cases} in, & T_x \subset \mathbf{i}S \\ out, & T_x \cap S^c \neq \emptyset \\ on, & \text{otherwise} \end{cases} \tag{17}$$

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

$$\partial[\texttt{sweep}(S, M)] = \partial[\texttt{unsweep}(S^c, \hat{M})] \tag{18}$$

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 (17) 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 problem 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].

4.2 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

$$A(t) = \begin{bmatrix} \multicolumn{3}{c}{\Theta(t)} & \mathcal{T}(t) \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \text{and} \quad A^{-1}(t) = \begin{bmatrix} \multicolumn{3}{c}{\Theta^T(t)} & -\Theta^T(t)\mathcal{T}(t) \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{19}$$

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 $-\mathcal{T}(t)$.

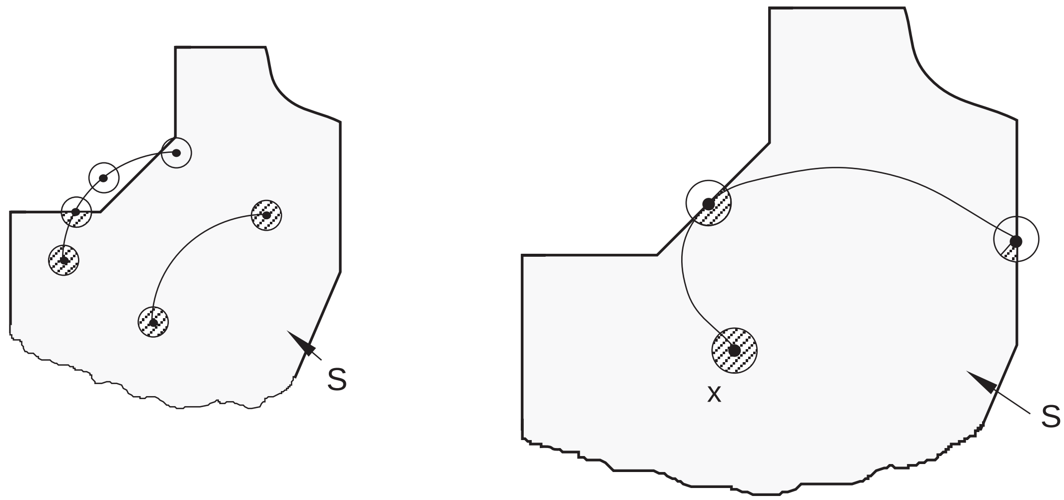
If a point x is represented by a vector $\vec{v}$, and a motion $M(t)$ is represented by a matrix $A(t)$, the trajectory T_x can be written in parametric form simply as $A(t) \cdot \vec{v}$. The parametric form of the curve T_x is suitable for computing the intersection of T_x with the boundary of a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

4.3 Implementation strategies

The two methods of defining $\texttt{unsweep}(S, M)$ naturally suggest two distinct approaches to computing the resulting set of points.

- Definition (2) gives a method for approximating $\texttt{unsweep}(S, M)$ 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



(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.

(b)

Neighborhood of point x gets trimmed every time its trajectory hits the boundary of S.

Figure 6: Neighborhoods of points moving on trajectories generated by motion M

raycasting software and hardware techniques, as described in [14, 13].

- The second formulation defining `unsweep`(S, M) in terms of trajectories of moving points (equation (10)) leads to a well-defined PMC procedure as described above. Ability to perform PMC can be used for computing `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 [16]:

  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 (18)) implies that all the methods used in generating bounding surfaces for computing `sweep` ([12, 1, 21]) 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 `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].

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.

4.4 Examples

Several two-dimensional examples of `unsweep`(E, M) already appeared in the previous sections of this paper. Figure 1(c) shows `unsweep` of a simple rectangular envelope by a clockwise rotation. Figures 5(a) and (b) show the difference between `unsweep`(E, M) and `unsweep`$(E, \hat{M})$ with the same L-shaped envelope E; M is a translation along a circular arc, and $\hat{M}$ is the translational motion along the reflected arc with respect to the origin. These restricted situations, which involve only polygonal envelopes and simple translations or rotations illustrate that the results of `unsweep` are not always intuitive and would be difficult to obtain manually.

Several simple three-dimensional examples of `unsweep` are shown in Figure 7. In Figure 7(a) the envelope 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 7(b) shows another example where the envelope 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 7(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 envelope 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. Note that, based on equation (2) the `unsweep` operation preserves convexity of the envelope; thus, if E is convex, then unsweep(E, M) will also be convex as in Figures 7(a) and 7(c).

Figure 7(b) shows that, in general, unsweep(S, M) does not have to be convex, but, intuitively, it will never be "any less convex" than the generator set S.

The final example in Figure 7(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 (16). 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.

5 Conclusions

5.1 Sweep or unsweep?

It may be tempting to think of `unsweep` as the inverse of `sweep`. This is certainly not true, in the sense that unsweep[sweep(S, M), $\hat{M}$] 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 [18]. While the notion of 'inverse sweep' seems to arise naturally in some applications[19], it does not appear to be well-defined in general. As a *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 significantly 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, 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. We expect that the *combined* properties of `sweep` and `unsweep` should lead to other new algorithms and applications.

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 packaging problems already mentioned in section 1.3, we anticipate many other applications in manufacturing planning, simulation, mechanical design, and analysis of moving parts and assemblies;
- We have shown that `unsweep` comes with attractive 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.

