Understanding Linkages.

Conference Paper · January 1993

Source: DBLP

CITATIONS
14

READS
670

1 author:

Howard E. Shrobe
Massachusetts Institute of Technology
123 PUBLICATIONS **3,370** CITATIONS

SEE PROFILE

Understanding Linkages*

Howard E. Shrobe
Massachusetts Institute of Technology
NE43-839
Cambridge, MA 02139
hes@zermatt.lcs.mit.edu

Abstract

Mechanical linkages are used to transmit and transform motion. In this paper we investigate what it means for a program to "understand" a linkage. Our system extracts its understanding by analyzing the results of a numerical simulation of the mechanism, finding interesting qualitative features, looking for symbolic relationships between these features and conjecturing a causal relationship between them. Our system is capable of understanding a variety of mechanisms, producing explanations very much like those in standard texts.

1 Motivation

Mechanical linkages are used to transmit and transform motion. They are a subset of the class of "fixed topology mechanisms", those consisting of rigid bodies in constant contact with motion being transmitted through joints, gears and cams. In this paper we investigate how a system can "understand" a linkage, i.e. how it can

- Decompose the mechanism into understandable submechanisms.
- Explain how the behavior of the whole arises from that of the parts.
- Assign a purpose to each of the components.
- Enable redesign by highlighting what interactions lead to the desired behavior.

Although the techniques in this paper apply to the broader class of fixed topology mechanisms, the running example in this paper will consist of a linkage with a single degree of freedom.

*This paper describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. Support for the author's artificial intelligence research is provided by the Advanced Research Projects Agency of the Department of Defense under Office of Naval Research contract N00014-91-J-4038.

Figure 1 shows a six-bar linkage[1] functioning as a dwell mechanism[2] with its explanation reproduced from [1][3]. (We have highlighted parts of this explanation). This paper presents a system which can "understand" this linkage, producing an explanation like that of the figure.

Several observations about the explanation of figure 1 are worth emphasizing:

- The explanation is compositional. The behavior of the whole is derived by first decomposing the mechanism into modules and by then composing the behaviors of the modules into an aggregate behavior: the device consists of "four-bar linkage ABCD" driving the pair of links 4 and 5.[4]However, the decomposition stops before reaching the primitive elements (joints and links). The explanation does not attempt to provide a mechanistic explanation of how the shape of the coupler curve of the four bar linkage ABCD is related to the sizes of its links.
- A crucial component of the explanation is a characterization of the qualitative shape of curves traced by "interesting points" in the mechanism: "point E (of link 2) ... traces a path of which portion a-a approximates a *circular arc* of radius FE with its center at point F".
- Although the explanation does not emphasize local causal propagations of the type made popular in [2; 4; 11; 12], it does have a causal flavor at a relatively high level of abstraction: "the shape of the coupler curve causes link 5 to have a dwell".

The system described in this paper is capable of producing such an explanation. Our approach is as follows:

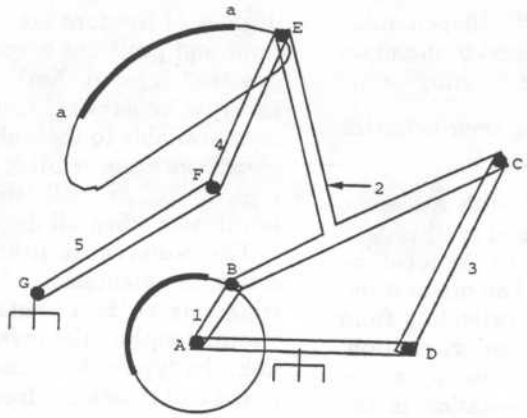
1. We numerically simulate the mechanism at a single time step.
2. The simulator is driven by geometric constraints. While satisfying these constraints, the simulator

[1] For those not familiar with linkages, we note that the set of links 1,2, and 3 together with the fixed frame, is a "four-bar linkage" (with joints A,B,C and D) and that the pair of links 4 and 5 (with joints F and G) is a "dyad". Link 2 is the "coupler" of the four-bar linkage; since point E is on the coupler, the curve it traces is called a "coupler curve". Four bar linkages are extremely flexible driving mechanisms; they can create a large number of coupler curves exhibiting a broad variety of shapes. The shape of the curve is a function of the (relative) sizes of the links and the position on the coupler link used to trace the curve.

