MARÍA JOSÉ GONZÁLEZ-LÓPEZ

USING DYNAMIC GEOMETRY SOFTWARE TO SIMULATE PHYSICAL MOTION

ABSTRACT. In this paper we analyse to what extent the computational model of the geometry implemented in a dynamic geometry environment provides models for physical motion, focusing on the continuity issues related to motion. In particular, we go over the utility of dynamic geometry environments to simulate the motion of mechanical linkages, as this activity allows us to compare, by means of dynamic drawings, the computable representation of geometric properties with the real motion of a mechanism. Analysing a simple example, we provide theoretical foundations for particular behaviours observed in the motion of a picture on the screen, which require a subtle interpretation to be understood in a purely physical context. In this way, we reflect on some requirements imposed by the computable representation of knowledge. We consider this work to be a necessary step to determine didactic consequences related to students' perceptions of the moving displays; in particular those concerning the uses of the dragging mode as a tool not only for automatic drawing of many instances of a construction, but also to produce continuous motion.

KEY WORDS: computable representation of knowledge, continuity, dynamic drawing, dynamic geometry software, physical motion simulation

"*motions is a subject of mechanics rather than of geometry*"
Freudenthal (1983, p. 342)

INTRODUCTION

In compulsory education, geometry is presented as a model to organize the physical environment. This approach also includes consideration of the motion of objects in space, conceptualized in geometry by means of geometric transformations. But, as Freudenthal (1983, p. 342) points out:

Motion is first of all something that occurs to an object, within space (or plane), within time. In order to view motions the way mathematicians are now used to, three steps must be taken: from the limited object to the total space (plane), from 'within' space (plane) to 'on' space (plane), from 'within time' to 'at one blow'.

In this way, Freudenthal establishes how geometric transformations model the 'initial state/ final state' relation in the motion of a physical object in space and time.

International Journal of Computers for Mathematical Learning **6:** 127–142, 2001.

Another way in which we relate geometric theory and the physical world is through pictures, which are representations both of geometrical concepts and of physical objects, and can be considered essential mediating tools for the acquisition of geometric knowledge. Classically, pencil and paper have mostly been used to form these representations, even though the pictures obviously cannot move. The development of dynamic geometry overcomes such limitations, allowing pictures to move on the screen in a way that keeps some geometrical relationships invariant. This feature adds new attributes to the pictures, which can be used to promote students' acquisition of geometrical knowledge from exploration and conjecture activities (Laborde, 1998; Laborde and Straesser, 1990).

But we have to take into account new epistemological and didactic questions derived from the computable representation of the knowledge (Balacheff, 1994). In dynamic geometry environments, the motion of pictures on the screen corresponds to a particular computational modelling of the geometry, which has consequences on the behaviour of the users and their reasoning style (Hölzl, 1996).

In this paper we analyse to what extent the computational model of the geometry implemented in a dynamic geometry environment provides models for physical motion. We focus our interest on the *continuity* issues related to motion; in particular, we go over the possibilities of Cabri-Géomètre to simulate the motion of mechanical linkages. By comparing the behaviour of the dynamic drawing of a mechanical linkage with its real motion, we can make an in-depth study of some theoretical questions that arise in this context:

- modelling aspects: the relationships between the perceptual level of the continuity of physical motion and the theoretical concept of continuity in a scientific theory (continuity of the functional dependence of different variables: position of a point with respect to another point, orientation, etc); the validity of dynamic geometrical models to perform continuity questions; the differences existing between the model and the knowledge to be modelled.
- algorithmic aspects: the consideration of diverse algorithmic constructions to highlight different aspects of the same mechanism.

Thus we present, in Section 1, the concept of continuity that, from physics and mathematics, is used to organize the attributes of physical motion, observing that the motion that is produced on the graphical screen of a dynamic geometry software has, *a priori*, a different referent, which is the euclidean geometry theory. In Section 2, we establish the distinct phases that have to be considered when using a dynamic geometry environment to simulate the motion of a mechanical linkage, sketching the particular

relationships existing between the different domains that come into play (physical, mathematical and computational). We progress to a particular behaviour of the moving drawings, detailing this feature in Section 3, where we propose different algorithms to construct a basic mechanism in Cabri-Géomètre, and interpret the corresponding results. Section 4 focuses on some theoretical foundations that support the kind of properties fulfilled by the computational modelling. We consider that only when this analysis has been carried out can didactical consequences be deduced, mainly in order to design activities for the students to gain new geometrical knowledge from their perceptions on the moving drawings. Therefore, in Section 5 we present some reflections on this educational perspective, a detailed analysis of which is still to be carried out.

