

# Student-Oriented Software for Application of Geometry in 3D Modeling and Animation

Pavel Boytchev

*Elica Team  
Bulgaria, Sofia  
pavel@elica.net*

## Abstract

The paper is focused on research software called Elica that allows students to build basic geometrical objects like spheres and cubes, and modify them geometrically. These objects can be grouped into higher-level metaobjects that can be further modified. The software includes translator of a programming language, with which all constructions are described in an object-oriented style, allowing extended functionality.

To explore this functionality, techniques for using geometry operations in order to animate 3D models are presented. Some examples are about building compound geometrical objects like chess figures; adding behaviors like moving and jumping, and eventually, creating microworlds that students can use to master both programming and mathematics skills.

Abstract geometrical ideas of space and coordinate systems can be effectively used to build and animate complex 3D characters widely used in Virtual Reality system and in video games. These characters consist of many parts that can move in various directions. Positions and orientations in space of some parts are often defined relatively to other parts. Elica provides mechanism for describing these dependencies in an easy to comprehend way, hiding all the complexity behind a carefully designed microworld.

The virtual skeleton of the 3D creatures is managed by space turtle object that leaves traces at joints. Each trace is a clone of turtle orientation vectors for a specific position of the turtle. It is possible to hierarchically multiply matrices in order to recreate any desired creature posture. Further on, Elica uses posture definitions to create automatically interpolated sequences of frames that comprise complex character movement.

## Keywords

Logo, Elica, turtle graphics, space turtle, 3D characters, animation

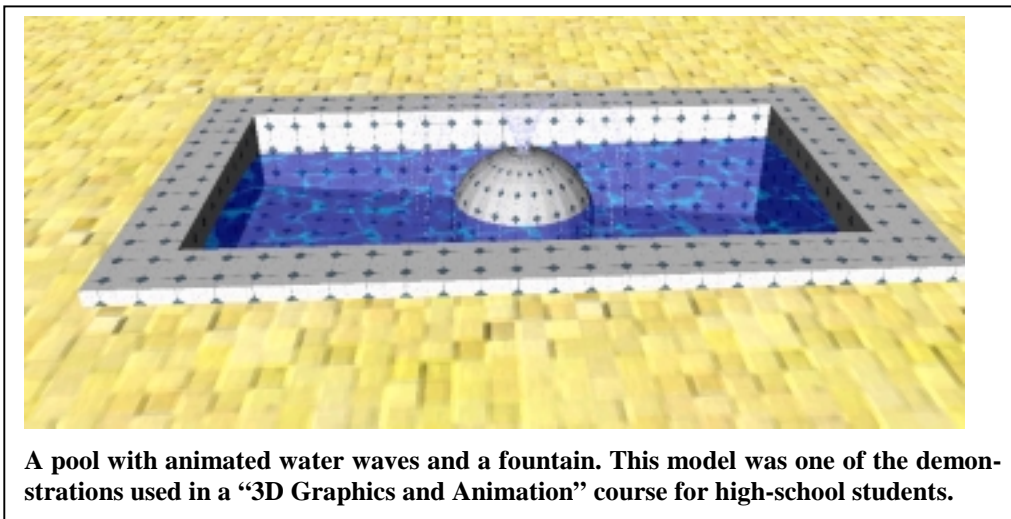
## 1. About Elica

Elica is a relatively new Logo implementation that expands the functionality of Logo language in various directions [Elica Team, (2003)]. One of the most recognizable of them is the ability to use friendly object-oriented style in building virtual worlds and animate virtual reality characters.

Originally the name ELICA stands for Educational Logo Interface for Creative Activities. However, it has been also suggested with good reason that it could also mean Enhanced Logo for Interactive Computer Animation.

The Logo dialect used in Elica is quite specific, and differs from all other mainstream Logo implementations. The distinctions are mainly in the set of reserved words. Other Logos hardcode all commands and functions (including those of Turtle Graphics) in their systems. Elica, on the other hand, has a small core that can process just a few Logo commands, namely: MAKE, PRINT, IF, WHILE, REPEAT, RUN, TO, END, LOCAL, OUTPUT and OB [Boychev P (2000)].

