Simplification and Abstraction of Kinematic Behaviors: Hierarchical Reasoning in Mechanical Devices *

Leo Joskowicz

IBM T. J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598
E-mail: josko@ibm.com

Abstract

Efficient problem-solving in any physical domain requires two main capabilities: the ability to ignore detail by reasoning at different levels of resolution (abstraction), and the ability to ignore irrelevant information by incorporating constraints and assumptions (simplification). Existing analysis methods for mechanical devices derive qualitative descriptions of a mechanism's kinematic behavior from the shape and the initial position of its parts. Although qualitative, these descriptions are sometimes too detailed and exceedingly complex, making the automation of common analysis and design tasks difficult, or even impossible. This paper presents a set of operators to simplify and abstract kinematic descriptions derived from configuration spaces. These operators define the hierarchy of resolutions necessary to effectively automate common reasoning tasks about mechanical devices. We show how mechanism comparison – determining when two mechanisms are kinematically equivalent – can only be done with simplification and abstraction operators.

(Also Submitted to IJCAI-89)

Word Count: 4200

*I wish to thank Sanjaya Addanki and Ernest Davis for many helpful comments on earlier versions of this work.

1 Introduction

Recent work on the analysis of mechanical devices has introduced methods to derive descriptions of a mechanism's kinematic behavior from the shape and the initial position of its parts [Faltings, 87a; 87b], [Forbus *et al.*, 87], [Nielsen, 88], [Joskowicz, 87, 88; 89a]. Although qualitative, these descriptions are sometimes too detailed and exceedingly complex, making the automation of common analysis and design tasks difficult, or even impossible. The complexity of the behavioral descriptions appears at two levels: (1) in the (local) kinematic pair descriptions, as a result of the objects' shape complexity, and (2) in the overall (global) mechanism descriptions, as a result of the combinatorial complexity of possible object positions.

Local descriptions are derived from partitions of two-dimensional configuration spaces[1] defined by the objects' degrees of freedom along fixed axes. The configuration space of a kinematic pair is computed by analyzing all pairwise contacts between object features (vertices, edges, and arcs). Each feature contact defines a half-space, bounded by a one-dimensional curve, (called a *configuration space boundary* or CS-boundary for short) that separates free and forbidden placements. The intersection of these half spaces defines the components of the configuration space, which are then partitioned into regions. Regions are defined by monotone CS-boundary segments and reflect the qualitatively different behaviors of the pair. In general, there can be as many regions as there are possible contacts; when this is the case, the resulting behavioral description can turn out to be too detailed.

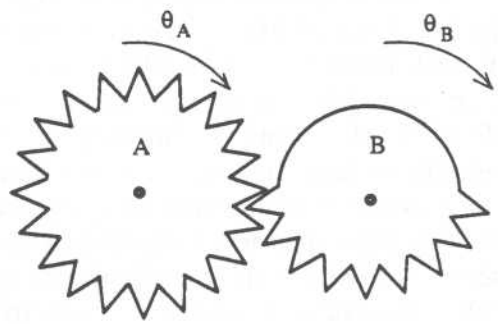

Figure 1: The Half-Gear Pair

As an example, consider the the half-gear pair in Figure 1, consisting of a 20-teeth gear and a 9-teeth half gear of equal diameter. Their configuration space is shown in Figure 2(a): dark areas correspond to forbidden object positions, and solid lines correspond to CS-boundaries. Figure 2(b) shows a detailed view of the CS-boundaries and the partition into regions[2]. Region R_0 corresponds to positions

[1] The configuration space of a mechanism defines the set of *free placements* (position and orientations) of objects in a mechanism so that no two objects overlap [Lozano-Pérez, 83].

[2] Both figures adapted from [Faltings, 87]; for clarity, only half of the "strips" are shown in (a).