1. CONTINUITY IN MOTION MODELLING

Modelling of physical motion is a recurrent theme in scientific / philosophical theories. It involves diverse features: space and time conceptions, geometrical relationships (relative position, orientation), dynamical aspects (force, speed), characteristics of the material devices (elasticity), etc. Zeno's paradoxes constitute a paradigmatic example, reflecting the difficulty of reconciling our perceptions with the foundations of those theories.

One of the possible ways to model the perceptive continuity of physical motion is to focus on the system of concepts developed in mathematics or physics around the continuum and the continuity of functions. In classical kinematics, the motion of a 'material point' is characterised by a continuous relation between four (space/time) coordinates. The consideration of 'material objects' (instead of points) implies new variables to be managed; for example, those related to orientation. This causes us to consider a more complex framework where continuity is not an absolute concept, but dependent on the particular set of parameters we are relating.

In a computational context of dynamic geometry, the foundations of modelled knowledge is euclidean geometry theory, where there are not, a priori, direct connections with the functional continuity. In fact, the initial proposal of the motion of a drawing built within a dynamic geometry environment is to perform the invariant geometric properties shared by any exemplification of the corresponding geometric construction. Therefore, the dragging mode can be identified with the facility given to the user to make many examples of the same construction, rapidly and effortlessly.

But, when using dynamic geometry software to simulate the behaviour of a mechanism (a pulley, a bicycle, a crank, etc.) we are trying to relate the

computational representation of geometrical concepts with the perception of physical motion, and we expect the dragging of the picture to reflect the continuity that we perceive on the real mechanism. We analyse in the following sections the extent to which our expectations could be fulfilled.

2. SIMULATING THE PERFORMANCE OF MECHANICAL LINKAGES IN DYNAMIC GEOMETRY SOFTWARE

In order to simulate the performance of a mechanism in a dynamic geometry environment we have to consider three different phases:

- Geometrical Phase: we examine the geometrical properties that make the mechanism work. This activity involves not only a pure geometry description, but also a functional description relating to the real mechanism. For example, if we are going to construct a bicycle, we remark that the chain is tangent to the gear, we determine its connection with the pedal, we identify the pedal as the part of the mechanism that brings the others into operation, etc.
- Algorithmic Phase: we make the geometric construction of the mechanism using an ordered sequence of system commands. In the case of the bicycle, we identify the geometric objects required (points, circles, segments, etc.) and their dependencies, in order to deduce the sequence to be implemented in the system. There can be several algorithmic descriptions producing the same geometric construction. We can also use different sequences to focus on particular behaviours of the same mechanism.
- Testing Phase: we use the dragging mode of the system to produce the motion of the drawing on the screen, in order to visualize the performance of the built 'ideal model of the mechanism'. The system keeps the geometric features of the construction invariant.

In this third phase we can check the fidelity of the construction by comparing the behaviour of the moving drawings with the real mechanism's performance. In a simulation, the motion of the drawings on the screen would ideally be the same as a video recording of the physical device. However, the geometrical laws supporting the motion of the pictures have no direct link with the physical motion laws nor the continuity of an analytic model. And there are still computational laws playing their role, as Hölzl (1996, p. 171) comments:

"*implementing a drag-mode demands decisions about the behaviour of geometrical objects when they are moved; and some of those characteristics may not all lie in the realm of geometry.*"

The situation we want to reflect is summarised in the scheme shown in Figure 1:

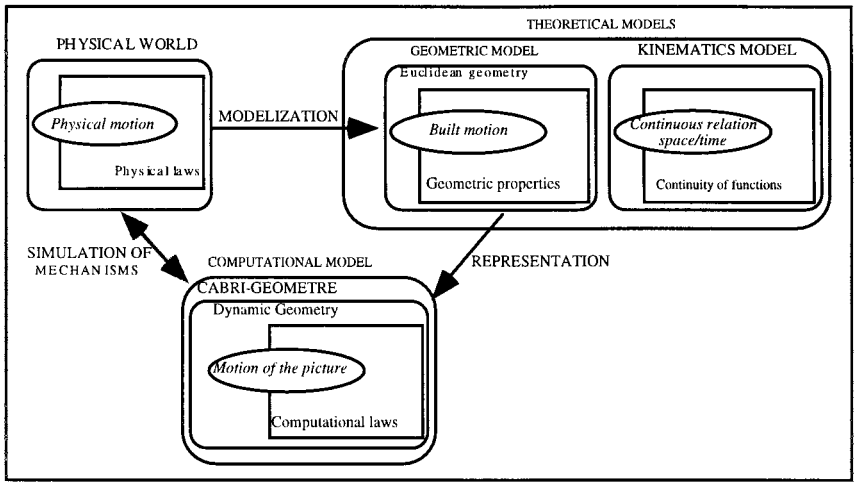

Figure 1. Relations between the physical, theoretical and computational contexts in a simulation activity.

These conditions are those responsible for particular behaviours of the moving drawings observed in the testing phase of a simulation. For example, while dragging one free object of a geometric construction, other elements can suddenly change their position on the screen, 'jumping' from one position to other and breaking the expected perceptive continuity. In the following section we detail an example, analysing the possibilities offered by the system to implement various algorithmic processes of a construction and interpreting the corresponding results.

3. A MEANINGFUL EXAMPLE

Let us consider a mechanical linkage as simple as a planar linkage composed by two rigid equal links with a common endpoint (Figure 2). The extreme O is fixed to a table and we move the mechanism on a surface by dragging P. Let us construct this mechanism in Cabri-Géomètre, following the phases outlined in the previous section:

Geometrical Phase: the two mechanism links are segments of equal length (r) in which the finishing point of one segment is identified with the point of origin of the other (point Q). Whatever the position of the mechanism,

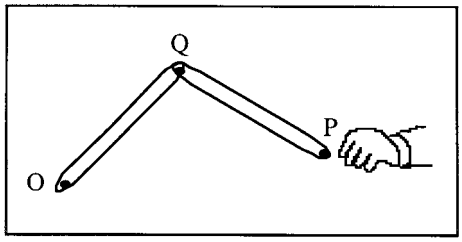

Figure 2. 2-links planar mechanical linkage.

Q is the same distance (r) from the extremes O and P. O is a fixed point and P is a free point whose motion pulls the rest of the mechanism.

Algorithmic Phase: Different sequences can be implemented to perform the requirements of the previous phase. We show three different possibilities:

- Algorithm 1: O and P are free points. Q is determined by the intersection of two circumferences: C_1 (centered in O and radius r) and C_2 (centered in P and radius r). We choose one of the two intersection points to define Q. Finally, we draw the segments OQ and QP (Figure 3(a)).
- Algorithm 2: O and P are free points. C is a circle centered in O and radius r. We draw a perpendicular to line OP through O and a parallel l to this line through the midpoint of the segment OP. We select Q to be one of the two intersecting points between C and l, and draw the segments OQ and QP (Figure 3(b)).
- Algorithm 3: O is a free point. We draw a line l passing through O. P is a point on l. Then we follow the same steps in Algorithm 2 (from P and O) (Figure 3(b)).

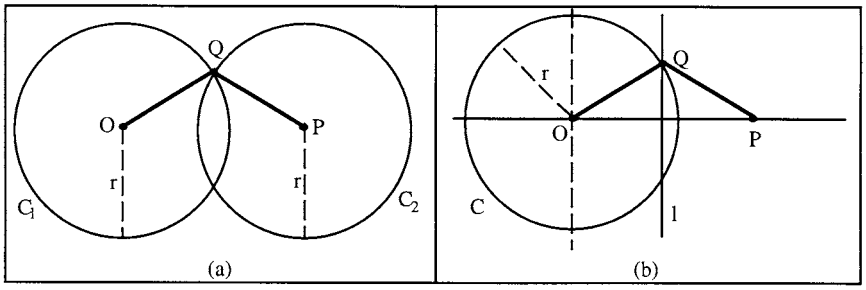

Figure 3. Two geometric implementations of the 2-linkage in Cabri Geométrè.
(a) From two equal circumferences.
(b) From a circumference and a line.

