On Shaping With Motion

Horea T. Ilieş Vadim Shapiro

Spatial Automation Laboratory*

ilies@sal-cnc.me.wisc.edu, vshapiro@engr.wisc.edu

Abstract

Mechanical parts are modeled as (predominantly rigid) solid shapes that may move in space in order to function, be manufactured (for example, machine or be machined), and be assembled or disassembled. While it is clear that such mechanical shapes are greatly influenced by collision, interference, containment, and contact constraints through prescribed motions, the motion itself is usually not part of these shape models. This in turn leads to proliferation of computational methods for modeling and analysis of various motion-related constraints.

We show that all motion-related constraints can be formulated and applied within the same computational framework that treats motion as an integral part of the model. Our approach relies on two computational utilities. The first one is the `unsweep` operation which, given an arbitrary n-dimensional subset of Euclidean space E and a general motion M, returns the largest subset of E that remains inside E under M. The second modeling utility is a disjoint decomposition of space induced by the operations of `unsweep` and the standard set operations. The proposed approach subsumes and unifies the traditional sweep-based modeling of moving parts, and provides improved computational support for mechanical shape design.

1 Introduction

1.1 Motion-related modeling problems

Most of mechanical parts must be moved in order to function, manufacture or be manufactured, or assembled. Recognition of this fact led to the requirement that solid models of mechanical parts should be closed under rigid motions [Requicha, 1980], even if the motion itself may or may not be part of the solid model. In the usual representation schemes, motion usually appears in two distinct forms: as a discrete transformation used to position a geometric primitive (for example in a Constructive Solid Geometry tree) or as a continuous one-parameter transformation defining a sweep-like geometric construction [Hui, 1994].

*Department of Mechanical Engineering, 1513 University Avenue, University of Wisconsin-Madison, 53706 USA.

In either case, specifying such transformations is a convenient construction method that does not generally correspond to any physical motion. Yet, it is clear that the geometry of a moving part is largely determined by motion-related considerations, as illustrated by some common examples.

Sweeping The set of all points of the physical space occupied ('swept') by a moving object A at some time during the motion M is usually modeled by $\texttt{sweep}(A, M)$. Sweeping has been the method of choice for modeling material removal processes such as NC machining [Lee and Chang, 1996, Wang and Wang, 1986], computing robot workspaces [Donald, 1985, Lozano-Perez, 1983], and assessing accessibility requirements for maintenance purposes [Schroeder et al., 1994].

Collision detection between two moving parts A and B is often reduced to deciding if $\texttt{sweep}(A, M)$ intersects part B, where M is the relative motion between the two parts [Ganter, 1985]. Amount of interference can be also computed this way, but not the changes (in motion, geometry of parts, or both) needed to avoid the collision when detected. Such problems are common in robotics and mechanism design [Donald, 1985, Uicker, 1997].

Spatial containment (packaging) of moving parts is one of the most common challenges in mechanical design. Every moving part is constrained to remain inside some containing space in order to avoid interference with neighboring parts or assemblies. This is particularly difficult when a new part or assembly is being designed, and designers often have to use conservative *ad hoc* techniques or rely on their experience and intuition. Such practices are known to induce complex iterative steps in the design process [Ilies and Shapiro, 1996]. Containment constraints also appear to dominate most redesign activities, where a previously designed moving part has to remain inside a modified containing space during a (possibly modified) motion. In this case, the lack of containment may be identified using collision detection techniques, but such methods do not provide systematic and reliable means for part and/or space modification required to achieve the specified containment constraints.

Contact constraints are probably among the most complex in the design and analysis of kinematic pairs[1] that have been studied extensively in [Chen, 1982, Litvin, 1994, Shigley and Uicker, 1995, Uicker, 1997, Gupta and Jakiela, 1993]. Cam shafts that open and close the engine valves and the worm-gear drives are two familiar examples of kinematic pairs. More generally, a kinematic pair transmits accurate motions and high loads while maintaining continuous contact. While the lower kinematic pairs (with contact along surfaces) have been extensively studied, few tools are available for the analysis of higher kinematic pairs with other kinds of contacts [Chen, 1982, Shigley and Uicker, 1995], and virtually all design tools are limited to specific parameterized situations.

[1] A kinematic pair consists of two objects that move relative to one another while maintaining a continuous contact [Bottema and Roth, 1979].

The common thread appearing in all four types of problems above is that the answer to the problem is largely determined by a given motion M and partial geometric information. It is obvious that $\texttt{sweep}(A, M)$ is completely determined by A and M; it may be less obvious that every packaging problem has a unique 'extreme' solution for every given motion and either containing space or a moving part; and it may be far from obvious that there is anything unique about kinematic pairs implementing a given motion.

The purpose of this paper is to show that *all* such problems can be formulated and solved within the same computational framework. More specifically, given a solid A and motion M, the whole space may be partitioned into exactly four non-empty sets, such that any problem involving logical statements about A and M has the answer in terms of the union of these four sets. In practical terms, this implies that all motion-related problems may be solved using the same set of computational tools in two stages: (geometric) decomposition of space into the four subsets, and (logically) combining inferences about the four sets into answers to specific problems. For some applications, it may make sense to treat motion M and the induced decomposition of space as part of the model.

1.2 Related work

Without surveying the vast literature on modeling and computing of motion-related tasks, we note that the large number of distinct problem formulations has resulted in an equally large number of computational techniques. Because each problem is usually formulated and solved independently (often using *ad hoc* and heuristic methods), comparison of their solutions is difficult.

The easiest approach (both conceptually and in terms on implementation) is to treat motion as a quasi-static problem, and a moving object as a sequence of static configurations. Then the collision detection problem becomes a multiple static interference detection problem which has known solutions [Boyse, 1979]. But collisions can be missed if the time intervals are poorly chosen and such algorithms are computationally very expensive. If a moving object is represented by its boundary, then collisions may be identified by solving higher order equations for different types of intersection between vertices, edges, and faces of the solids, e.g. [Kawabe et al., 1988, Canny, 1986]. For polyhedral objects, distance-based algorithms are quite popular [Gilbert and Hong, 1989, Rabbitz, 1994, Red, 1984, Shimada et al., 1988]. The basic idea is to repeatedly compute the distance between the objects during motion and constrain it to a strictly positive range. The computations become difficult for more general objects.

Properties of sweeps have been studied extensively, for example in [Boyse, 1979, Ganter and Uicker, 1986, Ganter, 1985, Kieffer and Litvin, 1991, Sambandan, 1990, Blackmore et al., 1997]. A number of algorithms have been formulated, but computing the boundary of the swept volume in the general case remains a difficult computational problem. Recent proposals include the use of implicitly defined solid generators [Pasko and Sourin, 1996] and tracing solutions to differential equations [Blackmore et al., 1997], [Blackmore and Leu, 1992]. Treating a moving object as a four-dimensional point-set, with time being the fourth dimension, is proposed by [Cameron, 1985] and several other researchers. However, representing and computing with four dimensional objects is non-trivial.

Even the simple motion-related problems may result in complex geometries. Because of this, all approaches rely on numerical computations (for example ray-casting [Cameron, 1991]), simplified representations (polyhedral, triangulated, etc.), and localization techniques for speeding up the computations (bounding boxes and spheres, coherence).

We finally mention here the `unsweep` operation, dual to the general `sweep` and formulated by the authors in [Ilieş and Shapiro, 1999, Ilies and Shapiro, 1997]. For a given set E and motion M, $\texttt{unsweep}(E, M)$ returns the largest subset of E that remains inside E during M, thus immediately yielding a solution to one of the packaging problems. We will also make an extensive use of this operation in this paper.