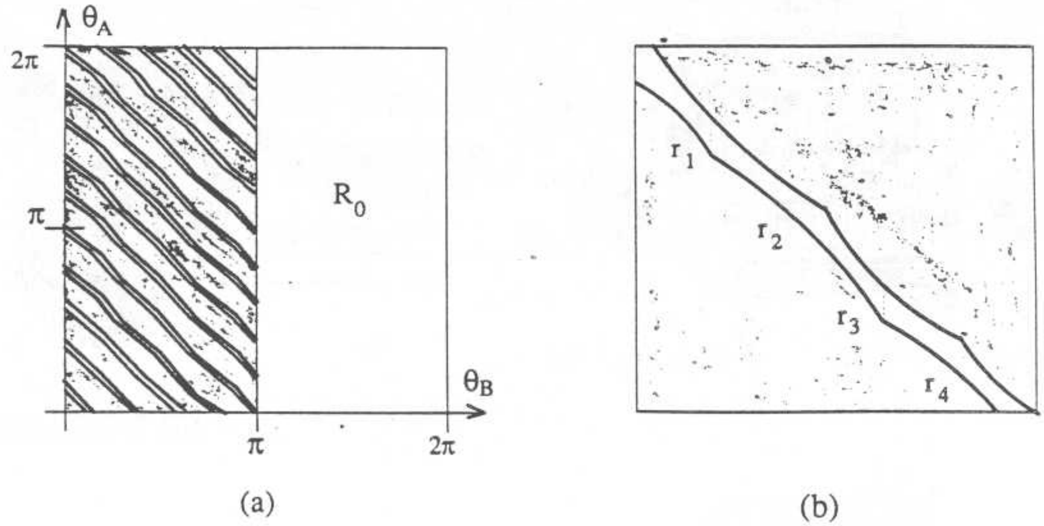

Figure 2: The Configuration Space of the Half Gear Pair

in which the rotations of A and B are independent. Regions r_i in the interval $\theta_B \in [0, \pi]_{mod 2\pi}$ correspond to positions in which A and B are meshed. The small interval between CS-boundaries in each region corresponds to gear interplay (backlash), and their fragmented nature indicates slight variations in the motion transmission ratio. Since there are two regions for each pairwise tooth contact, there are $20 \times 9 \times 2 = 360$ r_i regions. Although exact, this kinematic description is too detailed for a CAD system designing a complete gearbox.

Global behaviors are obtained by composing all local descriptions. For fixed axes mechanisms (mechanisms in which all parts move along straight axes that are fixed in space), the composition is done using a small set of composition rules [Joskowicz, 87]. In the worst case, the composition results in the cross-product of all local regions for every pair. As a consequence, descriptions of mechanisms with several degrees of freedom will contain many positional distinctions that, although feasible, require the application of input motions to all parts. In addition, these distinctions are sometimes irrelevant.

As an example, consider the cylinder lock in Figure 3, consisting of a main cylinder C rotationally mounted on a fixed frame F. Five pairs of pins, (P_i, Q_i), of different lengths, are mounted inside five aligned cylindrical holes in the cylinder and the frame. The pins are kept in contact by springs and can only translate along the axes of the holes; their role is to prevent the rotation of the cylinder. When the appropriate key is inserted, the pins are raised so that the top of the lower pins and the bottom of the upper pins coincide exactly with the outer surface of the cylinder. In this case, no obstacle prevents the rotation of the cylinder: a rotation of the key will cause the cylinder to rotate with it. The rotation of the cylinder causes a tumbler (not shown in the figure) to translate, thereby locking the door by preventing it from rotating around its axis. The analysis distinguishes three qualitatively different positions of the pins: (1) the upper and lower pins are