This scanty collection of commands is the cause of the Elica's highly flexibility. The core is sufficient to rebuild all other Logo command and functions, including Turtle Graphics. The flexibility extends beyond the recreation of the standard Logo and reaches areas which have been routinely neglected by Logo – 3D character animation in real time [Boychev P, (2003)]. This neglecting is often supported by the nature of the Logo language – it is often implemented as interpreter. Additionally, it is expected that Logo should be focused on list/word processing and Turtle Graphics.

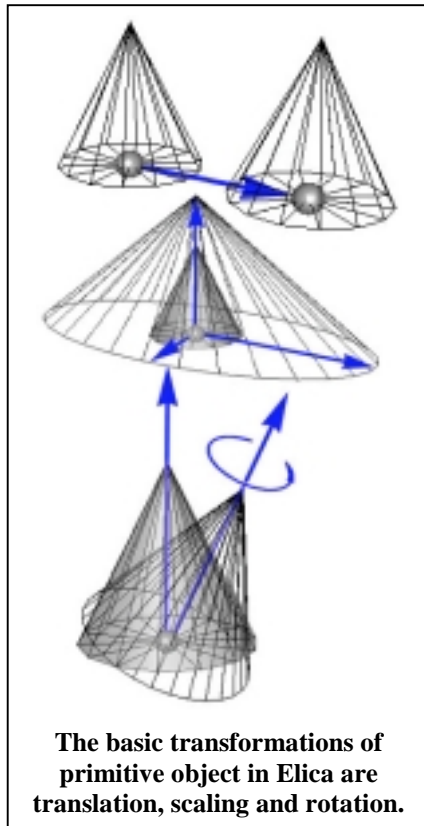


Another important feature of Elica is that it has been free software since its beginning. Although the core is not Open Source, the sources of all Logo-based libraries, microworlds and applications are available to the user. For the last three years Elica spread all over the world and as of today there are more than 300 registered Elica users, more than 3000 Elica downloads, and more than 30,000 visits of the Elica home page.

## 2. Static objects and the fixed-joint approach

### 2.1. Basic 3D objects

The core idea behind Elica 3D graphics is that almost all objects are composed of primitive objects. These objects are just four – cube, sphere, cylinder and cone. Each of them



can be modified in several ways. More complex objects are composed of shapes that are modifications of the primitive objects.

By default primitive objects can be changed with various basic transformations – translation, scaling and rotation. All the transformations are done by modifying objects' properties. Elica graphics has been designed in such a way, that all primitive objects have the same properties. This allows students to apply knowledge about one object onto another one.

All primitive objects have a center. This is a point in 3D defined by three numbers – its x, y and z coordinates. When students run a program that modifies the center of an object, they move the object and can monitor this movement on the screen.

In a similar manner there are parameters that control the scaling of an object in each of the main directions. These parameters are called *radiusx*, *radiusy* and *radiusz*, because they were initially made to work for spheres. Later on, in an attempt to increase the consistency of objects, these parameters have been made

available for all other primitive objects. For example, the *radiusz* parameter of a cylinder, is actually its height. This unified way to treat object size appears to cause no problems to students. Even junior high-school students can easily understand and use the concept of radii that determine the size.

One of the most difficult things for beginners is the orientation of objects. In Elica it can be defined in numerous ways. Each of them is suitable for some tasks and unsuitable for others. The most straightforward method for changing an object's orientation is to set its direction as a vector. However, this is not enough, because if an object is turned around its own main axis, then its direction remains effectively the same. To control orientation students use parameters *focus* and *spin*. *Focus* is a vector that determines the direction of the object. By default all objects have *focus* (0,0,1). The spinning around this vector is controlled by the *spin* parameter, which is a scalar. The problem in orientation comes from the *focus* – it is a vector defining a direction, not a point in the space.