1.3 Outline

The properties of `unsweep` and its relationship to `sweep` are summarized in section 2.3 based on [Ilieş and Shapiro, 1999]. We shall see that `unsweep` provides a convenient and a natural way to formulate many of the motion-related problems, and is the primary tool in inducing the desired decomposition of space. Section 3 shows that all motion-related problem reduce to properties of the four well-defined and disjoint sets forming the partition of the whole space. It is shown that no other sets need be computed in order to answer logical queries about a single moving solid. Section 4 explains how problems of collision detection and elimination, spatial containment, and contact analysis of kinematic pairs may be formulated in terms of the sets in the computed decomposition. Significance of our results in the larger context of design in summarized in section 5. Although for clarity we use throughout the paper mostly two dimensional examples and simple motions, all results hold for the 3-dimensional space and general motions. All pictures in this paper were produced using the geometric data output by the Parasolid geometric kernel.

2 Preliminary Definitions

2.1 Motions

Consider a set of points A with its own coordinate system $\mathcal{F}_A$ moving in a d-dimensional Euclidean space $\mathcal{W}$ with respect to some global fixed coordinate system $\mathcal{F}_{\mathcal{W}}$. A configuration of an arbitrary object is a specification of the position of every point in this object relative to a reference frame [Arnold, 1978]. Therefore, a configuration q of A is a specification of the position $\mathcal{T}$ and orientation Θ of $\mathcal{F}_A$ relative to $\mathcal{F}_{\mathcal{W}}$. The configuration space $\mathcal{C}$ of A is the space of all configurations q of A. Mathematical properties of such a configuration space are well understood and they are extensively discussed in [Latombe, 1991].

A motion M is a one parameter family of transformations $M(t)$, where the parameter $t \in [0, 1]$. We normalize all other intervals $[a, b]$ to $[0, 1]$, so that the results presented here remain valid when $t \in [a, b]$, $0 <$

$a < b$. For the purposes of this work, "motions" and "transformations" are interchangeable and are commonly represented by matrices.

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 $\mathcal{C}$. Each instantaneous transformation $M(a)$, for any $a \in [0,1]$, has a unique inverse $\hat{M}(a)$ such that $x = \hat{M}(a)[M(a)x]$. For a range of values of $t \in [0,1]$, the *inverted* motion $\hat{M}(t)$ of a motion $M(t)$, is the inverse of $M(t)$ for every instance of t.

Each point x of the moving object A that moves according to M describes a trajectory

$$T_x = \{x^q, q \in M\} \tag{1}$$

where x^q denotes point x at configuration q. Consequently, in what follows we will denote set A at configuration q by A^q. From the coordinate system $\mathcal{F}_A$ attached to the moving object A, point x will appear to be moving along the *inverted* trajectory

$$\hat{T}_x = \{x^p, p \in \hat{M}\} \tag{2}$$

Importantly, the relationship between the trajectory and the inverted trajectory of point x is generally not simple. A more detailed discussion can be found in [Ilieş and Shapiro, 1999].

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 $\mathcal{F}_A$ and $\mathcal{F}_\mathcal{W}$. Let M_A and M_W be the *absolute* motions of $\mathcal{F}_A$ and $\mathcal{F}_\mathcal{W}$ relative to a fixed coordinate system in $\mathcal{W}$. The relative motion between $\mathcal{F}_A$ 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 $\mathcal{F}_A$ by

$$M_{A/W} = {M_W}^{-1} M_A \tag{3}$$

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

$$M_{W/A} = {M_A}^{-1} M_W. \tag{4}$$

Equation (3) expresses the motion of $\mathcal{F}_A$ observed from $\mathcal{F}_\mathcal{W}$, while equation (4) expresses the motion of $\mathcal{F}_\mathcal{W}$ as observed from $\mathcal{F}_A$. Note that both equations (3) and (4) express the *same* physical relative motion between $\mathcal{F}_A$ and $\mathcal{F}_\mathcal{W}$, but in different coordinate systems.

2.2 The `sweep` operation

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 A, then the `sweep` of A along M is the set of points swept (or occupied) by A at some time during the motion. Formally,

$$\texttt{sweep}(A, M) = \bigcup_{q \in M} A^q \tag{5}$$

where A^q denotes set A positioned according to q. Figure 1 shows a moving disk whose center translates along the dotted line and the set swept by the disk during its motion.

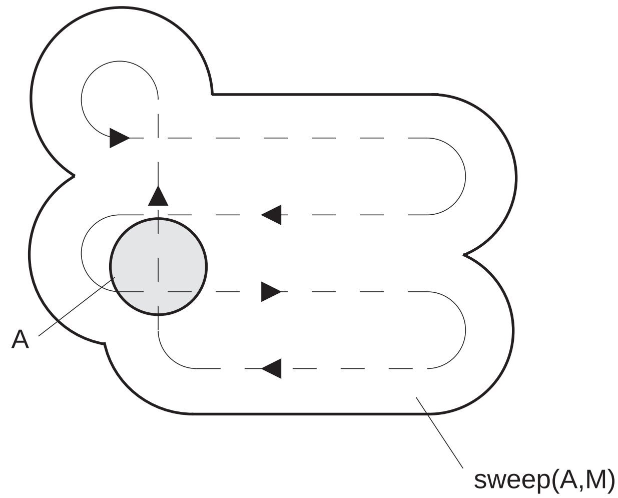


Figure 1: A translating disk sweeping a 2-dimensional set

Sweeps are considered to be one of the basic representation schemes in [Requicha, 1980], 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 [Vossler, 1985, Hui, 1994, Chen and Hoffmann, 1994]. In graphics, allowing object A to deform as it moves along M is often used to generate complex scenes and visual effects [Snyder and Kajiya, 1992, Sourin and Pasko, 1995]. In mechanical design, sweeps of moving parts can be used for collision detection [Ganter, 1985] in assemblies. Sweeping a solid (cutter) along the specified trajectory (tool path) is the preferred method of NC machining simulation [Wang and Wang, 1986, Menon and Robinson, 1993]. Finally, sweeps arise naturally in most situations involving moving bodies, e.g. in studies of robot workspace [Davidson and Hunt, 1987].

By definition, sweeping a moving object A through a motion M requires that both the object A and motion M are given. The result of sweeping is always a set of points containing the original object A; in other words, `sweep` is a 'material growing' operation.

2.3 Unsweep, the operation dual to sweep

If set A and motion M are as before, and $\hat{M}$ denotes the inverted motion, then the dual of the `sweep` is defined as:

$$\texttt{unsweep}(A, M) = \bigcap_{q \in \hat{M}} A^q \tag{6}$$

A second definition of unsweep can be given in terms of trajectories of moving points. If T_x is the trajectory of a moving point $x \in A$, then

$$\texttt{unsweep}(A, M) = \{x \mid T_x \subset A\} \tag{7}$$

The unsweep operation, the precise nature of its duality with sweep, and its computational properties are detailed in [Ilieş and Shapiro, 1999]. Here, we only briefly summarize the principal properties of unsweep:

1. $\texttt{unsweep}(A, M)$ is the largest set of points that remains inside A under M;

2. definition (7) can be used to test whether a given point is "in", "on" or "out" of the set $\texttt{unsweep}(A, M)$;

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

$$[\texttt{unsweep}(A^c, \hat{M})]^c = \texttt{sweep}(A, M) \tag{8}$$

   Intuitively, the duality (8) may be visualized as follows: instead of sweeping the disk A of Figure 1 according to M, one can unsweep the complement of A with the inverted motion $\hat{M}$ and then take the complement of the resulting set. But note that the inverted trajectory of a point under $\hat{M}$ is generally not the same as the trajectory of the point under motion M.