[2] A dwell mechanism is one in which some part moves (in this case oscillates) most of the time, but for some period of time stands still (i.e. dwells).

[3] In this picture, the links are draw as bars, except that link 2 has a long finger projecting from it to point E making it look like an inverted T. Circles are used to indicate the joints between the links. The "ground" symbols are used to indicate that link AB is rigidly connected to the fixed frame and that joint G connects link 5 to the fixed frame.

[4] Such a pair of links is called a dyad.



The lengths of the links comply with the conditions: BC = 2 AB, DC = 5.2AC, EC = 3.6AB, EF = 3.6AB, GF = 11.4AB, AD = 6AB, GD = 8.4AB and AG = 11AB. Link 4 is connected by turning pairs E and F to link 2 of *four-bar linkage ABCD* and to link 5 which oscillates about fixed axis G.
When point B of crank 1 travels along the part of the circle indicated by a heavy continuous line, point E of connecting rod 2 *describes a path of which portion a-a approximates a circular arc* of radius FE with its center at point F. During this period link 5 almost ceases to oscillate, i.e. *it practically has a dwell.*

Figure 1: A Dwell Mechanism and Its Explanation

records its inferences as a "mechanism graph" showing how motion propagates from link to link through the joints of the mechanism.

3. The mechanism graph is "parsed" into a more structured form which decomposes the system into driving and driven modules. To the extent possible the parsed graph consists of standard building blocks (e.g. four-bar linkages, dyads). The system knows which parameters of the standard building blocks are significant and these are identified as *important parameters.* Also the coupling points between the driving and driven modules are identified as *important parameters.*

   A complete simulation of the linkage is run, stepping the mechanism through its full range of positions (in our example this amounts to spinning link 1 through a full 360 degrees and for each step determining the positions and orientations of all the remaining components). During this simulation, the values of all *important parameters* (including the trajectories of the points connecting driving and driven modules) are recorded.

4. The shapes of the captured curves are analyzed and qualitative features extracted.

5. Qualitative relationships between these features are derived and accounted for by geometric reasoning.

Section 2 describes the simulator and how it supports the rest of this process. Section 3 then examines the process of mechanism extraction and section 4 describes curve characterization. Section 5 show how these facilities work together to construct an explanation of the mechanism. Section 6 discusses to what degree the interpretation produced is an adequate "understanding" of the mechanism. Finally, in section 7 we compare our work with other work on understanding mechanisms.

2 The Simulator

Our simulator is based on Kramer's TLA [10]. However, since our work (at least for now) only involves planar mechanisms we have simplified TLA to a 2-D simulator. Also we have extended the geometric solution techniques to handle gears and cams as well as pure linkages.

2.1 Basic Object Types

The simulator is at its core a geometric constraint engine. This engine reasons about the following physical objects:

- **Links**: These are rigid bodies connected by joints. All links are assumed to be aligned in parallel planes. Each link has its own local coordinate system. Each link also has a transformation matrix mapping its coordinate system into the global coordinate system. (We will often refer to the global coordinate system as the "fixed frame").
- **Joints**: A joint is a fixed connection between two links which couples their motion. We handle the following joint types (see figure 2):
  1. **Revolute**: The two links are connected at a single point; they rotate relative to each other about this point. A hinge is familiar example. All the joints in our example are revolute joints; these are sometimes called "turning pairs".
  2. **Pin in Slot**: A round "finger" from the first link slides in a guide path in the second link. The first link can translate along the direction of the slot; it can rotate relative to the second link as well. The guide track of a folding door is an example.
  3. **Prismatic**: The first link slides along the second link, but is not free to rotate relative to it. A piston in its cylinder is a familiar example.
  4. **Gears**: The two links are spur gears coupled by the meshing of their teeth. This includes planetary as well as fixed gears.

5. **Cams**: One link is an irregularly shaped rotational device; the other is constrained to maintain contact with the perimeter of the rotating cam.