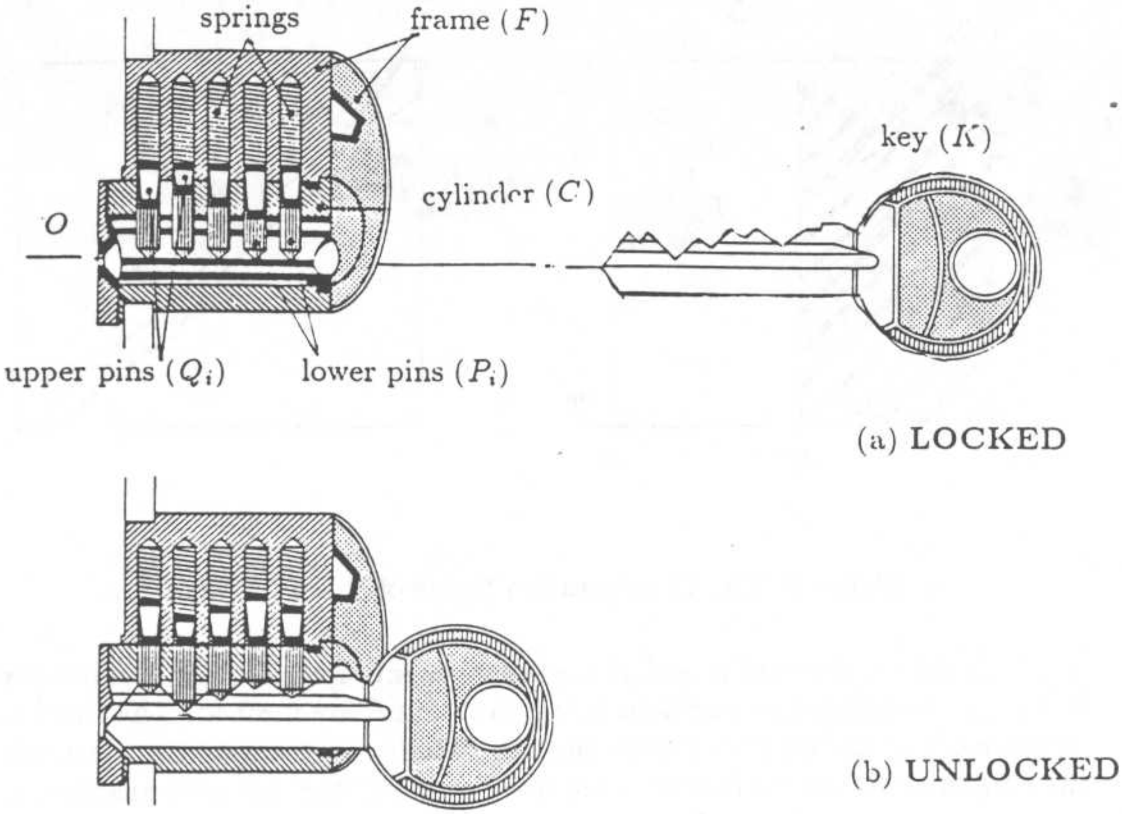

Figure 3: The Cylinder Lock

in contact with the cylinder, preventing it from rotating; (2) the upper and lower pins are in contact with the fixed frame, preventing the cylinder from rotating; (3) the top of the lower pin and the bottom of the lower pin coincide with the top of the cylinder. The cylinder is then free to rotate, in which case the pins cannot translate. The key/cylinder pair is described by two characteristic positions: (1) the key and the cylinder are not in contact; the rotation of the cylinder is independent from the rotation and/or translation of the key; (2) the key is inside the cylinder; the key is free to translate inside the cylinder, but the rotations of the key and cylinder are directly dependent. In the global description, consisting of 963 qualitatively different configurations [Joskowicz, 89b], there are $3^5 = 243$ configurations corresponding to all the possible combinations of pin positions when the key is outside the cylinder. To reach these positions, however, input motions must be applied directly to the pins! If we assume that input motions can only come from the key, all these positions become unreachable. Another source of detail comes from the positions of the key inside the cylinder. As the key is inserted, the pins follow the upper contour of the key, thereby changing characteristic positions. If we are only interested in the behavior of the key and the cylinder, all the distinctions introduced by these different pin positions become irrelevant.

The previous two examples illustrate the need for simplifying and abstracting kinematic behaviors. Simplification consists of incorporating additional information – in the form of assumptions and constraints – that rules out unreachable behaviors and discards others by considering them irrelevant to the desired final description. Abstraction consists of describing behavior at a lower level of resolution by ignoring certain types of behavioral differences. Note that whereas simplification produces descriptions at the same level of resolution, abstraction produces coarser descriptions. In this paper, we introduce a set of simplification and abstraction operators that produce multiple-level resolution descriptions of kinematic behavior; without these operators, basic tasks such as mechanism comparison cannot be realized.

2 Describing Kinematic Behavior

Objects in a fixed axes mechanism can only rotate or translate (or both) along axes that are fixed in space. We can thus classify all their possible motions in five categories: no motion, rotation around an axis, translation along an axis, independent rotation and translation along an axis, and related rotation and translation along an axis (helical motion). To each category, we associate a predicate indicating the type of motion and the axis along which the motion takes place. To describe the extent of the motion, we associate *motion parameters* (one for each degree of freedom along the axis) bound by intervals that define the legal range of motion. The type of motion, together with the motion parameters and their intervals constitute a complete description of an object's kinematic behavior. We call such a description a *possible motions label.* The five possible motion labels for object A along axis O are: $fixed(A, O, p)$, $p\_rotation(A, O, \theta)$, $p\_translation(A, O, X)$, $p\_cylinder(A, O, X, \theta)$, and $p\_helical(A, O, X, \theta)$.