The first property implies that unsweep corresponds to a material removal operation. 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 and computational properties of unsweep to general sweeps and vice versa.

As an example, Figure 2 shows a square A that rotates around the indicated axis (perpendicular to the plane of the square) and the corresponding set $\texttt{unsweep}(A, M)$. Here, the set $\texttt{unsweep}(A, M)$ is the shaded set and contains all points of A that do not "go outside" of A during M.

2.4 Computational Considerations

A convenient way of representing transformations in the d-dimensional Euclidean space E^d is by using homogeneous coordinates and $(d+1) \times (d+1)$ matrices [Foley et al., 1990]. 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). When the motion M is a pure rotation, $\mathcal{T}(t) = 0$, and $A^{-1}(t)$ is obtained from $A(t)$ by replacing the orthonormal

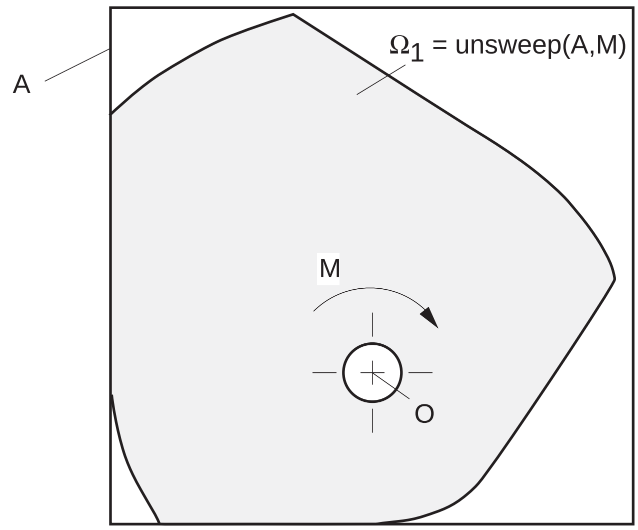


Figure 2: The largest set that remains inside a square that rotates around O by an angle of 45° in the clockwise direction.

sub-matrix $\Theta(t)$ with its transpose $\Theta^T(t)$. For a pure translational motion, $\Theta(t)$ is the identity, and $A^{-1}(t)$ is obtained from $A(t)$ by replacing $\mathcal{T}(t)$ with its reflection $-\mathcal{T}(t)$.

There are two distinct approaches to the computation of `unsweep`: one based on definition (7) of the `unsweep` operation in terms of a trajectory test, the other based on definition (6) in terms of an infinite intersection.

In the first case, the formulation defining `unsweep`(A, M) in terms of trajectories of moving points (equation (7)) leads to a well-defined Point Membership Classification (PMC[2]) procedure as described in [Ilieş and Shapiro, 1999]. Ability to perform PMC can be used for computing the `unsweep` 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 [Requicha and Voelcker, 1985]:

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 fact that `sweep` and its dual `unsweep` share the same boundaries implies that all the methods used in generating bounding surfaces for computing `sweep` ([Martin and Stephenson, 1990, Blackmore and Leu, 1990, Sourin and Pasko, 1995]) are also applicable to computing `unsweep`. The degree of difficulty of the second

[2]A Point Membership Classification procedure is a function that, given a set A, classifies a point x as being 'in', 'on', or 'out' of A.

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 $\texttt{unsweep}(A, M)$ using the PMC procedure, as described in [Ilieş and Shapiro, 1999].

If a point x is represented by a vector $\mathbf{x}$, and a motion $M(t)$ is represented by a matrix $A(t)$, the trajectory T_x of x can be written in parametric form simply as $A(t) \cdot \mathbf{x}$. 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) A, as needed for the PMC procedure. 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 [Jüttler and Wagner, 1996].

On the other hand, definition (6) gives a method for approximating the `unsweep` by a finite intersection of sets $A^{\hat{M}(t)}$ positioned at discrete time intervals according to the inverted motion $\hat{M}$. Intuitively, at every time step $t = a$, the "unwanted" portion of $A^{\hat{M}(a)}$ that protrudes outside of $A^{\hat{M}(0)}$ 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 A is a solid, good performance and quality of the approximations can be obtained by using raycasting software and hardware techniques, as described in [Menon and Robinson, 1993, Menon et al., 1994].

3 Space Partition by Motional Primitives

3.1 Motional Primitives

We postulate a *motional primitive* to be a set whose shape is determined only by the geometry of the moving object and the specified motion, such that no motional primitive contains another. Finding all motional primitives for a set X and motion M is important, because solutions to many problems involving X and M can be constructed from the motional primitives. Observe that `unsweep` is a natural tool for generating motional primitives, because `unsweep` (in contrast to `sweep`) always results in a smaller shape. For illustration, consider a two-dimensional square X that rotates relative to the reference frame according to a rigid body motion M as shown in Figure 3. Observe that if X moves relative to the world $\mathcal{F}_{\mathcal{W}}$ according to M, then $\mathcal{F}_{\mathcal{W}}$ moves relative to X according to the inverted motion $\hat{M}$ defined in section 2.1. We define two motional primitives as:

$$\Omega_1 = \texttt{unsweep}(X, M) \tag{10}$$

and by

$$\Omega_4 = \texttt{unsweep}(X^c, \hat{M}) \tag{11}$$

The two resulting shapes are illustrated in Figure 3. To see that Ω_1 and Ω_4 are indeed primitives, observe that $\Omega_1 \subset X \subset \Omega_4^c$, and any `sweep` of X and X^c results sets that are respectively larger than X and X^c. The two primitives have important physical interpretations. Ω_1 is the largest subset of X that remains inside X

during the motion M or, equivalently, does not collide with the complement X^c of X. On the other hand, Ω_4 contains all points that remain inside X^c or, equivalently, do *not* collide with X, while X moves relative to them according to M.

One way to find the other motional primitives is to construct all possible sets that answer the *logical* statements regarding X and its motion M, and then take their intersections to determine the smallest useful sets. Fortunately, the set of all such statements forms a *finite* Boolean algebra generated by Ω_1, Ω_4, X, and the standard set operations. The motional primitives correspond to the atoms of this algebra, or canonical intersection terms [Shapiro, 1997]. The Appendix shows that there are only four such non-empty terms, including Ω_1 and Ω_4. The other two sets are:

$$\Omega_2 = X \cap [\mathtt{unsweep}(X, M)]^c \tag{12}$$

and

$$\Omega_3 = X^c \cap [\mathtt{unsweep}(X^c, \hat{M})]^c. \tag{13}$$

Ω_2 contains all points of X that go outside of X or, equivalently, that collide with the complement X^c of X, while Ω_3 contains all points of X^c that go outside of the complement X^c or, equivalently, that collide with X, while X moves relative to them according to M. As Figure 3 illustrates, the four motional primitives form a partition of the whole space, i.e.

$$\Omega_1 \cup \Omega_2 \cup \Omega_3 \cup \Omega_4 = \mathcal{W}; \qquad \Omega_i \cap \Omega_j = \emptyset, i \neq j \tag{14}$$

Observe that definitions of sets Ω_1 and Ω_2 are expressed in terms of motion M, while the definitions of sets Ω_3 and Ω_4 contain the *inverted* motion $\hat{M}$. This is due to a switch in the reference frames: in equations (10) and (12) the points of X are moving according to M relative to the points outside of X, while in equations (13) and (11) the points outside of X are moving according to $\hat{M}$ relative to X. Using the duality relationship (8) between `unsweep` and `sweep`, one may express the primitive sets also in terms of the `sweep` operation and set complement, but not by `sweep` alone.[3]

3.2 Properties of the Partition

If we start with a closed[4] set X, then $\Omega_1 = \mathtt{unsweep}(X, M)$ are also closed sets. But if X is closed, then X^c is an open set and hence $\mathtt{unsweep}(X^c, \hat{M})$ and Ω_4 are also open sets. The remaining two primitive sets are in general neither open nor closed since they result from an intersection between an open set and a closed set. From the properties of `unsweep` [Ilieş and Shapiro, 1999], it is also possible that sets Ω_1, ... , Ω_4 may have "dangling" edges, or, in other words, they are not necessarily regular[5] sets.