Links and joints are modeled by reducing their behavior to the following computational constructs:

- **Markers**: Each marker is associated with a specific link (although each link may have several markers). A marker has two components specified in the local coordinate system of its link: a point and an orientation. A marker can be thought of as a line extending from the point in the direction specified by the orientation. The simulator may restrict a marker to occupy a specific position or to have a specific orientation in the global coordinate system. If this has occured we say that the marker has *invariant* position or orientation.
- **Constraints**: Constraints are the mechanism used to build a computational model of Joints. Each joint is modeled as a bundle of constraints. A constraint is imposed between two links by relating two markers, one from each link. Our simulator has the following constraint types:
  1. **Coincident**: The two markers are forced to be at the same location in the global coordinate system. A revolute joint is modeled as a single coincident constraint.
  2. **Inline**: The location of the first marker is on the line described by the second marker. A Pin-In-Slot joint is modeled by a single Inline constraint.
  3. **Cooriented**: The two markers' orientations are forced to be the same in the global coordinate system. A prismatic joint is modeled as a combination of a Cooriented and an Inline constraint.
  4. **Rotational Multiplication**: Used to model gears. The angular deflection of the first marker from its initial position is a constant (the gear ratio) times the deflection of the second marker from its initial position.
  5. **Perimeter Contact**: Used to model cams. The marker on the follower is constrained to be in contact with the perimeter of the Cam link.
- **Anchors**: An anchor is a distinguished type of marker attached to the global coordinate system rather than to a link. Constraints between anchors and markers on links are used to orient or set the position of a link in the global coordinate system. Input variables are supplied to the system as the position or orientation of an anchor; typically, the anchor controls the position or orientation of a link via a constraint to one of the link's markers.

2.2 The Constraint Engine

As in Kramer's TLA, the constraint engine solves the geometric constraints using local geometric techniques. These techniques take the form of "constraint" and "locus intersection" methods (described below). As the constraint engine runs, it monitors the degrees of freedom remaining to each link; it also records for each marker whether its global position and orientation are *invariant*. As the links' degrees of freedom are reduced and the markers' orientations and positions become invariant, the geometric methods are triggered. Each method moves or rotates a link to satisfy a constraint, further reducing the degrees of freedom available to the link; each method may also cause the global position or orientation of some marker to become invariant. This will trigger other methods. The process terminates when all degrees of freedom are removed.

The **constraint methods** are triggered when the location or orientation of a marker becomes invariant. Its triggering pattern contains the invariant marker, a constraint coupling the invariant marker to some marker on a link, the type of the constraint, and the degrees of translational and rotational freedom remaining to the link. When a constraint method is triggered, it translates or rotates (or both) the link to satisfy the constraint. In doing so it reduces the degrees of freedom available to the link; it also causes the global orientation or position of the marker on the link to become invariant. When a link is reduced to 0 degrees of rotational freedom, the orientation of every marker on it becomes *invariant* in the global coordinate system; when a link is reduced to 0 degrees of both rotational and translational freedom, the position of every marker on the link becomes *invariant* in the global coordinate system. Figure 3 shows a constraint method for the coincident constraint used to model a revolute joint.

Locus intersection methods are used after the constraint methods. When a link's degrees of freedom have been sufficiently reduced, the markers on the link are constrained to move in simple curves. For example, if a link has 0 degrees of translational freedom and 1 degree of rotational freedom, then every marker on the link (except the one about which the link rotates) is constrained to move in a circle. If two markers coupled by a constraint are both restricted to move in simple curves, then there are only a small number of locations that the markers can consistently occupy. For example, if two markers coupled by a coincident constraint are both restricted to move in circles, then the markers must be located at one of the two intersection points of the circles. In simulating the linkage of figure 1, driving link 1 is rotated into its desired position, fixing the position of B; this means that link 2 is allowed only to rotate about B. Similarly, the position of D is fixed, so link 3 may only rotate about D. C must, therefore, be at an intersection point of the circular paths allowed to the ends of links 2 and 3.[5][6]

2.3 Animating a Linkage

The motion of a linkage can be simulated by repeatedly incrementing the position or orientation of the driving link and allowing the constraint engine to determine the correct locations and orientations for the other links. A simple animation can be produced by showing successive snapshots.

The simulator can attach "probes" to any marker in the mechanism; these record the position and/or orientation

[5] Notice that locus intersection methods lead to ambiguous results, since two circles may intersect at more than one point; the simulator must chose between the geometrically allowable results using physical principles such as continuity of motion.

[6] When there are no further constraint method or locus intersection methods to be employed but the constraints have not been solved, then an iterative numerical solution technique is employed. To save space and maintain continuity of presentation we omit the details; the examples in this paper never require iterative techniques.