In Algorithm 1 we directly preserve the distance (r) from Q to P and O by means of two circumferences. Algorithm 2 combines one circumference and the perpendicular bisector of OP. Algorithm 2 could be made simpler (by drawing the perpendicular bisector of OP directly, without using the perpendicular to OP through O), but we have chosen that description to compare Algorithms 2 and 3.

Testing Phase: When dragging P on the screen we observe the following facts (which we compare with the performance of the real mechanism in order to explain the 'simulation of mechanisms' arrow in Figure 1):

a) When moving P away from O in the three algorithms the mechanism drawing disappears (circumference-circumference or circumference-line do not intersect). In the limit cases the two segments lie end to end on the same line (circumference-circumference or circumference-line are tangent). This behaviour has a direct reflection in the real mechanism, where P cannot be more than 2r from O. However, we can avoid these limit positions, as Cabri-Géomètre allows us to implement 'conditional geometrical constructions'.
b) In Algorithms 1 and 2, when P comes close to O the two segments tend to be the same. If we drag P exactly through O then Q 'jumps' suddenly. This behaviour can be seen more easily if we redefine P as 'point on' a line through O: the dragging of P from one side to the other of this line produces a jumping of Q from one of the half-planes determined by the line to the other (Figure 4(a)). However, dragging P over O in Algorithm 3 produces no jump of Q (Figure 4(b)). This motion through O is possible in a physical context if we consider our example as the planar model of a real mechanism where the two links move in distinct parallel planes.

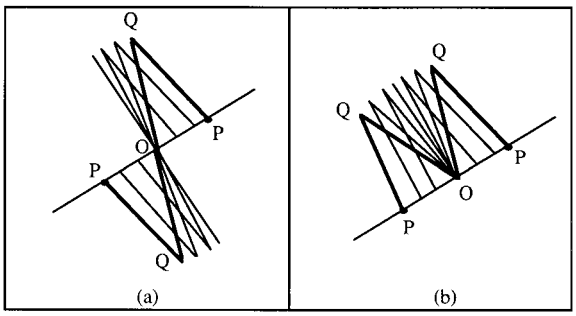

Figure 4. Behaviour of the construction when P is dragged through O (a) in Algorithms 1 and 2; (b) in algorithm 3.

4. THEORETICAL CONSIDERATIONS

There is no behaviour in the real mechanism equivalent to this jumping of Q in Algorithms 1 and 2. Otherwise, Algorithm 2 has a very similar description to Algorithm 3, but a different performance when passing through O. We can find the coherence of these facts in the computational modelling of the euclidean geometry implemented in Cabri-Géomètre. In this section we describe some characteristics of this theoretical framework.

4.1. *Orientation of Geometrical Objects*

The existence of two intersection points between circumference / circumference or circumference / line (in Algorithms 1 and 2) reflects the fact that for one given position of P there are two possible configurations of the mechanism (Figure 5). We arbitrarily choose one, as both of them have, a priori, the same physical and geometrical status for any fixed configuration (i.e. in a static geometrical view). But this choice is not innocuous from the dynamic point of view.

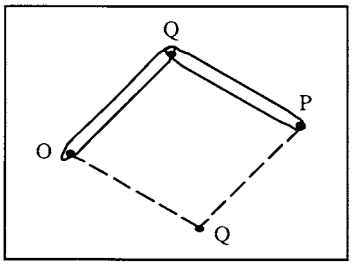

Figure 5. Two distinct configurations reach the same point P.

The dynamic perspective involves the need to distinguish the intersection points of two objects in a geometric construction (see, for example, Laborde, 1997) for the reflection of a segment with respect to a line, or the construction of a rhombus). For this purpose most of the dynamic geometry environments use an *internal orientation* of the geometrical objects. For example, a line through two points A and B is oriented 'from A to B'; the orientation of a perpendicular to a line is obtained by turning $\pi/2$ the orientation of the line counter-clock-wise (CCW); two parallels have the same orientation.