[3] It nevertheless may be more convenient to have the primitive sets expressed in terms of `sweep` instead of `unsweep`. One reason might be, for example, that a computational utility for computing `sweep` may already exist.

[4] In a set-theoretic sense, a *closed* set is a set that contains its boundary points. By contrast, an *open* set does not contain its boundary points. Formal definitions of open and closed sets can be given in terms of neighborhoods of points [Armstrong, 1983].

[5] Informally, a *regular* set is a set that does *not* contains lower dimensional subsets, such as faces, edges and vertices, that are not adjacent to the set's interior [Requicha, 1980].

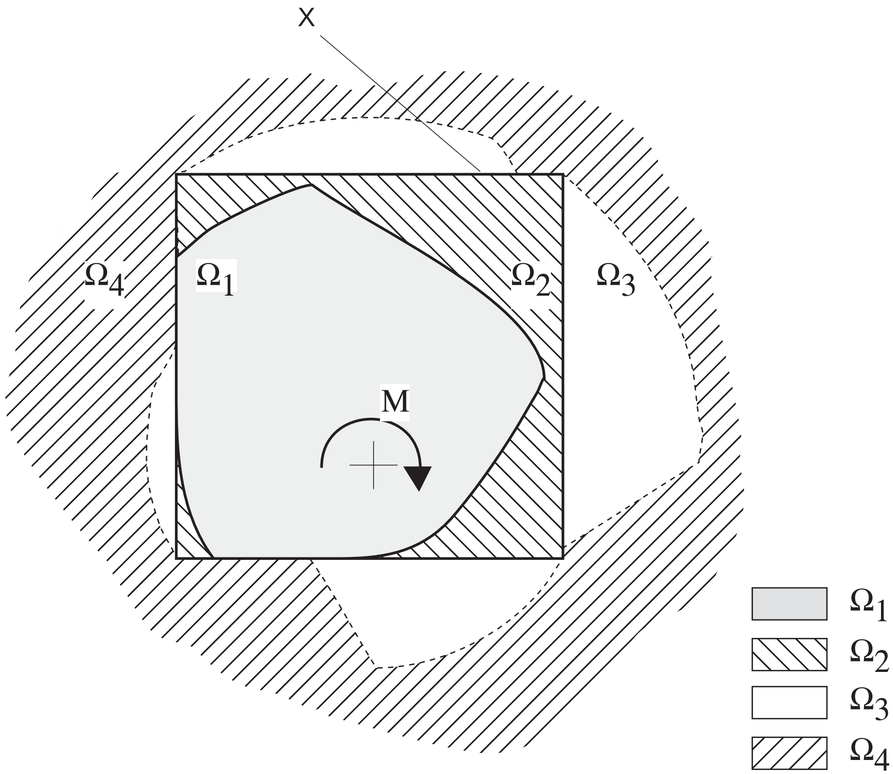


Figure 3: Space partition induced by the operation of `unsweep` and the standard set operations. All points of each of the four sets behave the same relative to X under motion M: either all remain inside X (Ω_1), go outside X (Ω_2), collide with X (Ω_3) or do not collide with X (Ω_4).

Depending on the motion M and the geometry of X, some of the four sets defined in equations (10),(11), (12) and (13) might be empty. This is easily observed when X is a 2-dimensional disc rotating around its center. As shown in Figure 4(a), in this case $\Omega_1 = X$, $\Omega_2 = \emptyset$, $\Omega_3 = \emptyset$ and $\Omega_4 = X^c$. In other cases, it is also possible that $\Omega_1 = \emptyset$, for example when X is a square translating along a line (as in Figure 4(b)), case in which Ω_2, Ω_3 and Ω_4 are nonempty. At the other extreme, since Ω_4 is unbounded, it can not be empty for any finite motion that does not cause X to cover the whole space.

The above definitions of motional primitives $\Omega_1, ..., \Omega_4$ remain valid for any object X moving according to a general motion M in an n-dimensional space. By the properties of Boolean algebra, any other logical expression about the motion M of X relative to the world $\mathcal{F}_\mathcal{W}$ or inverted motion $\hat{M}$ ($\mathcal{F}_\mathcal{W}$ moves relative to X) *must* evaluate to the union of some motional primitives, and there are at most $2^4 = 16$ possible outcomes. For example, it is easy to see that duality (8) between `sweep` and `unsweep` implies that

$$\texttt{sweep}(X, M) = \Omega_1 \cup \Omega_2 \cup \Omega_3 = \Omega_4^c \tag{15}$$

More generally, all points in each of the motional primitives form an equivalence class because they all have

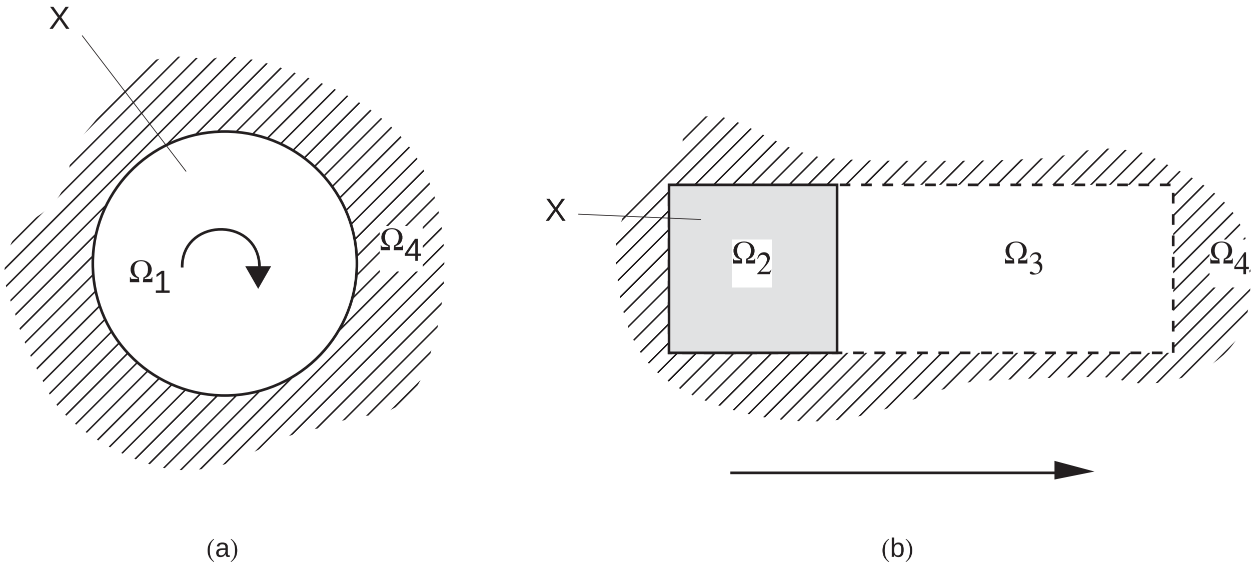


Figure 4: Some of the four sets might be empty, depending on the motion: in (a) Ω_2 and Ω_3 are empty sets, while in (b) Ω_1 is an empty set

the *same type of interaction* with set X under motion M, and solution of every motion-related problem is determined by the choice of these equivalence classes. Thus, the same space partition contains all the geometric information needed for solving any problems involving collision, interference, containment and contact constraints. We briefly discuss each type of these problems in next section.