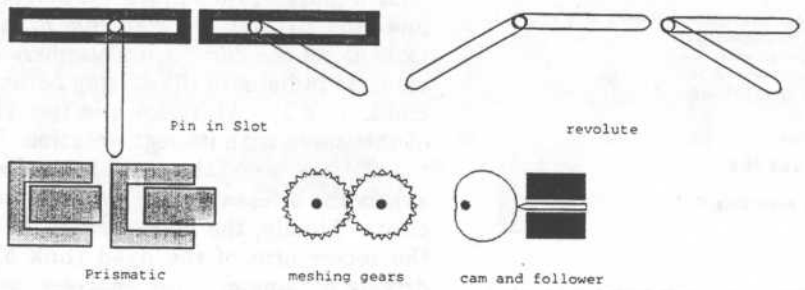


Figure 2: The Joints Modeled in the System

```
If   There is a coincident constraint between M-1 and M-2
     M-1 has invariant global position
     M-2 is on link L-2
     L-2 has 2 degrees of translational freedom
Then Measure the vector from M-2 to M-1
     Translate L-2 by this vector
     Reduce the translational degrees of freedom of L-2 to 0
     Constrain M-2 to have invariant global position
```

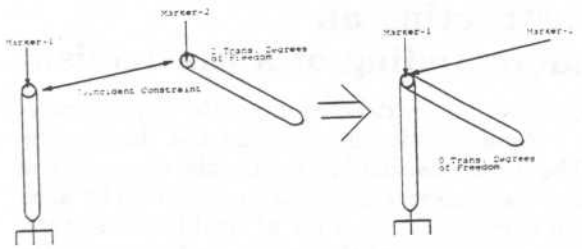


Figure 3: A Constraint Method for The Coincident Constraint

of the marker at each time step of the simulation. Thus, a probe captures the complete trajectory of a marker (e.g. the trajectory of point E in figure 1) or the history of values of some property of a marker (e.g. the global orientation of Marker G which is the same as the angular deflection of Link 5). This information is used later in analyzing the mechanism, see section 4.

2.4 Building the Mechanism Graph

The simulator can record its deductions using a truth maintenance facility. The simulator maintains in each link a special data structure called a *link-state-entry*. This contains the number of degrees of rotational and translational freedom available to the link at that point in the process of satisfying the geometric constraints.

When a constraint method entry updates the state of a link, it creates a new link-state-entry for the link. It also creates a *justification*. The antecedents of the justification are the current link-state-entries of the links coupled by the constraint; the consequent of the justification is the new link-state-entry for the affected link. The justification also records the constraint which caused the update. The link-state-entries may be thought of as the nodes of a truth maintenance graph and the justifications as directed arcs from the old link-state-entries to the new one. Locus intersection methods also create new link-state-entries and special justifications connecting them. The resulting graph records the steps of the process of satisfying the geometric constraints by moving (or rotating) then links while reducing their degrees of freedom.

This structure is similar to the "mechanism graph" of [2].

3 Mechanism Extraction

The first step in understanding the linkage mechanism shown in figure 1 is *mechanism extraction*, in which the assembly is decomposed into sub-assemblies and the relationship between driving and driven components is established. The input to this process is the mechanism graph produced by the simulator.

Mechanisms are identified as patterns of constraint solution within the mechanism graph; the patterns are identified by parsing rules like those shown in figure 4. The parsing rules build up a hierarchy of sub-modules. For the linkage of figure 1, the first rule characterizes link 1 as a crank; the third rule characterizes links 2 & 3 as a dyad. The second rule then notices that crank 1 drives the dyad formed by links 2 & 3. The last rule characterizes links 1,2 & 3, together with the fixed frame, as a four-bar linkage. Next, links 4 & 5 are characterized as another dyad which is driven by marker E on the coupler of the four bar linkage.

The rules shown in figure 4 cover most uses of four bar linkages. Pantographs, scotch-yokes, planetary-gear sets, slider-cranks, etc. are identified by similar sets of rules.

The structure produced by the parsing rules is used to identify points of interest in the mechanism. Part of what the system knows about each type of module is what points in the module are likely to play "interesting" roles in the larger mechanism. In particular, the system knows that the trajectory of coupler points of four-bar linkages are usually interesting, particularly if the coupler point drives another identifiable mechanism. Also the system knows that the deflection angle of the rocker arm of a driven dyad is interesting.