## 2.2. Advantages and disadvantages

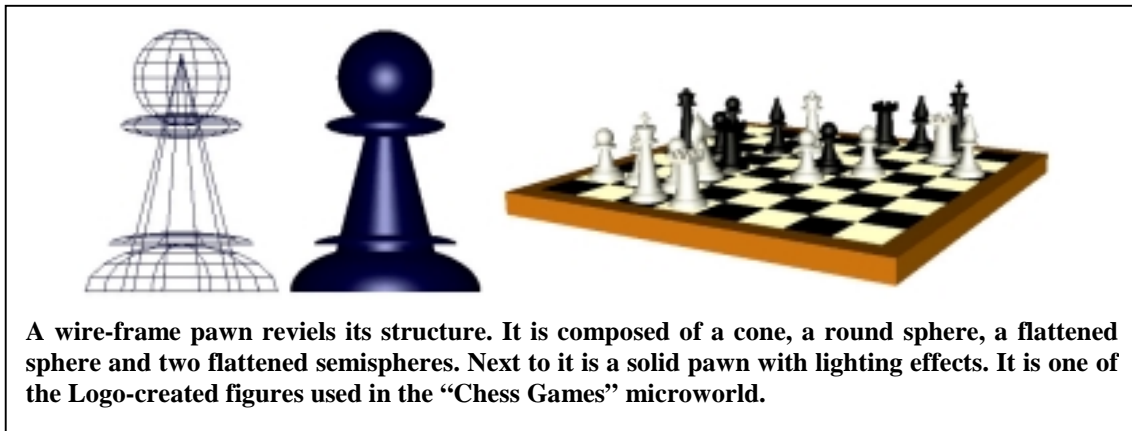
The construction of objects using primitive objects is one of the most direct ways to create and animate 3D scenes. An obvious advantage of this approach is that all objects are described geometrically. This gives a clear picture of what is shown in the screen and how it looks like. Another advantage is that all modifications to primitive objects are done in a consistent way. The unification of object properties appears to overcome the problems of treating heights as radii.

The first courses on 3D graphics based on Elica show another benefit. In some cases, depending on the problem being solved, students like to pinpoint objects at specific positions. They directly change the centers of the objects, because they know that centers determine absolute positions.

The nature of the primitive objects and their properties makes it easy to construct and animate fixed-joint object. These are objects that are composed of rigid parts that do not change dynamically.

The approach of geometrically defining parameters of each object causes some problems. Positioning and scaling does not appear to be difficult to students. However 3D orientation is a serious problem to some of them and is not appropriate to young students without a carefully planned teacher advice.

Finally, the biggest disadvantage of the fixed-joint approach is that it cannot be used to do flexible-joint objects without knowledge and experience in analytical geometry.

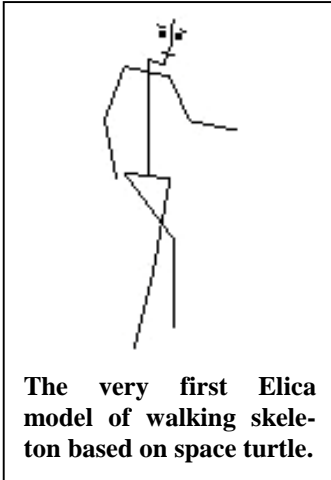


### 3. Dynamic objects and the flexible-joint approach

#### 3.1. Back to the roots of Turtle Graphics

To resolve the problems of the fixed-joint approach, Elica enhance Logo with 3D Turtle Graphics. The traditional turtle in other Logo implementations is represented as a triangle that moves in the plane of the screen and draws lines and curves [Abelson H and diSessa A, (1981)]. Elica, on the other hand, contains a library that defines several types of turtles – from the simplest one to more advanced space 3D-shaped turtles [Loethe H, (1992)]. The latter can be used to draw 3D constructions by controlling the turtle in the space.

This method of controlling objects is very popular nowadays among students. They play a lot of video games where characters are controlled almost as if they are turtles. Even if it is quite unacceptable to call fast-moving complex 3D characters in video games turtles, they are effectively fancy turtles [Platinum Pictures Multimedia, (2003)]. The general distinction is that these turtles are controlled via the keyboard, with no programming efforts.