4 Applications

In this section we show how mechanical design problems involving collision interference, containment, packaging, and contact constraints may be formulated within the computational framework of space partition into the motional primitives described in section 3.

4.1 Design for spatial containment

Consider a common task of designing the shape of a part S that must move according to a prescribed motion M relative to a reference frame $\mathcal{F}_{\mathcal{W}}$, and has to remain inside a given containing set E at all times. For example, in Figure 2 set E is the square, and the motion is a rotation around the center O of the hole shown. A typical approach to the design of a new part is to predict an initial shape S, and then sweep S according to M to see if it ever goes outside of E. If the shape does not fit, the designer knows that some material of S needs to be removed, which in turn may require material to be added elsewhere. Unfortunately, this sweep-based test does not indicate, in general, the information on the required and allowed modifications of the initial shape. Therefore, the designer heuristically estimates the required changes, then sweeps the modified part again, and so on until the sweep of the part is contained inside the required space. In this scenario, even an experienced designer is likely to be forced into time consuming iterations that considerably slow down the design process.

This problem is easily formulated and solved using the `unsweep` operation. Specifically, the motional primitive $\Omega_1 = \texttt{unsweep}(E, M)$ immediately yields the largest subset of E that would remain inside it during M. Therefore, any functional part must be contained in Ω_1 and difference of any given part with Ω_1 is exactly the material that needs to be removed in order to satisfy the containment constraint. No further guessing or design iterations are needed.

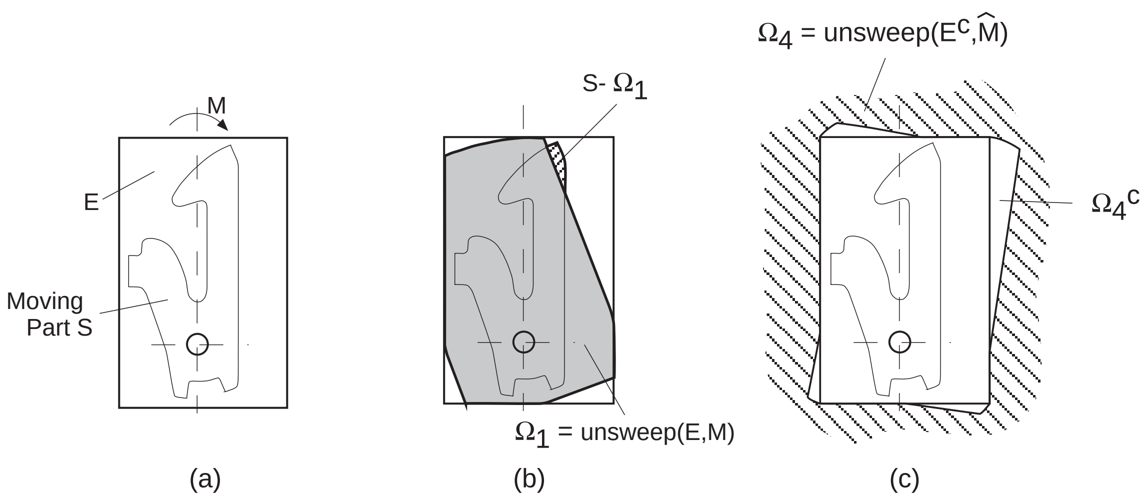


Figure 5: The moving part S has to fit within the given containing set E

The latter observation is particularly significant since most mechanical designs today are in fact redesigns of existing parts and assemblies. For example a secondary hood latch designed by a major automotive supplier is shown in Figure 5(a). The range of the motion M is known, and the latch must remain inside the shown rectangular region. The computed set Ω_1 shown in Figure 5(b) indicates the largest possible latch that can fit inside the space, while the difference of the latch and Ω_1 shows exactly which points of the latch protrude during the motion.

Suppose now that for some reason, modifications to latch S are not possible. Then the designer has no choice but to adjust the size of the containing set E. It would be straightforward to set $E = \texttt{sweep}(S, M)$, but this would also mean that this space must recomputed every time the part is changed. Alternatively, it is also easy to determine the containing set that would accommodate *every* part that fits inside the rectangle E: it is simply $\texttt{sweep}(E, M)$ or the complement of Ω_4 and is shown in Figure 5(c).

4.2 Design for Contact

Mechanical machinery heavily relies on moving parts that maintain contact with each other in relative motion and which are commonly known as kinematic pairs [Hunt, 1978]. Based on the type of contact, one can distinguish two types of kinematic pairs: lower pairs[6] which enjoy a surface-surface contact and higher pairs that comprise objects displaying any other type of contact such as point-surface, edge-surface etc. Few familiar examples of higher pairs are cam-follower mechanisms, gear drives, and Geneva mechanisms. While it is known that there are only six lower pairs, namely prismatic, cylindrical, spherical, helical, revolute and planar joints, there are infinitely many higher pairs [Hunt, 1978, Phillips, 1984]. This implies that the shape design problems raised by these higher pairs are more difficult and involve more complex shapes than those found in the six lower pairs.

In this section, we will show how the space partition introduced in section 3 allows the formulation of contact constraints and their translation into contacting shapes. We will use the fact that the portions of the boundaries of the objects that move relative to and maintain contact with each other are subsets of the boundaries of the four primitive sets. To illustrate this, consider a circular object A that translates according to a motion M as shown in Figure 6, such that the center of the disk moves along the dotted line. The problem is to find an object B that remains in contact with A such that A moves relative to B according to M without interfering with B. There is no loss of generality in assuming that one of the shapes is stationary as long as M is the relative motion between A and B.

Not all the points of the boundary of B need to come in contact with the boundary of A: that particular portion of the boundary of B that does come in contact with A is referred to as the *contact boundary* of B. Recall that the motional primitive Ω_4 defined in equation (11) is the largest set (and, hence, contains *all* points) that will not collide with A during M, and that this set is unbounded. Moreover, observe that its boundary $\partial\Omega_4$ in Figure 6 remains in contact with circle A at all times while A moves according to M, i.e., at any given instance, the intersection between the boundary of A and the boundary of Ω_4 is not empty. In fact, $\partial\Omega_4$ must contain all points that will come in contact with circle A during its motion. Hence, the contact boundary of any object B has to be a subset of $\partial\Omega_4$, and therefore any such set B must be a subset of the set $\Omega_4 \cup \partial\Omega_4$.

As we already saw in Figure 1, for a given object A an arbitrary motion M, the boundary of Ω_4 may not maintain contact with A at all times. But Ω_4 represents the *largest* object that does not collide with A and hence the largest object that moves in contact with A, if such an object exists at all for the given A and M. Therefore, if A does lose contact with $\partial\Omega_4$, then the desired kinematic pair with A and M *does not exist*, and the design problem cannot be solved as posed. This analysis applies to all kinematic pairs, including common situations when A is a follower, a rack, or a Wankel piston or combustion chamber.

It should be noted that existence of contact boundary in itself is not a guarantee of correct kinematic

[6] A mechanism whose joints are all lower pair is called a linkage.