Relationships between object motions, indicating how objects constrain each other's motion through contact, are specified by motion parameter relations. Possible behaviors of a mechanism are described by assigning one possible motion label to each object, and specifying the dependencies between motion parameters. Such a description is called a *possible motions region* of the mechanism's behavior[3]. A complete account of a mechanism's kinematic behavior is described with a *region diagram.* A region diagram is an undirected graph whose nodes represent possible motion regions and whose edges represent possible transitions between regions.

In order to distinguish between qualitatively different behaviors, all regions in the diagram must be qualitatively different. Two possible motions regions, R_i and R_j, are qualitatively different iff at least one of the following holds: (1) the motion type of at least one object is different in R_i and R_j; (2) the motion parameter intervals defining R_i and R_j cannot be merged into continuous intervals forming a new region $R_k = R_i \cup R_j$; (3) motion parameter relations in R_i and R_j are not identical; (4) motion parameter relations in R_i and R_j are identical but at least one of these relations is monotonically increasing one region and monotonically decreasing in the other. Region diagrams constitute a symbolic, qualitative description of all the

[3]These descriptions are called regions because they correspond to *regions* of the mechanism's configuration space

possible kinematic behaviors of a mechanism.

3 Simplification and Abstraction Operators

The goal of simplifying and abstracting behavioral descriptions is to produce coarser and more constrained descriptions of a mechanism's behavior. To keep the resulting description consistent, these operations must be sound and complete. Soundness guarantees that no new behaviors are introduced. Completeness guarantees that no possible behaviors were lost in the simplification/abstraction process. To make the computation of the global behavior efficient, operators must be compositional. Compositionality guarantees that simplifications/abstractions can be applied to local descriptions *before* composing them without altering the final description. In the following, we briefly describe operators to simplify and abstract behaviors. For detailed algorithms and complexity analyses, see [Joskowicz, 89b].

Local Operators:

Local operators are applied to two-dimensional configuration spaces defined by n monotone CS-boundary segments. We assume that the configuration space to which they are applied has already been computed. There are ten local operators:

Kinematic Constraint Simplifications: These operators incorporate user-specified constraints indicating the object to which input motions can be applied, the type of these motions, and their range (Input-Part, Input-Type, and Input-Range, respectively). Their effect is to rule out possible behaviors that become unreachable as a consequence of this constraining information. Input-Part transforms a two-dimensional region into a set of more constrained one and two-dimensional regions. The new regions indicate that the static object can only change its position when the moving object is in contact with it. Input-Type reduces the number of pairwise configuration spaces to be analyzed (one for every pair of fixed axes). Input-Range rules out behaviors by deleting regions that are completely outside the specified motion range, and restricting those who lie on the boundary. These simplifications can all be implemented in $O(n)$.

Non-Kinematic Constraint Simplifications: These operators incorporate physical constraints resulting from the action of gravity, the effect of springs, or friction. Three kinematic constraints are used to model these phenomena: (1) a constant contact relation between two objects; (2) a preferred (or default) position of an object when it is not subject to contact constraints, and; (3) conditions on the motion relation parameters, indicating when motion transmission is impossible. The first two constraints model the effects of springs and gravity. An object subject to a force remains in contact with its neighboring objects in the direction fo the force. Thus, all positions in which the two objects are not in contact can be ruled out. When no contact occurs, the object moves in the direction of the force until it reaches a stable position. To avoid introducing time information, we assume that motions due to forces are infinitely faster than input motions. The third constraint models friction. Motion can only be transmitted through feature contacts whose corresponding CS-boundary tangent is greater than a certain coefficient μ, regardless of the force applied to the object. This constraint rules out possible transitions between regions and reduces region intervals. All these operators can be implemented in $O(n)$

Gap-Closing Simplification: This operator merges CS-boundaries defining narrow "channels" of free placements. Such channels reflect the effects of backlash (Figure 2(b)) and tolerancing errors. The small positional variations inside the channels reflect negligible behavioral variations that can usually be ignored. Gap-Closing reduces a two-dimensional region defined by two CS-boundaries to a one-dimensional region defined by one CS-boundary. In this new region object motions are tightly coupled – one object moves iff the other does. The operator examines every region for a possible boundary merge, and can be implemented in $O(n)$.