4 Characterizing Curve Shape

At this point, mechanism extraction has parsed the linkage into 2 sub-assemblies (a four-bar linkage and a dyad) and established a driver-driven relationship between them (the dyad is driven by a coupler point on the four-bar). However, the overall behavior of the mechanism depends on a specific feature of the shape of the curve traced by point E (it is a circular arc).

The next step of the analysis is to capture the relevant curves and to characterize their shapes. This is done by

```
If   Marker M-1 is on Link-1
     A-1 is an anchor and C-1 is a coincident constraint between M-1 and A-1
     The position of M-1 is determined by satisfying C-1
     A-2 is an anchor providing an input parameter
     C-2 is a cooriented constraint coupling A-2 and M-1
     The orientation of M-1 is determined by satisfying C-2
Then Link-1 is acting as a crank

If   Marker M-1 is on Link-1
     There is a constraint C between M-1 and M-2
     M-2 is on Link-2
     The position of M-2 is determined by satisfying C
Then Link-2 is driven by Link-1

If   M-0 and M-1 are on Link-1
     M-2 and M-3 are on Link-2
     The positions of M1 and M2 are determined by a circle-circle locus method
     A-2 is an anchor coupled by a coincident constraint C-2 to M-3
     The position of M-3 is determined by satisfying C-2
     M-4 is a marker coupled to M-0 by a coincident constraint C-1
     The position of M-0 is determined by satisfying C-1
Then Links 1 and 2 form a Dyad Dyad-1
     Link-1 is the coupler of Dyad-1
     Link-2 is the rocker of Dyad-1

If   Dyad-1 is a Dyad
     C-1 is the coupler of Dyad-1
     R-1 is the rocker of Dyad-1
     Crank-1 is acting as a crank
     C-1 is driven by Crank-1
Then C1, R-1 and Crank-1 form a four-bar linkage Four-bar-1
     Crank-1 is the crank of the four-bar
     C-1 of Dyad-1 is the coupler of Four-bar-1
     R-1 of Dyad-1 is the rocker of Four-bar-1
```

Figure 4: Rules for Parsing a Mechanism Graph

running a complete simulation of the linkage (i.e. by stepping the driving link through its complete range of motion); during this simulation, probes are attached to those points identified as interesting by the mechanism extraction: the coupler curve traced by point E and the angle of the rocker arm 3.

The following analyses are then performed:

- For each graph, the extrema of values are located (by finding the zero crossings of the first derivatives).
- For each trajectory traced, the system calculates the "Theta-S" representation which maps distance along the trajectory to the orientation at that point on the trajectory (in this representation a circular arc on the original curve appears as a straight line and a straight line in the original curve appears as a horizontal line).
- For each graph (other than traces of the trajectory of a point, but including the Theta-S curve for such a trajectory) a segmentation into linear approximations is performed using the "Split-Merge" technique.

  For each trajectory traced, the segmentation of the Theta-S representation is mapped back into a segmentation of the original curve. This segmentation approximates the original curve with linear and circular segments.
- For each trajectory traced, the system calculates the radius and centers of curvature at each point.
- For each trajectory traced, the system calculates the points of self intersection.
- For each graph, segments of constant value are located.
- Fourier transforms of graphs are calculated if there is reason to suspect that periodic motion is present.

Figure 5 shows the coupler curve of the dwell mechanism of Figure 1. The segmentation of the curve is indicated by "hatch marks"; the approximation of the curve by straight lines and circular arc segments is shown by dashed lines. Dots along the curve with numbers attached indicate the value, in radians, of the driving parameter (the angle of the crank, link 1). Also shown is the Theta-S representation of this curve with its segmentation. The horizontal axis is the distance along the curve normalized to 2Υ, the vertical axis is the orientation (in radians) at that position on the curve. Finally, the figure shows the angle (in radians) of the rocker arm of the dyad (link 5) plotted against the driving parameter same analyses, as is also shown in the figure.

Note that the coupler curve is well approximated by a circular arc (between about 1.1 and 4.0 radians of the driving parameter). Also note that the rocker arm's orientation is very nearly constant between about 1.0 and 4.1 radians of the driving parameter (the vertical scale of the graph is much larger than the horizontal scale, which obscures this fact).