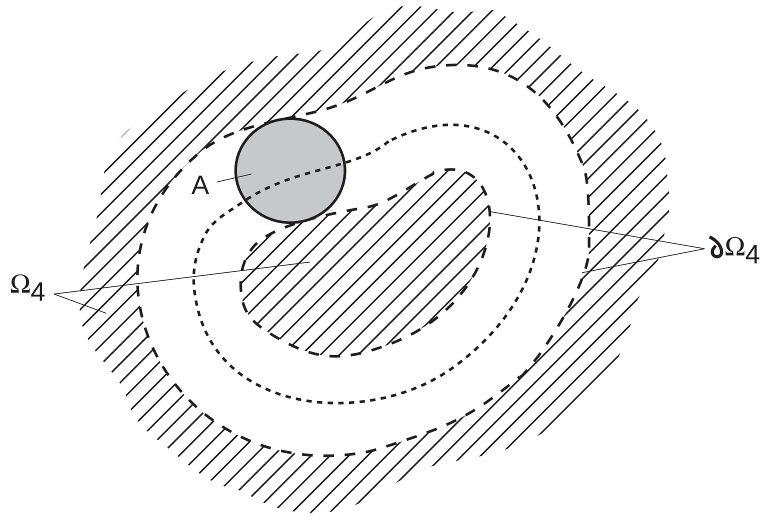


Figure 6: Any stationary object B that maintains contact with the moving A must be a subset of $\Omega_4 \cup \partial\Omega_4$. If we assume that A is the follower, then the bounded set may be the cam that would move relative to A according to the given relative motion M.

---

functionality. For example, our analysis says little about possibility of undercutting, where the type of contact may actually change the intended motion of the kinematic pair. Methods for undercutting detection and elimination based on the theory of singularities [Bruce and Giblin, 1992] have been studied in [Litvin, 1994].

4.3 Collision Detection and Elimination

As a final example, we take a fresh look at the old problems of interference detection and elimination stated in terms of the motional primitives in the space partition. Consider a moving object A moving relative to an object B as shown in Figure 7. It should be obvious that in this case the two objects will collide. For the moment, let us forget about object B and focus on the space partition in terms of motional primitives of A. In this example, $\Omega_1 = \emptyset$ and $A = \Omega_2$; also recall that Ω_4 contains all points that will not collide with A, while Ω_3 is largest set that collides with A while A moves relative to Ω_3 and Ω_4 according to M.

Therefore, a collision between A and B occurs if and only if $\Omega_3 \cap B \neq \emptyset$, or, equivalently, if $B \not\subseteq \Omega_4$. Moreover, the amount of interference between A and B is given by $\Omega_3 \cap B$, which also gives the material that needs to be removed from B in order to eliminate the collision. In addition, any modifications in the shape of B for reasons other than those related to collision and interference between A and B may be performed freely, as long as the B remains a subset of Ω_4.

By symmetry, the partition of space by motional primitives of B and the inverted relative motion should be used when collision elimination must be achieved through changes in the shape of A. Thus, each moving

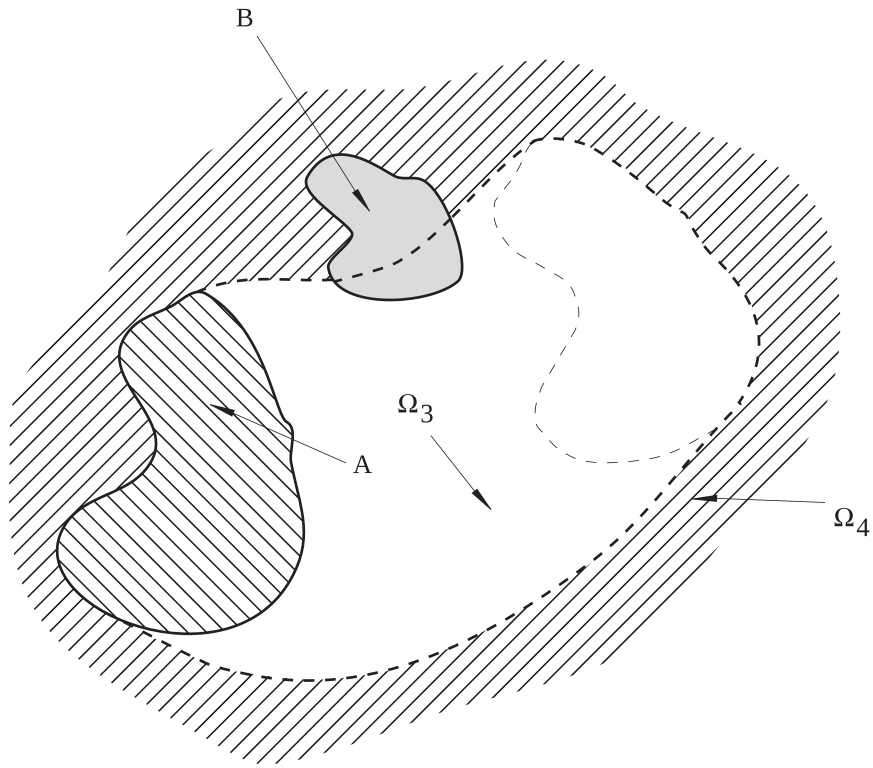


Figure 7: Object A, which moves relative to B as shown, induces a partitioning of the space in which it moves. Set Ω_4 can be used to detect collision and eliminate it by changing the shape of B.

shape induces a distinct partition of space; either one of the partitions is sufficient to detect the collision, but both partitions may be useful for collision elimination.

5 Conclusions

By computing the space partition in terms of the motional primitives of a moving object, one transforms a difficult dynamic problem into a much simpler static one. Once the space partition is computed, only static geometric queries about the four motional primitives remain to be performed, such as point membership classification tests, logical and set operations, and other standard geometric computations. We showed that this approach to modeling motion-related problems unifies and enforces a systematic treatment of all problems that may be formulated in terms collision, interference, containment and contact constraints.

Many applications involve multiple moving and static shapes and primitives; in such situations finer space decompositions may be useful [Shapiro, 1997]. The limitation of our approach, that is common to all sweep-based formulations, is that the motional primitives as defined in this paper do not capture the kinetic phenomena.

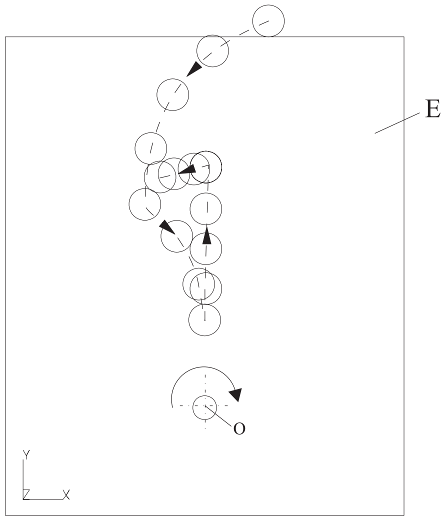
The proposed approach to shaping moving parts does not completely solve the design problem for several reasons. The shape of mechanical parts is determined by many additional constraints and considerations spanning numerous disciplines, including kinetic, strength, thermal, manufacturing, and aesthetic issues. In most cases, the motion-related constraints do not completely define the shape of the part, and the same part geometry is likely to serve many diverse functional goals [Ulrich and Seering, 1990]. It is important that the proposed space partition will not impose further arbitrary restrictions on the shape, but will identify the extremal (i.e. largest contained or smallest containing) shapes satisfying the specified constraints and may be used for specifying families of functionally equivalent parts. For example, a secondary latch must remain inside the given containing set E, under a rotation in the clockwise direction with an angle of $21°$ about point O; the same latch must remain in contact with a vertically moving striker during its operation while the striker passes through a given set of configurations *relative* to the latch as shown in Figure 8(a). Furthermore, both the containing set of the latch, and the striker which the latch comes in contact with induce two different space partitionings of E^3. One contains the largest set remaining inside E, shown in Figure 8(b), and the other partitioning contains the largest set coming in contact with the striker, shown in Figure 8(c). The largest secondary latch that satisfies imposed contact and containment requirements is shown in Figure 8(d) and obtained from a Boolean intersection of objects shown in Figures 8(b) and (c). Any other functioning latch satisfying the same requirements[7] must be a subset of this largest latch.