Linearization Abstraction: This operator replaces CS-boundaries by piecewise linear approximations. Every CS-boundary is divided into monotonically continuous segments, which are then replaced by a set of lines. The new linear CS-boundary might intersect other CS-boundaries, thereby producing a topologically inconsistent abstraction. This is avoided by further subdividing the CS-boundary into smaller segments. Assuming that each segment is broken only a constant number of times. the linearization can be efficiently implemented using a line intersection algorithm in time $O(n \log n)$.

Qualitative Abstraction: This operator merges contiguous monotone CS-boundaries and creates *qualitative CS-boundaries.* These boundaries indicate the qualitative motion parameter relation (monotonically increasing or decreasing) in the new regions. The purpose of this operator is to abstract the motion parameter relations and hide the details of feature contacts that produce fragmented behavioral descriptions. Region merging also involves testing for CS-boundary intersections, and thus can be implemented in $O(n \log n)$. Linearization and Qualitative abstraction have been used for the design of new object shapes in kinematic pairs [Joskowicz and Addanki, 88].

Behavior Parametrization: This operator does not simplify or abstract behaviors, but exposes similarities between them. It compares two behaviors (described as two possible motions regions) and attempts to produce a common description of them by parametrizing their possible motion labels. Possible motions regions have three candidates for parametrization: the region intervals, the axes of motions. and the motion parameters' relations. Two regions can be parameterized in their intervals when (1) their motion type is identical, and defined along the same axis: (2) the relations between motion parameters are identical; and (3) their interval ranges are different but proportionally scaled; a parameter is introduced to reflect this scaling. Two regions can be parameterized in their axes when the axes of motion are not required to be identical. Finally, two regions can be parametrized in their motion parameter's relations when their relations have one of the following similarities: Relation parametrization consists of comparing two motion parameter relations and finding a parametric similarity between them: linear ($y \leq f(x)$ and $y \leq f(x) + c$), scalar ($y \leq f(x)$ and $y \leq c.f(x)$), phase ($y \leq f(x)$ and $y \leq f(x+c)$), or any combination of them. This operation takes constant time.

Periodicity Abstraction: This operator finds patterns of repetitive behavior. It creates a common parametrized description of similar behaviors and reduces the region diagram by identifying parametrically isomorphic subgraphs. Two subgraphs are parametrically isomorphic iff they are topologically equivalent and there is a one-to-one parametric matching between regions. Although graph isomorphism can be tested efficiently – region diagrams are graphs with a planar embedding – subgraph isomorphism, even for planar graphs, is in NP-Complete, and is thus computationally expensive.

The order of application of local operators is as follows: Simplification operators (both Kinematic and Non Kinematic) are applied first, because they constrain possible behaviors without changing the level of resolution. Abstraction operators are then applied in increasing order of coarseness: first Linearization, then Qualitative abstraction. Gap-Closing can be applied both before or after each one of the previous two simplifications. Finally, once the appropriate level of abstraction is obtained, Periodicity abstraction is applied.

As an example, consider the effects of local operators on the half-gear pair's configuration space of Figure 2. We assume that input motions are only applied to gear B and that the initial placement is $\theta_A = \theta_B = 0$. The application of the Input-Part simplification reduces R_0 to a set of disjoint two-dimensional strips bounded by the linear relations $\theta_A \geq \theta_i$ and $\theta_A \leq \theta_i + \epsilon$ in the interval $\theta_B \in [\pi, 2\pi]$. Only two strips, $\theta_i = 0$ and π, are reachable from the initial placement. As a consequence, in the interval $\theta_B \in [0, \pi]$, only the regions connected to these two strips are reachable. All other regions are unreachable and are thus discarded. Linearization and Qualitative abstraction merge the remaining CS-boundaries in the interval $\theta_B \in [0, \pi]$ into two regions. each defined by two parallel lines. Gap-Closing merges the parallel lines in these regions as well as the two strips in the interval $\theta_B \in (\pi, 2\pi]$. The successive effects of these operators on the configuration space are shown in Figure 4. Periodicity abstraction identifies two parametrically isomorphic subgraphs, (R_0, R_1) and (R_2, R_3)[4]. They are commonly described with a parameter i, where $i = 0$ describes (R_0, R_1) and $i = 1$ describes (R_2, R_3):