The dragging tool preserves these orientation laws, thus keeping a 'continuous' behaviour of the orientation variable. Therefore, passing P through O in Algorithm 2 keeps the line OP oriented from O to P, this fact being directly responsible for the jumping of Q (Figure 6(a)). Instead, in Algorithm 3 we start with a line through O, oriented in a certain way,

then we choose P on this line. But the dragging of a 'point on' an object does not change its orientation. This slight modification of Algorithm 2 makes the perpendicular lines' orientation independent of the position of P (Figure 6(b)).

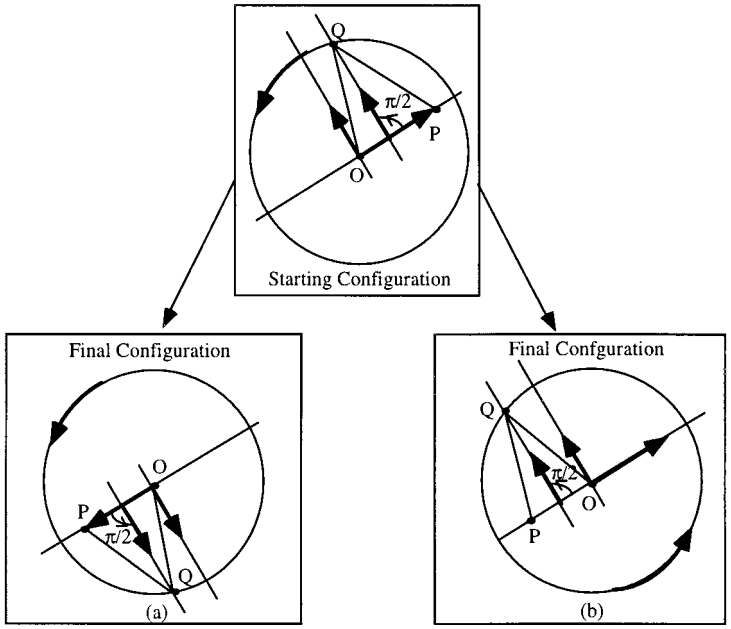

Figure 6. Orientation behaviour of the geometric elements when P is dragged through O.
(a) In Algorithms 1 and 2.
(b) In Algorithm 3.

Thus, we note that a continuous behaviour of the objects' orientation can correspond to a visual discontinuity in the position of Q. Therefore, depending on the feature we want to focus on, a subtle interpretation of continuity is required.

4.2. *Position Continuity and Orientation Continuity*

We present in this section the handling proposed in the realm of kinematics for the 2-linkage mechanism. This approach allows us to relate the two position/orientation perspectives in a common context.

We consider a system of coordinates (O,X,Y), where we are going to relate P and Q coordinates in terms of the following angles: let θ_1 be the (oriented) angle from the positive X-axis to the first link and θ_2 the (oriented) angle from the extension of the first link to the second one, both measured in a CCW direction (Figure 7). In order to simplify the

expressions, we consider the lengths of the two links to be equal to 1 (r = 1). Thus, we have:

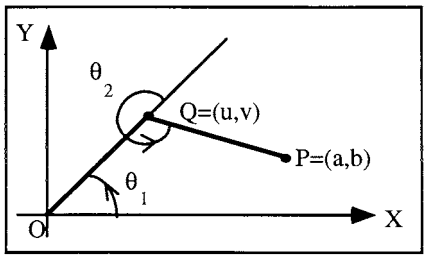

Figure 7. Angles measurement in the kinematics model.

$$Q = (\cos(\theta_1), \sin(\theta_1)); P = (\cos(\theta_1)+\cos(\theta_1+\theta_2), \sin(\theta_1)+\sin(\theta_1+\theta_2))$$

A continuous function F gives us the position of P corresponding to a pair of angles:

$$F(\theta_1, \theta_2) = (\cos(\theta_1) + \cos(\theta_1 + \theta_2), \sin(\theta_1) + \sin(\theta_1 + \theta_2))$$

In this way we consider two different spaces:

a) 'joint space', to denote the set of angles (θ_1, θ_2) representing the configurations of the whole mechanism, and
b) 'operational space', to denote the plane in which P = (a,b) moves.