Much of the geometry of the latch shown in Figure 5 was chosen rather arbitrarily and without serving a functional purpose [Ilies and Shapiro, 1996]. Such designs are never quite correct and undergo numerous changes before being put into production. More generally, mechanical designs rarely stay unchanged for a long period of time. Changes in customer specifications, relationships with suppliers, engineering improvements, product failures, and economics of manufacturing are few of the many reasons why mechanical designs are constantly being modified. It is during these modifications when we discover that the geometric models of the parts alone do not carry much of the required information. The space partitioning proposed in this paper fills in one such informational gap.

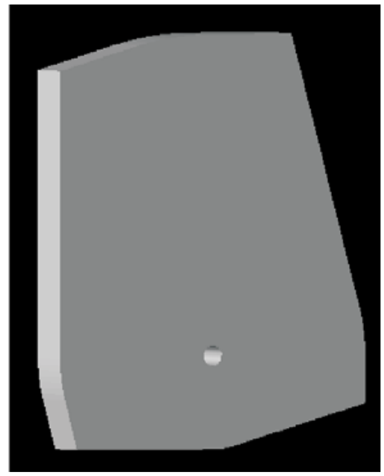
Acknowledgments

This work is supported in part by the National Science Foundation grants DMI-9502728 and DMI-9522806 and DARPA contract N00014-96-C-0168. The examples were created using the Parasolid modeling system, courtesy of EDS Unigraphics.

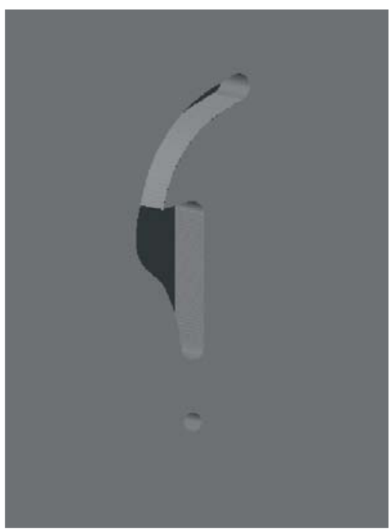
[7]For detailed analysis of the latch's functionality, see [Ilies and Shapiro, 1996].



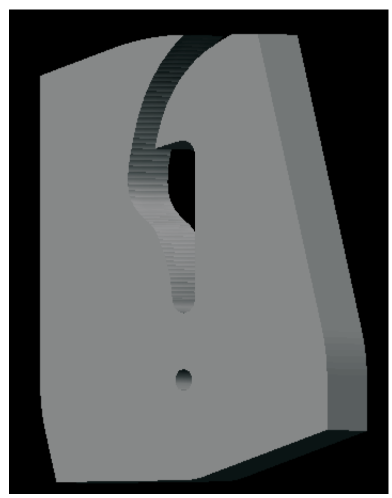
(a)



(b)



(c)



(d)

Figure 8: The largest secondary hood latch (d) is obtained from a Boolean intersection of the largest shape satisfying the containment constraint (b) and the largest shape satisfying the contact constraints (c). This largest latch remains inside the specified containing set, and hence it does not collide with the neighboring parts, and it also satisfies the kinematic functionality.

Appendix

Here we show that set X and the two motional primitives $\Omega_1 = \texttt{unsweep}(X, M)$ and $\Omega_4 = \texttt{unsweep}(X^c, \hat{M})$ together with the Boolean operations $\cup$, $\cap$, and $-$ generate a finite Boolean algebra with at most 16 elements. This algebra has at most four atoms (canonical intersection terms) that define the partition of the d-dimensional space in which X moves. In other words, any finite sequence of Boolean set operations on X, Ω_1 and Ω_4 can be expressed as a union of these four atoms, which in turn correspond to the four motional primitives defined by equations (10)-(13). We also note that there are many different ways to generate the same Boolean algebra, but the resulting partition of space is unique[Shapiro, 1997].

Recall that

$$\Omega_1 = \texttt{unsweep}(X, M) \qquad \Omega_4 = \texttt{unsweep}(X^c, \hat{M}) \tag{16}$$

The following relationships follow immediately from the definition of `unsweep` and can be observed in Figure 3:

$$\Omega_1 \subset X; \qquad X^c \subset \Omega_1^c \tag{17}$$

$$\Omega_4 \subset X^c; \qquad X \subset \Omega_4^c \tag{18}$$

$$\Omega_1 \subset \Omega_4^c; \qquad \Omega_4 \subset \Omega_1^c \tag{19}$$

$$\Omega_1 \cap \Omega_4 = \emptyset \tag{20}$$

$$X \cap \Omega_4 = \emptyset \tag{21}$$

$$X^c \cap \Omega_1 = \emptyset \tag{22}$$

All eight canonical intersection terms formed by Ω_1, Ω_4, and X, and their set complements Ω_1^c, Ω_4^c, and X^c are shown in the first column of Table 1. Based on the above containment relationships, each term is simplified in the second column, with the final value shown in the third column of Table 1. It is easy to check that there are only four non-empty sets, two of which are the original motional primitives Ω_1 and Ω_4, and the other two are defined by the last two rows of the table and correspond to our definitions of the motional primitives Ω_2 and Ω_3 by equations (12) and (13) respectively. Observe that all such combinations (from the first column) result in sets that are either one of the four primitive sets defined by equations (10)-(13) or the empty set $\emptyset$.

References

Armstrong, M., 1983, *Basic Topology*, Springer-Verlag.

Arnold, V., 1978. *Mathematical Methods of Classical Mechanics*, Springer-Verlag, New York.

Blackmore, D. and Leu, M., 1990, A Differential Equation Approach to Swept Volumes, *Proceedings of the 2nd International Conference on Computer Integrated Manufacturing*, pp. 143–149.

Combinations of sets		Intermediate results		Results	Equations used
$\Omega_1 \cap \Omega_4 \cap X$	=	$\Omega_1 \cap \Omega_4$	=	$\emptyset$	(17), (20)
$\Omega_1 \cap \Omega_4 \cap X^c$	=	$\Omega_1 \cap \Omega_4$	=	$\emptyset$	(18), (20)
$\Omega_1 \cap \Omega_4^c \cap X$	=	$\Omega_1 \cap X$	=	Ω_1	(19), (17)
$\Omega_1 \cap \Omega_4^c \cap X^c$	=	$\Omega_1 \cap X^c$	=	$\emptyset$	(19), (22)
$\Omega_1^c \cap \Omega_4 \cap X$	=	$\Omega_4 \cap X$	=	$\emptyset$	(19), (21)
$\Omega_1^c \cap \Omega_4 \cap X^c$	=	$\Omega_4 \cap X^c$	=	Ω_4	(19), (18)
$\Omega_1^c \cap \Omega_4^c \cap X$	=	$\Omega_1^c \cap X$	=	Ω_2	(18)
$\Omega_1^c \cap \Omega_4^c \cap X^c$	=	$\Omega_4^c \cap X^c$	=	Ω_3	(17)

Table 1: All possible combinations of Ω_1, Ω_4, X and their set complements with the boolean set operations reduce to the cases shown in the first column.