R: $p\_rotation(A, O_A, \theta_A),\ p\_rotation(B, O_B, \theta_B)$
$$\theta_A \in [i.\pi,\ (i+1).\pi]_{mod2\pi},\quad \theta_B \in [0, \pi]_{mod2\pi},\quad \theta_A = -\theta_B$$

R': $fixed(A, O_A, \theta_A),\ p\_rotation(B, O_B, \theta_B)$
$$\theta_A = i.\pi_{mod2\pi},\quad \theta_B \in [\pi,\ 2\pi]_{mod2\pi}$$

$Transition(R, R')$: $\theta_A = (i+1).\pi,\quad \theta_B = \pi$

[4]The parameterization of (R_1, R_2) and (R_3, R_0) is equally valid.

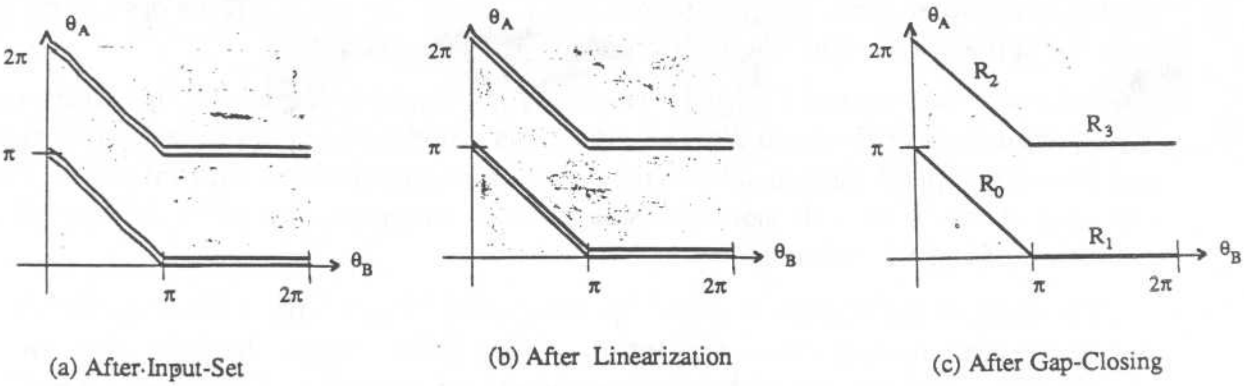

Figure 4: Configuration Space after Simplification and Abstraction

Global Operators

Global operators are applied after local operators have been applied to every kinematic pair description. There are four global operators, whose complexity is proportional to the number of regions in the global diagram:

Input-Set Simplification: This operator is a generalization of the Input-Part, Input-Motions, and Input-Range simplifications for more than two objects. The motion restrictions specify a subset of objects to which input motions can be applied, the type of these motions and their range. This operator further discards unreachable regions and restricts possible motions in the remaining regions. Input-Set simplification can be applied *while* composing local region diagrams, thereby ruling out potential regions without computing them.

Relevance-Set Simplification: This operator filters out behavioral distinctions created by unintersting objects. The user indicates a set of objects, called the relevance set, whose specific behaviors he is interested in (the most common relevance set is the input/output parts set). This operator projects each region into a new region from which the possible motion labels and relations of uninteresting objects are removed. As a consequence, contiguous regions whose difference lied solely in the behavior of uninteresting objects become qualitatively similar and can thus be merged. This operator can only be applied *after* the composition of local region diagrams.

Region-Difference Abstraction: This operator produces coarser behavioral descriptions by relaxing the criteria that make two behaviors (regions) qualitatively different. We consider two relaxations: (1) motion parameter relations need not be identical, but rather *monotonically identical*, i.e., they both specify the same relation ($\leq$, $\geq$, or $=$) and their functions are both simultaneously increasing or decreasing in the given interval. For example, $y \leq x$ and $y \leq x^2$ are monotonically identical in the interval $x \in [0, 1]$ and monotonically different in the interval $[-1, 0]$; (2) motion predicates need not be all identical. For example, to distinguish between objects that move and do not move, regardless of the type of motion, we define two categories: "no-motion", consisting of the motion type *fixed*, and "motion", consisting of all other motion types (*p_rotation*, *p_translation*, etc.). This operator can only be applied *after* the composition of local region diagrams.