5 Constructing an Understanding of a Mechanism

At this point, we have extracted from the simulation of the device a decomposition into driving and driven components. The decomposition has guided the choice of trajectories and displacement histories to collect. The analysis of these curves leads to a set of qualitative features characterizing the shapes of the curves. In the case of the dwell mechanism of figure 1, the system notes that:

- The angle of the rocker arm has a period of constant value.
- The coupler curve has a segment of constant curvature (i.e. a circular segment).

The final step in constructing an understanding of the mechanism is to notice relationships between these features as well as relationships between the curve features and metric properties of the links of the mechanism. It must then attempt to explain these relationships through geometric reasoning.

In particular, the system notes that:

- The radius of curvature of the circular segment traced by point E, the coupler point of the four bar linkage is nearly equal to the length of link 4, the coupler arm of the dyad.
- The distance from the fixed end of link 5, the rocker of the dyad to the center of curvature of the circular arc traced by point E is nearly equal to the length of link 5, the rocker of the dyad.
- There is a substantial overlap between the period during which the coupler of the four bar traces the circular arc and the period during which the rocker arm's angle holds steady.

Having found these overlaps, the system conjectures that the dyad has a dwell period which is *caused* by the coupler arm moving through a circular arc whose curvature is the same as the length of the driven arm of the dyad and whose center of curvature is at the location occupied by the dyad's joint when the circular arc is entered.

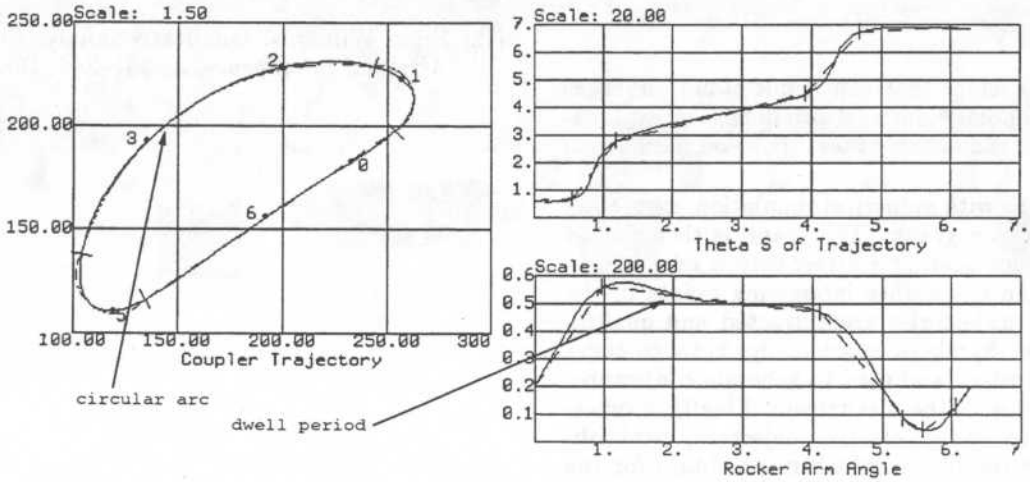


Figure 5: Curvature of The Coupler Curve and Angle of The Rocker Arm

Notice that this conjecture does not itself refer to any specific metric information from the simulation. If we can support the conjecture with reasoning which also does not depend on metric information specific to this linkage, then we will have deduced a universal principle applicable to a broader class of devices.

The final step is to use geometric reasoning to support the conjecture. The geometric knowledge needed to support the conjecture is very basic:

- Two circles intersect in at most two points.
- The center of a circle (the center of curvature of a circular arc) is the unique point equidistant (by the radius) from more than two points on the circle.

The reasoning supporting the conjecture is quite simple (and we omit it for brevity). It completes the interpretation of the mechanism and uses no metric information from the simulation but only qualitative shape features of the curves and symbolic relationships between joint positions. Any other mechanism satisfying these symbolic relationships will have the same behavior. **General information has been extracted from the simulation of a specific device.**

6 Adequacy of the Interpretation

An understanding of a mechanism should:

- Decompose the mechanism into understandable sub-mechanisms.
- Explain how the behavior of the whole arises from that of the parts.
- Assign a purpose to each of the components.
- Enable redesign by highlighting what interactions lead to the desired behavior.

Our explanation of the dwell mechanism meets all these criteria. It decomposes the linkage into two well known sub-linkages and explains how the shape of the coupler curve causes the dyad to dwell. The two sub-linkages have well understood purposes.