Blackmore, D. and Leu, M., 1992, Analysis of Swept Volume via Lie Groups and Differential Equations, *The International Journal of Robotics Research*, 11(6):516–536.

Blackmore, D., Leu, M., and Wang, L., 1997, The sweep-envelope differential equation algorithm and its application to NC machining verification, *Computer Aided Design*, 19(9):629–637.

Bottema, O. and Roth, B., 1979, *Theoretical Kinematics*, Dover Publications.

Boyse, J., 1979, Interference Detection Among Solids and Surfaces, *Communications of the ACM*, 22(1):3–9.

Bruce, J. and Giblin, P., 1992, *Curves and Singularities*, Cambridge University Press.

Cameron, S., 1985, A Study of the Clash detection Problem in Robotics, *IEEE*, pp. 488–493.

Cameron, S., 1991, Efficient bounds in constructive solid geometry, *IEEE Computer Graphics and Applications*, 11(3):68–74.

Canny, J., 1986, Collision Detection for Moving Polyhedra, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8:200–209.

Chen, F., 1982, *Mechanics and Design of Cam Mechanisms*, Pergamon Press.

Chen, X. and Hoffmann, C. H., 1994, Towards Feature Attachment, Technical Report CSD-TR-94-010, Department of Computer Science, Purdue University, West Lafayette, IN.

Davidson, J. and Hunt, K., 1987, Robot Workspace of a Tool Plane: Part1 - A Ruled Surface and Other Geometry, *Journal of Mechanisms, Transmissions and Automation in Design*, 109:50–60.

Donald, B., 1985, On Motion Planning with Six Degrees of Freedom: Solving the Intersection Problems in Configuration Space, *IEEE*.

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

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

Ganter, M. A. and Uicker, J. J. J., 1986, Dynamic Collision Detection Using Swept Solids, In *American Society of Mechanical Engineers Design Engineering Technical Conference.*

Gilbert, E. and Hong, S., 1989, New algorithm for detecting the collision of moving objects, In *IEEE International Conference on Robotics and Automation*, pp. 8–14, Scottsdale, AZ, USA.

Gupta, R. and Jakiela, M., 1993, Simulation and Shape Synthesis of Kinematic Pairs via Small-Scale Interference Detection: Advanced Examples Demonstrating Shape Optimization and Non-Fixed Axis Members, In *Proceedings of the 5th International Conference on Design Theory and Methodology*, pp. 25–35. ASME.

Hui, K., 1994, Solid Modelling with Sweep-CSG Representation, *Proceedings of CSG 94 Set Theoretic Solid Modelling; Techniques and Applications*, pp. 119–131.

Hunt, K., 1978, *Kinematic Geometry of Mechanisms*, Clarendon Press – Oxford.

Ilies, H. and Shapiro, V., 1996, An Approach To Systematic Part Design, In *Proceedings of the 5th IFIP WG5.2 Workshop on Geometric Modeling in CAD*, pp. 383–392.

Ilies, H. and Shapiro, V., 1997, `UNSWEEP`: Formulation and Computational Properties, In *Proceedings of the fourth ACM Symposium on Solid Modeling and applications: Solid Modeling '97*, Atlanta, Georgia.

Ilieş, H. and Shapiro, V., 1999, The Dual of Sweep, *Computer Aided Design*, 31(3):185–201.

Jüttler, B. and Wagner, M., 1996, Computer-Aided Design with Spatial Rational B-Spline Motions, *Journal of Mechanical Design*, 118:193 – 201.

Kawabe, S., Okano, A., and Shimada, K., 1988, Collision Detection Among Moving Objects in Simulation, In *Proceedings of the USA-Japan Symposium on Flexible Automation - Crossing Bridges: Advances in Flexible Automation and Robotics.*

Kieffer, J. and Litvin, F., 1991, Swept volume determination and interference detection for moving 3-D solids, *Journal of Mechanical Design*, 113(4):456–463.

Latombe, J.-C. 1991, *Robot Motion Planning*, Kluwer Academic Publishers, Boston / Dordrecht / London.

Lee, Y. and Chang, T., 1996, Application of Convex Hull For Tool Interference Avoidance in 5-Axis CNC Machining, In *Proceedings of the 1996 ASME Design Technical Conferences.*

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

Lozano-Perez, T., 1983, Spatial Planning: A configuration Space Approach, *IEEE Transactions on Computers*, C-32(2):108–120.

Martin, R. and Stephenson, P., 1990, Sweeping of Three Dimensional Objects, *Computer Aided Design*, 22(4):223–233.

Menon, J., Marisa, R. J., and Zagajac, J., 1994, More Powerful Solid Modeling through Ray Representations, *IEEE Computer Graphics and Applications*, 14(3):22–35.

Menon, J. P. and Robinson, D. M., 1993, Advanced NC Verification via Massively Parallel Raycasting, *Manufacturing Review*, 6(2):141–154.

Pasko, A. and Sourin, A., 1996, Function representation for sweeping by a moving solid, *IEEE Transactions on Visualization and Computer Graphics*.

Phillips, J., 1984, *Freedom in Machinery*, Cambridge University Press.

Rabbitz, R., 1994, Fast Collision Detection of Moving Convex Polyhedra, In *Graphics Gems IV*, pp. 83–109. Academic Press, Boston.

Red, E., 1984, Minimum Distances for Robot Task Simulation, *Robotica*, 1:231–238.

Requicha, A., 1980, Representations for rigid solids: Theory, Methods and Systems, *Computing Surveys*, 12(4):437–463.

Requicha, A. A. G. and Voelcker, H. B., 1985, Boolean Operations in Solid Modeling: Boundary Evaluation and Merging Algorithms, *Proceedings of the IEEE*, 73(1):30–44.

Sambandan, K., 1990, *Geometry Generated by Sweeps of Polygons and Polyhedra*, PhD thesis, Cornell University.

Schroeder, W., Lorensen, W., and Linthicum, S., 1994, Implicit Modeling of Swept Surfaces and Volumes, In *Proceedings of the 1994 IEEE Visualization Conference*, pp. 40–45.

Shapiro, V., 1997, Maintenance of geometric representations through space decompositions, *International Journal of Computational Geometry and Applications*, 7(4):383–418.

Shigley, J. and Uicker, J., 1995, *Theory of Machines and Mechanisms*, McGraw-Hill, 2nd ed.

Shimada, K., Okano, A., and Kawabe, S., 1988, A Collision detection algorithm for geometric simulation, In *Proceedings of the USA-Japan Symposium on Flexible Automation - Crossing Bridges: Advances in Flexible Automation and Robotics*.

Snyder, J. M. and Kajiya, J. T., 1992, Generative modeling: A symbolic system for geometric modeling, In *Computer Graphics (SIGGRAPH '92 Proceedings)*, volume 26, pp. 369–378.

Sourin, A. and Pasko, A., 1995, Function Representation for Sweeping by a Moving Solid, In *Third International Symposium on Solid Modeling and Applications*, pp. 383–392.

Uicker, J., 1997, Matrix Methods in the design and Analysis of Mechanisms, ME751, Lecture notes, University of Wisconsin, Madison.

Ulrich, K. and Seering, W., 1990, Function Sharing in Mechanical Design, *Design Studies*, 11(4):223 – 234.

Vossler, D. L., 1985, Sweep-to-CSG Conversion Using Pattern Recognition Techniques, *IEEE Computer Graphics and Applications*, 5(8):61–68.

Wang, W. and Wang, K., 1986, Geometric Modeling for Swept volume of Moving Solids, *IEEE Computer Graphics Applications*, 6(12):8–17.