Behavior Parametrization and Periodicity Abstraction: These two operators are direct extensions of the local diagram operators. Regions are parameterized by taking into account all motion labels, motion parameter relations, and intervals. Periodicity abstraction finds isomorphisms between subgraphs by using the extended definition of region similarity.

The order of application of global operators is as follows: Input-Set simplification is applied first during the computation of the global region diagram because it discards potential regions without computing them explicitly. Then, Relevance-Set is applied to focus on the behaviors of relevant objects. Next, Region-Difference is applied to create a coarser behavioral description; if it does not reduce the size of the region diagram, this type of abstractions are unnecessary. Periodicity abstraction is applied at the end.

As an example, consider the effects of global operators on the description the cylinder lock of Figure 3. We assume that input motions can only come from the key, and are along axis O. The application of the Input-Set simplification to the local region diagram of the pairs key/lower pins, (K, P_i), discards all the regions in which K and P_i are not in contact, except for the region containing the initial placement. This eliminates $3^5 - 1$ potential global regions which correspond to all the combinations of pin positions when the key is outside the cylinder. This operator also eliminates all the regions where pins not in contact with the key take different positions as the key is inserted; there are $3 \times (3^4 + 3^3 + 3^2 + 3^1) = 360$ such potential global regions. In total, 602 potential global regions are discarded without even computing them. Assuming that the relevance set is formed by the key (input) and the cylinder (output), all the regions specifying different placements of the pins, but no difference in the angular position of the cylinder or the key can be merged. The resulting projected region diagram consists of three regions:

OUT The key and the cylinder are not in contact. The key can both rotate and translate along O, and the cylinder cannot rotate.

IN The key is inside the cylinder. The key translate along O, but cannot rotate. The cylinder cannot rotate either.

UNLOCK The key is at the end of the cylinder. The key can rotate together with the cylinder, but it cannot translate.

No further abstraction and/or simplification is applicable at this point.

4 Comparing Two Mechanisms

Mechanism comparison – determining if two mechanisms are kinematically equivalent – is an essential constituent of the design process. It is used to evaluate alternative solutions and determine if parts can be substituted. Directly comparing the mechanisms' region diagrams is too restricting: to be considered equivalent, both mechanisms must have the same number of parts moving along the same axes by in the same intervals and by the same amount. We want, instead, to find the necessary assumptions and the appropriate resolution for which two behavioral descriptions become identical. For this purpose, we use simplification and abstraction operators.

The algorithm proceeds by successively applying the operators to both region diagrams in the order indicated in Section 3. After each step, the size of the region diagrams are compared. If they have the same number of regions, the algorithm tries to establish a parametrized match between the regions, using abstractions when necessary. When the two diagrams match the algorithm stops. The output is common description for both region diagrams, and the list of operators applied to achieve the match. This list, together with the region parametrization, establishes the smallest set of conditions necessary to determine when two mechanisms are equivalent (see [Joskowicz, 89b] for details).