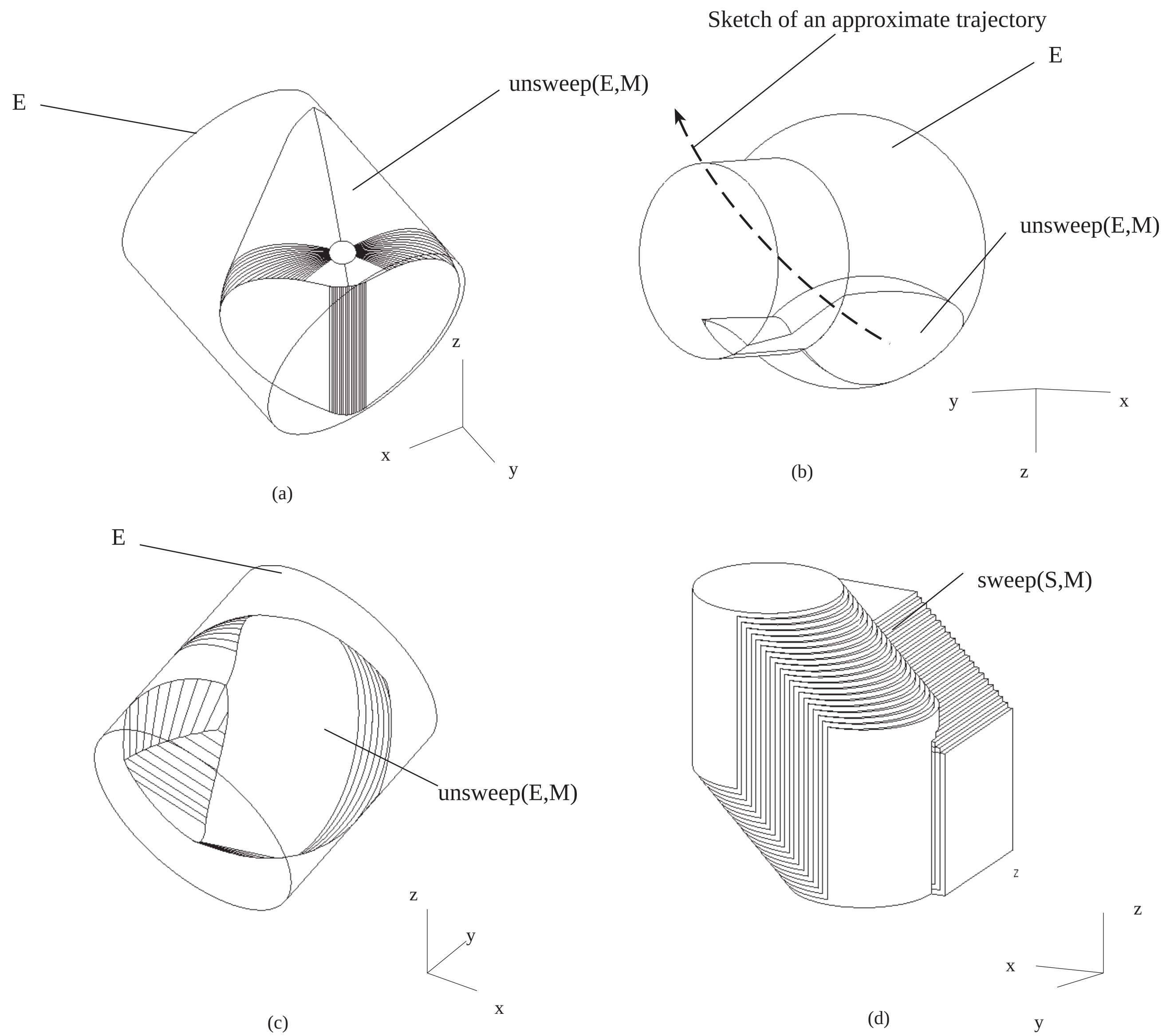


Figure 7: Examples of `unsweep` applications:
(a) E is a cylinder and M is a rotation around the axis aligned with the center of the shown hole.
(b) E is the union of a cylinder and a sphere and M is inverted from a 'helical' motion $\hat{M}$.
(c) E is a cylinder and M is a sequence of two rotations.
(d) A translational sweep of a simple solid S: the solid is constructed as the union of a cube and a cylinder.

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 set-theoretic and topological properties of `unsweep`, detailed PMC procedures for specific sets (e.g. curves, planar, solid, open) and motions (translations, rotations, general rigid, with deformation, etc.), regularization, and others. Extending other properties of Minkowski operations to `sweep` and `unsweep` may also prove interesting and useful.

In the spirit of [17], 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. The authors are grateful to John Uicker for stimulating discussions and to Malcolm Sabin for reading the early draft of this paper and suggesting a number of editorial changes.

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 ans Automation in Design*, 109:50–60, 1987.

[4] G. Evans, R.C. Koppelman and V.T. Rajan. Shaping geometric objects by cummulative 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] K. Kuratowski and A. Mostowski. *Set Theory.* North Holland, Amsterdam/New York/Oxford, 1976. Studies in Logic and Foundations of Mathematics, Vol 86.

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

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

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

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

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

[15] 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.

[16] 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.

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

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

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

[20] 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.

[21] 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.

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

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

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

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