By analysing the relations between both spaces we can identify the so-called 'singular points' of the mechanism, which in this case correspond, without describing technical details, to completely stretched (dist(P,O) = 2r) or completely folded (P = O) configurations. In geometrical terms, these singular points are characterised by two equal or tangent circumferences, or by a change in the objects' orientation.

The kinematic model distinguishes trajectories f(t) = (a(t),b(t)) of P in the operational space, from the whole mechanism trajectories g(t) = (θ_1(t),θ_2(t)) in the joint space. Both types are related by means of F (when $F_o g$ = f), and a central problem in kinematics is that of determining functions g for a given f. This is not a simple question, as F is not a one-to-one mapping: there are positions (a,b) of P reached with two angles (Figure 8(a)) or infinite pairs of angles (Figure 8(b)), associated with two or infinite positions of Q.

For a given trajectory f of P, the two possibilities for Q (Figure 8(a)) describe two different 'branches', each one of them corresponding to a function g. If P passes through O there is an undefinition at the point where

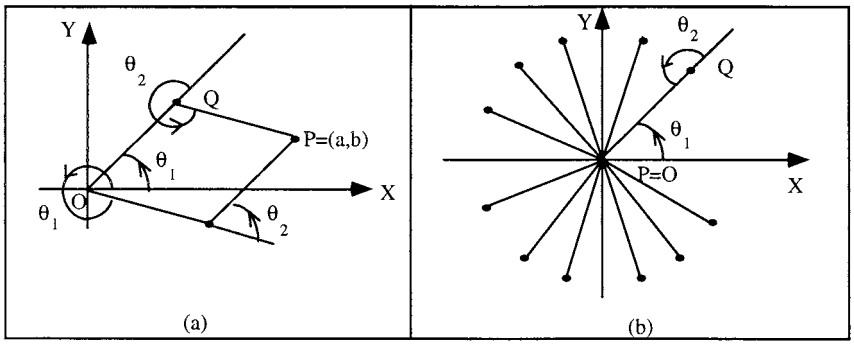

Figure 8. Configurations associated to a given position of P.
(a) Two different paris (θ_1,θ_2) reach the point P.
(b) If P = O there are infinite possibilities: any pair (θ_1,θ_2), θ_1 in (0,2π).

both branches should connect (P = O, with infinite possibilities for Q). The definition of g at this point can be chosen in response to two different criteria:

a) *Orientation continuity*: extending to O the behaviour of the mechanism in a neighbourhood of O (Figure 9). Algorithms 1 or 2 should represent this extension.

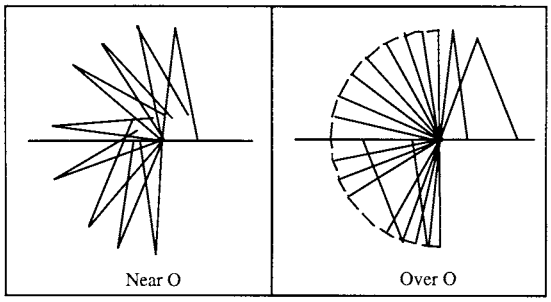

Figure 9. The motion of the mechanism when P passes through O can be considered an extension of its motion near O.

b) *Position continuity*: considering the continuous dependency of the y-coordinate of Q with respect to the x-coordinate of P. Algorithm 3 constitutes an exemplification of this choice.

The computational model of the geometry implemented in Cabri-Géomètre has allowed us to represent both choices, giving us the possibility of representing, in geometrical terms, the two continuity criteria stated in the kinematics model.

4.3. *Proximity and Domain of Validity*

J.M. Laborde (1999, p. 13) defines dynamic geometry as:

"*... the situation in which one can construct a figure following one of its possible procedural descriptions, progressing with base points [...] in such a way that the result of the construction meets the specifications and meets them also for* ***points close to the base points***".[1]

But in the same paper Laborde establishes that 'being close' in a dynamic geometry environment is not the same as in the realm of the topology of the plane. In this last case, a circular neighbourhood of A contains its close points; while in a dynamic geometry environment, a neighbourhood of A is an angle sector with vertex in A. Thus, every geometric construction in a dynamic geometry system has a 'domain of validity' that limits the motion of the free points to certain regions (of points close to them for the system), outside of which the implemented construction can lose its sense. In the 2-linkage mechanism, point O is outside the domain of validity of P in Algorithms 1 and 2, as dragging P through O modifies the sense of the construction producing a visual discontinuity that requires a particular interpretation in terms of orientation. This can be considered a computational constraint, as the orientation of a geometrical object is a concept internally controlled by the system, but not *a priori* required from the user's point of view. Once the orientation laws have been established, we acquire a new perspective on the tools provided by the system, which allows us to handle them in a more subtle way: we can combine the geometrical and computational requirements to improve the authenticity of the model (as in Algorithm 3).