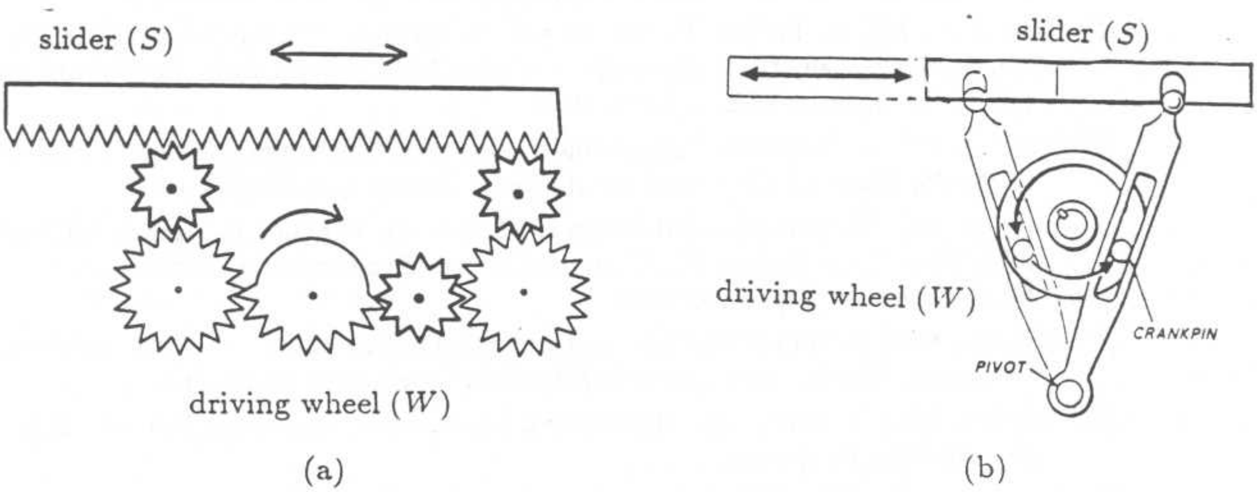

Figure 5: Two Crank Mechanisms

As an example, consider the two crank mechanisms in Figure 5. They both transform a continuous rotation of the driving wheel W (Input-Set) into a reciprocating translation of the slider S. Assuming that the only behaviors of interest are those of the wheel and the slider (Relevance-Set), both mechanisms are equivalent under the following three assumptions: (1) motion parameter relations are monotonically identical (Linearization and Qualitative Abstraction); (2) the slider's displacement is within the same range (Behavior Parametrization); (3) backlash is ignored (Gap-Closing Simplification).

5 Conclusion

Future Intelligent CAD systems must be able to problem-solve using qualitative descriptions at different levels of resolution. Despite their generality, existing methods for qualitative reasoning at multiple levels of resolution, [Falkenhainer and Forbus, 88] and [Murthy, 88], are inappropriate to deal with geometric shapes and kinematic behaviors. This paper presented a set of operators to simplify and abstract kinematic descriptions derived from configuration spaces. These operators define the hierarchy of resolutions necessary to effectively automate common reasoning tasks about mechanical devices. We showed that mechanism comparison is an example of a task requiring simplification and abstraction operators. Ongoing research has shown that teleological reasoning – deriving function form behavior is feasible using this approach. The implementation of the operators is planned for the near future.

References

[Falkenhainer and Forbus, 88] B. Falkenhainer and K. Forbus, "Setting up Large-Scale Qualitative Models" *Proceedings of the 7th National Conference on Artificial Intelligence*, St. Paul, Minnesota, USA, 1988.

[Faltings, 87a] B. Faltings, "Qualitative Place Vocabularies for Mechanisms in Configuration Space", Phd. Diss, Report UIUCDCS-R-87-1360, University of Illinois, 1987.

[Faltings, 87b] B. Faltings, "Qualitative Kinematics in Mechanisms" *Proceedings of 10th International Joint Conference on Artificial Intelligence*, Milano, Italy, 1987.

[Forbus *et al.*, 87] K. Forbus, P. Nielsen and B. Faltings, "The Inferential Structure of Qualitative Kinematics", *Proceedings of 10th International Joint Conference on Artificial Intelligence*, Milano, Italy, 1987.

[Joskowicz, 87] L. Joskowicz, "Shape and Function in Mechanical Devices", *Proceedings of the 6th National Conference on Artificial Intelligence*, Seattle, 1987.

[Joskowicz, 88] "Reasoning about Shape and Kinematic Function in Mechanical Devices" PhD. Diss, Tech. Report 402, Computer Science Department, Courant Institute, New York University, September 1988.

[Joskowicz, 89a] L. Joskowicz, "Reasoning about the Kinematics of Mechanical Devices", to appear, *International Journal of Artificial Intelligence in Engineering*, 1989.

[Joskowicz, 89b] L. Joskowicz, "Hierarchical Reasoning in Mechanical Devices" IBM Technical Report, to appear.

[Joskowicz and Addanki, 88] L. Joskowicz and S. Addanki, "From Kinematics to Shape: An Approach to Innovative Design", *Proceedings of the 7th National Conference on Artificial Intelligence*, St. Paul, Minnesota, USA, 1988.

[Lozano-Pérez, 83] l T. Lozano-Pérez, "Spatial Planning: A Configuration Space Approach", *IEEE Transactions on Computers*, Vol. **C-32**, No. 2, 1983.

[Murthy, 88] S. Murthy, "Qualitative Reasoning at Multiple Levels of Resolution", *Proceedings of the 7th National Conference on Artificial Intelligence*, St. Paul, Minnesota, USA, 1988.

[Nielsen, 88] P. Nielsen, "A Qualitative Approach to Mechanical Constraint" *Proceedings of the 7th National Conference on Artificial Intelligence*, St. Paul, USA, 1988.