We also claim that this explanation of the mechanism enables redesign. Although our system is not a redesign system, we claim that a redesign system could use the kind of information we generate. In particular, it is clear that the four bar linkage could be replaced by any other mechanism which generates a curve with a similar circular arc segment.

We have run our system on several mechanisms from [1] and several other source books of mechanisms. We handle multiple dwells mechanisms, frequency multipliers, quick returns and a variety of stand-alone uses of four bar linkages. The modules understood by the system include planetary gears, scotch-yokes, pantographs, dyads, cams, four-bar linkages, dyads, slider-cranks, etc.

7 Comparison to Other Approaches

There have been other projects on understanding kinematic mechanisms, (e.g. [5; 3; 8; 6]. These have been concerned mainly with determining when state transitions occur, typically when contact between bodies is established and broken. Although this is an important and difficult issue in the general case, it does not occur in the domain of linkages (or more generally fixed topology mechanisms).

Our central concern is deriving qualitative features of the shapes of curves generated by driving mechanisms; and this is quite different from those generated by these systems.

With the exception of [6; 8], most of these systems are based on qualitative simulation.

One system [9] attempts to apply qualitative simulation to linkages. Kim's system conducts a form of envisioning of the behavior of a four-bar linkage. However, the system as described does not predict the shape of coupler curves, nor does it deal with more complex systems which use four bar linkages as driving mechanisms.

The shape of a coupler curve is governed by highly non-linear equations (it is a 6th degree curve). [7] points out the difficulty of relating link sizes to coupler curve shapes and catalogues several thousand coupler curves as a service to designers. Because the equations are highly non-linear, it is unlikely that qualitative simulation can derive the shape properties of coupler curves.

8 Summary

We have shown a system that can "understand" linkages (and other fixed topology devices) producing an explanation very similar to that given in textbooks on mechanical design.

Our system begins with numerical simulation, extracting from this a mechanism graph. The graph is then parsed into familiar modules bearing a driver-driven relationship to one another. This identifies interesting points in the mechanism whose trajectories are extracted and qualitatively characterized. Symbolic relationships between curve features are then noticed and used to generate conjectures about the functioning of the mechanism. Finally, geometric reasoning is used to support the conjecture, establishing the qualitative conditions which must obtain for the observed behavior to result.

This process has been shown to extract general design principles from specific mechanisms.

References

[1] I.I. Artobolevsky. *Mechanisms in Modern Engineering Design.* MIR Publishers, Moscow, 1975.

[2] Johan deKleer. Causal and teleological resoning in circuit recognition. Technical Report TR-529, Massachussetts Institute of Technology, AI Lab., Cambridge, Mass., September 1979.

[3] Boi Faltings. A theory of qualitative kinematics in mechanisms. Technical Report UILU-ENG-86-1729, University of Illinois at Urbana-Champaign, Urbana, Illinois, May 1986.

[4] Kenneth D. Forbus. Qualitative process theory. Technical Report TR-789, Massachussetts Institute of Technology, AI Lab., Cambridge, Mass., July 1981.

[5] Kenneth D. Forbus, Paul Nielsen, and Boi Faltings. Qualitative kinematics: A framework. Technical Report UILU-ENG-87-1739, University of Illinois at Urbana-Champaign, Urbana, Illinois, June 1987.

[6] Andrew Gelsey. The use of intelligently controlled simulation to predict a machine's long-term behavior. In *Proceedings of the National Conference on Artificial Intelligence*, pages 880–887. AAAI, 1991.

[7] J.A. Hrones and G.L. Nelson. *Analysis of The Four-bar Linkage.* MIT Press and John Wiley & Sons, New York, 1951.

[8] L. Joskowicz and E.P. Sacks. Computational kinematics. Technical Report CS-TR-300-90, Princeton University, Princeton, N.J., April 1990.

[9] Hyun-Kyung Kim. Qualitative kinematics of linkages. Technical Report UILU-ENG-90-1742, University of Illinois at Urbana-Champaign, Urbana, Illinois, May 1990.

[10] Glenn A. Kramer. Solving geometric constraint systems. In *Proceedings of the National Conference on Artificial Intelligence*, pages 708–714. AAAI, 1990.

[11] Benjamin J. Kuipers. Qualitative simulation. *Artificial Intelligence*, 29:289–338, 1986.

[12] Brian Williams. Qualitative analysis of mos circuits. *Artificial Intelligence*, 24:281–346, 1984.