4.4. *The Conservative Law*

We have seen that dragging a point outside its validity domain modifies the sense of the construction. But, we can see that when the dragged point returns to its starting position the construction recovers again its original configuration. This property, called *conservative*, can be stated as follows (Laborde, 1999, p. 14):

For any succession of positions of any base point P on a closed loop $P_1, P_2, \ldots, P_n = P_1$, *final state* $F(P_n)$ *is equal to the initial state* $F(P_1)$.

The computational model for the geometry implemented in the dynamic geometry software as Cabri-Géomètre guarantees that all the constructions we make verify the conservative property, which need not necessarily occur in other kinds of modelling. For example, in the field of dynamic geometry environments, the Cinderella system has recently been proposed

(Richter-Gebert, Kortenkamp, 1999). Its implementation is supported by different background theories (for example, complex analysis). This provides a different behaviour of the dragging mode: for example, Cinderella preserves the continuity of point Q with respect to P in Algorithms 1, 2 and 3, although it does not preserve the orientation nor verify the conservative property previously stated.

4.5. *Theoretical Constraints*

As we have seen, there are different algorithmic descriptions leading to the same geometric construction, each of them providing different domains of validity to the free objects. Thus, the authenticity of the simulation depends partly on the process followed by the user in the Algorithmic Phase. But there may be theoretical constraints that restrict our expectations in a simulation process. We will refer again to the comparison with the kinematic model to analyse this question in the rest of this section.

As we have said, a central question treated in the kinematic model is that of determining algorithms to generate trajectories in the joint space associated to a given trajectory of P in the operational space. The algorithms of this kind that verify the conservative property are called cyclic (Baker, 1990, p. 72). A particular conclusion of Baker's work is that there are no cyclic algorithms for the 2-linkage mechanism in a neighbourhood of point O. In other words, there is no global algorithmic solution independent of the trajectory described by P, in the case of its going through singular points. The reason is, briefly, that there are directions which cannot be reached from O and that they vary depending on the trajectory we have followed to arrive at this point.

As this is a physical restriction, we can expect that any other model used to simulate physical motion respecting the conservative law will have difficulties when passing through singular points; in particular, the computational model implemented in Cabri-Géomètre. Therefore, independently of the sequence followed in the algorithmic phase, passing through O would have needed particular treatment. This has been reflected in our process: Algorithms 1 and 2 constitute global approaches (for any trajectory of P) where we have detected that trajectories through O require a particular treatment; Algorithm 3 performs only rectilinear motion through O.

5. CONCLUSION AND DIDACTICAL CONSIDERATIONS

The characteristics of the motion of a picture constructed by using geometric laws allow us to state the peculiarities encountered in order to simulate physical motion in a dynamic geometry environment and reveal features of the euclidean geometry model when it comes to implementing continuity questions. The computational system allows us to perform different behaviours in a simulation, although this requires a profound knowledge of the internal implementation of the system commands. This has allowed us to reflect on some characteristics of the computable representation of knowledge, as a step towards determining their influence on educational uses of dynamic geometry systems.

In the educational context, Goldenberg-Cuoco (1998) reflect on new mathematical and pedagogical issues arising when using dynamic geometry software:

"*in designing curricula that make significant use of experimentation and investigation, we must understand what students glean from their experiments, and that clearly depends on what features of the experiment they notice.*" (p. 352)
"*Optimal curriculum design must be aware of places [. . .] where reasoning may be confounded by behaviours that are built into DG* [Dynamic Geometry] *tools but are not part of the (conscious) postulate set of the students. Research must also clarify what students do, in fact, notice, and how they do interpret what they see.*" (p. 354)

It is clearly important to know how students perceive a graphical screen on which pictures move, and how they manage to integrate these perceptions with their geometric knowledge in order to produce new knowledge. From the analysis presented in this work we conclude that when dynamic geometry software is used to model physical situations, some answers produced by the system can be separated from the geometrical knowledge of the student, as they are a consequence of the implemented computational modelling. Therefore, the list of properties to be perceived by the student in a picture (and research is trying to clarify which of these are seen as essential or inessential) must be enlarged with the characteristics derived from the computational model.