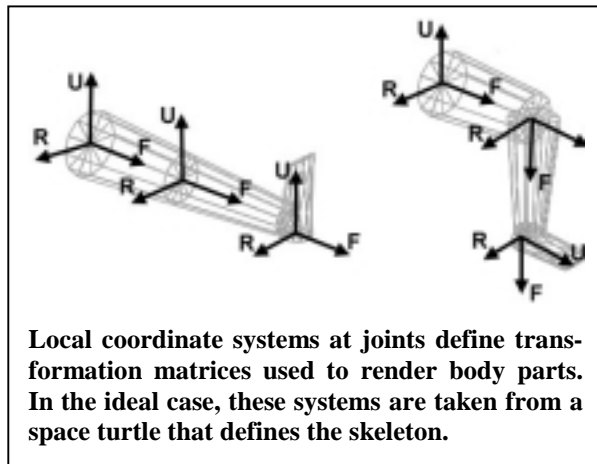
The space turtle in Elica is defined by 4 vectors. One vector for the positions and three vectors for the orientation<sup>1</sup>. These three vectors are de-facto a local coordinate system for the turtle. When it moves and turns in 3D space, its local coordinate system echoes all activities in such a way, that it repeats all turtle activities. Turtle graphics can be used to define the skeleton of a fixed-joint object. In the places of “bones” the turtle just goes forward. At “joints” it turns towards the required direction.

In fact, the very first implementation of 3D characters in Elica (about 3 years ago) was entirely based on this technique. Unfortunately, this does not address the problem of adding flesh to 3D models. In other words, if the model is supposed to be built up from primitive objects, then their positions and orientations could be determined only after some time-expensive calculations. The only way to quickly add flesh is to know the position and the orientation of each object. Because primitive object by default have unit sizes, are placed at (0,0,0) and are directed upwards, it is only needed to store a single affine transformation matrix that transforms the default object into the desired object.

One of the crucial goals of Elica is to find out a way to merge both techniques – the one that uses Turtle Graphics to define the skeleton, and the other that uses local transformation matrices at every joint, thus making it fleshy. If the merge is possible, then it might be relatively easy to build a 3D character and possibly animate it in real time.

### 3.2. 3D Characters

Fortunately the design of the graphical features of Elica is based on both techniques, and as a result they can be used together. For example, every graphical turtle<sup>2</sup> in Elica have a method called CLONE. It is use to create a copy of another object (a cone, for example), at the current position of the turtle and with the current orientation of the turtle. Internally this is implemented in a direct way – turtle’s local coordinate system is used as an affine transformation matrix for rendering the cloned object.



<sup>1</sup> Three vectors for orientation are too much as long as only two are enough. However, this is done for performance reasons, as it make any type of rotation entirely symmetrical in terms of algorithm and data processing.

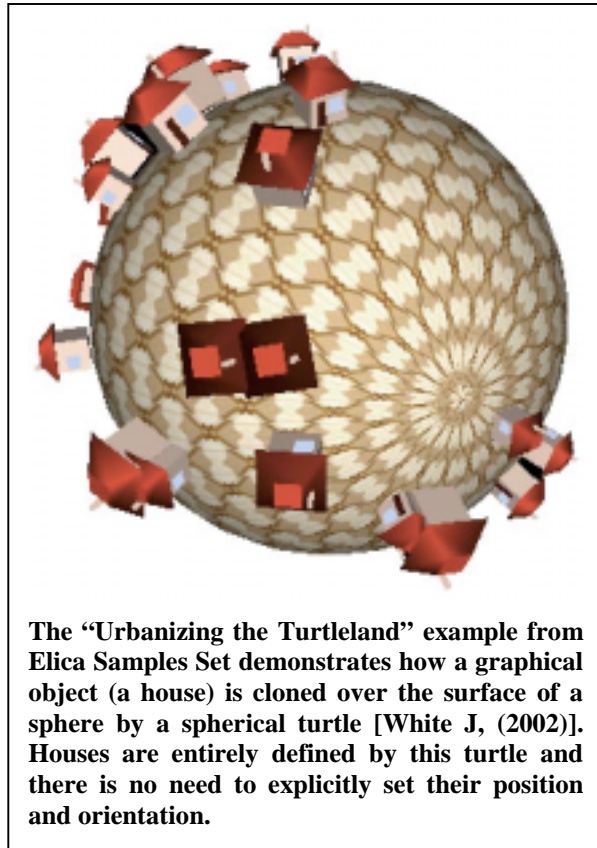
<sup>2</sup> Graphical turtles are all those turtles which live in 2D or 3D space, leave traces, and can be used to draw or to define drawings. However, there are non-graphical turtles, like the camera and the binary-file turtles.