Bearing in mind that activities proposed to students usually have an exploratory and conjectural character, the fact that some unexpected performances can occur on the screen can be understood as a situation in which the student is faced with conflicts. This is a profitable approach when the solution lies within the students' reach, with or without guidance.

Similarly, we suggest that the design of geometric activities has to consider as a main point the domain of validity of every construction, outside of which the claimed geometrical properties require a particular

interpretation associated with the computational model used. A preliminary point we have verified with students is that they tend to avoid the dragging that produces unexpected answers, when they consider that the steps followed to make the construction are right. We can interpret this behaviour as the students' separating the geometrical aspects from the features imposed by the computational implementation, identifying (perhaps unconsciously) the domains of validity of every element of a construction. But it may also be possible that the student is using the dragging mode provided by the system merely as a tool that allows many instances of the same construction to be drawn automatically, without relating this feature to the continuity of motion. The interrelation between both facets could be improved if dynamic geometry systems were to make explicit the geometric properties arising when the dragging of the picture reaches singular points, and should allow the user to choose between the different possibilities offered.

The study of these questions is a subject of research into didactics of geometry concerned with establishing students' perceptions of a moving drawing, and the knowledge they generate from their perceptions. This paper has contributed to the research by determining some features of a picture moving in a dynamic geometry computational environment.

ACKNOWLEDGMENTS

I am grateful to Moisés Coriat, Laureano González-Vega and Tomás Recio for many valuable comments on the subject of this paper. This work has been partially supported by DGESIC PB98-0713-C02-02.

NOTE

[1] Emphasis mine.

REFERENCES

Baker, D. (1990). Some topological problems in robotics. *The Mathematical Intelligencer* 12(1): 66–76.

Balacheff, N. (1994). Didactique et intelligence artificielle. *Recherches en Didactique des Mathematiques* 14(1.2): 9–42.

Freudenthal, H. (1983). *Didactical Phenomenology of Mathematical Structures*. Dordrecht: Reidel Pub. Co.

Goldenberg, E.P. and Cuoco, A.A. (1998). What is dynamic geometry? In R. Lehrer and D. Chazan (Eds), *Designing Learning Environments for Developing Understanding of Geometry and Space* (pp. 351–367). Lawrence Erlbaum Assoc.

Hölzl, R. (1996). How does dragging affect the learning of geometry. *Int. Journal of Computers for Mathematical Learning* 1: 169–187.

Laborde, J.M. (1997). Exploring non-euclidean geometry in a dynamic geometry environment like Cabri-Géomètre. In James King and Doris Schattschneider (Eds), *Geometry Turned on* (pp. 185–191). MAA Notes, N. 41. The Mathematical Association of America.

Laborde, C. (1998). Visual phenomena in the teaching/learning of geometry in a computer-based environment. In C. Mammana. and V. Villani (Eds), *Perspectives on the Teaching of Geometry for the 21st Century* (pp. 113–121). Dordrecht: Kluwer Academic Publishers.

Laborde, J.M. (1999). Some issues raised by the development of implemented Dynamic Geometry as with Cabri-géomètre. *Proceedings 15th European Workshop on Computational Geometry* (pp. 7–19). H. Brönnimann Ed. INRIA Sophia Antipolis.

Laborde, J.M. and Straesser, R. (1990). Cabri-Géomètre: A micro-world of geometry for guided discovery learning. *Zentralblatt fur didaktik der mathematik* 5: 171–190.

Richter-Gebert, J. and Kortenkamp, U. (1999). *Cinderella – The Interactive Geometry Software*. Berlin: Springer Verlag.

Departamento de Matemáticas, Estadística y Computación
Facultad de Ciencias
Universidad de Cantabria
Av. Los Castros s/n
39071 Santander
Spain
E-mail: glopez@matesco.unican.es