This technique, although simple by idea and by implementation, is a milestone on Elica's way to 3D characters. The turtle can already walk along all "bones" of a 3D model. It uses its local coordinate system as an affine transformation at every "joint". Thus the transformation is applied to the "flesh" of the "bone" attached to this "joint". In this way the flesh will appear on the correct place in the space and will be orientated in the correct direction.

From mathematical point of view, the process of transforming an object by a sequence of matrix transformations can be nicely represented as a single multiplication by a matrix, which is the product of all transformation matrices.

When students define 3D characters with Elica they do not deal with matrices and do not pay attention, which of them should be multiplied, and in what order. Elica turtle takes care of this in a very natural way. While it travels over the skeleton, its local coordinate system

accumulates all its movements and rotations. So, at every joint the system already has a ready-to-use transformation matrix – this is local coordinate system of the turtle.



### 3.3. Advantages and disadvantages

It stands to reason that the biggest advantage of the flexible-joint approach is that it allows student to create more complex and brisk characters. These characters can be loaded with behavior and gestures, thus increasing the expressive power of the model. The use of Turtle Graphics is crucial not only for the internal processing of animation, but also in a pedagogical aspect. Turtles eliminate the need to directly work with absolute coordinates and directional vectors, and this opens new horizons to 3D animation. Namely, much younger student can now construct and animate their models.

Unfortunately, the fact that students are free from performing all the time-consuming calculations does not mean that these calculations have been eliminated. In reality, they have been embedded in Elica's objects (like turtles). So, if the model is complex, and if it has lot of moving body parts, then the "real-timeness" of the animation can be a problem, especially on computers with older hardware.

This has quite a negative impact on the use of 3D Logo-based animation in schools. Local schools do not have with sufficient number of well-equipped computers. However, there is a solution to this problem too. A course called "3D Graphics and Animation" is designed to address the inspiration and knowledge of high-school students. Special ar-



rangements are made with the principal of a local school and the owner of a game club, so the course is done in the game club, not in the classroom.

#### 4. Challenges

The future development of Elica is quite challenging. This is not only because there are many things that can be incorporated in Elica, but because Elica can effectively port new technologies into education. This is quite important issue, according to the author, because in most local schools there is a huge mismatch between technology used in education, and technology outside the school.

Education should definitely go out of the era of chalk and blackboard. However it will be disastrous if this is done by simply “implanting” computers in the classroom, without providing effective educational methods.

The core challenge ahead of Elica is to become a more classroom oriented tool. A tool that student can use to learn and explore things, as well as a tool that teacher can use to represent ideas and concepts.

#### 5. References

- Abelson H. and diSessa A (1984), *Turtle Geometry*, MIT Press, 144
- Boychev P (2003), *Turtle Metamorphoses (From “FD 1” To 3D Animated Characters)*, Proceedings of 9<sup>th</sup> European Logo Conference EuroLogo’03, Portugal
- Boychev P (2000), *Programming as Poetry (The Power of the Simplicity in Programming)*, Proceedings of 29<sup>th</sup> Spring Conference of the Union of Bulgarian Mathematicians, Lovetch, Bulgaria
- Elica Team (2003), *Elica*, <<http://www.elica.net>> (Mar 2003)
- Loethe H (1992), *Conceptually Defined Turtles*, Learning Mathematics and Logo, MIT Press, 55-95
- Platinum Pictures Multimedia (2003), *Animals / Insects*, <<http://www.3dcafe.com/asp/animals.asp>>, Platinum Pictures Multimedia, Inc. (Mar 2003)
- White J (2002), *Mathwright Microworld – Spherical Logo*, <[http://www.mathwright.com/book\\_pgs/book625.html](http://www.mathwright.com/book_pgs/book625.html)> (Mar 2003)