

# Nonlinear Space Transformations and Educational Software: One-Step Transformations<sup>1</sup>

Vladimir Nodelman

[nodelman@hit.ac.il](mailto:nodelman@hit.ac.il)

Department of Computer Sciences,  
Holon Institute of Technology  
52 Golomb St., Holon, 58102, ISRAEL

## Abstract

*The possibility to study such fundamental notions of modern mathematics as “space” and “spatial transformation” is almost absent in educational software. Instead, this software handles **affine transformations of objects** lying in 2D or 3D space: specifically, compositions of rotations, translations and homothety, all applied to geometric figure.*

*It is relatively easy to implement affine transformations of the **whole space** programmatically due to internal nature of computer graphics mechanism. The problem is to support **nonlinear spatial transformations** in a manner that is user-friendly and seamless.*

*This is the first part of three papers which describes the author’s noncommercial software “VisuMatica”, its **2D- and 3D-nonlinear transformational abilities and their didactic potential**. As result, the software becomes a powerful tool, which helps to discover the unity of mathematics, to visualize and dynamically explore new mathematical environments and phenomena. In particular, the paper includes discussion of nonlinear space transformations’ application to studies of algebra, complex analysis, vector fields, differential equations and modeling.*

## 1. Introduction

Commercial Dynamic Geometry Software<sup>2</sup> (DGS) provides a user-friendly interface for construction of 2D and 3D geometric objects and manipulation with them. Both activities implicitly or explicitly widely utilize *affine transformations of these objects*. Very often different objects (figures) *transformed differently in the same model*.

One can say with certainty that the *smooth and gradual* figures’ transformation is the main mechanism and the main characteristic of these programs that ultimately led to their common name “*Dynamic*”....

Is the smooth and gradual figures transformation adequate to the mathematical notion of transformation? – *Not exactly*. Transformations are functions. Therefore, they are one-step actions defined as a correspondence. Thus, nobody thinks about quadratic function and its graph as a “result” of smooth and gradual “deformation” of straight line (axis  $x$ ) to the parabola shape. Opposite, domain-line becomes range-parabola in a one-step transformation.

The only correct meaning of DGS as dynamic system is the possibility to manually change and deform the studied model.

As a rule, *transformations are not processes (series of actions)*. They are processes and their actions are “changeable” in time in a specific case of *dynamical systems* we will discuss later in the later in the *following two papers of this issue of EJMT*.

---

<sup>1</sup>See *colored figures and animations*: <https://sites.google.com/site/nonlinearspacetransformations/>

<sup>2</sup>Cabri® II Plus, Cabri® 3D, The Geometer's Sketchpad®, Cinderella etc. *VisuMatica* provides *dynamic* processing, but the *geometry* subjects do not limit it. It supports deep studies of different mathematical courses and subjects, in particular, the one, discussed here.

## 2. Nonlinear transformations

In “the "real" world problems and issues on the frontiers of modern scientific, technological, economic, and social research are often nonlinear in nature. In this nonlinear world, many of the mathematical concepts and tools learned and applied in traditional undergraduate, and even graduate, science courses are simply inadequate and new mathematical tools must be introduced.” [4].

In general, the demarcation line between the world of linear and nonlinear phenomena based on the following fact: at the same increment of the independent variable, a nonlinear function responds differently depending on what value given an increment, while linear function responds identically. A fundamental property of nonlinear functions is the almost complete indifference to changes of some values of the independent variables and increased sensitivity to changes of other values of the independent variables.

Nonlinear phenomena described by nonlinear mappings or nonlinear differential equations. *Not formally*, we classify nonlinear spatial transformations (NLST) supported by *VisuMatica* by the:

- Space dimension (**1D, 2D, 3D**);
- Underlying computational domain (**R, C**);
- Transforming action (**One-Step, Continuous, and Discrete**)

Table 1 illustrates this arrangement by some examples (*in black*) and suitable notions.

Visualization of spatial transformation is radically different from visualization of figure transformation. In the case of transformation of figure both the figure and its image are located in the same domain and can be shown and explored there, while space transformation “changes” the space itself and therefore assumes simultaneous presentation of both spaces – domain and range. As a result, the regular Cartesian way becomes unusable in case of more than one-dimensional domain<sup>3</sup>.

*VisuMatica* solves this problem by presenting the Domain and Range in two separate “synchronized” views.

What are the students’ mental activities while studying NLST, which the software has to help to interiorize? Mainly, they are the same as in case of any other transformations: NLST, being functions, require consolidation of the following two complementary actions:

- ❖ *Finding image of an element or a subset of the Domain,*
- ❖ *Finding preimage of an element or a subset of the Range*

Although these activities are the most important and their assimilation includes inter alia mastering mapping, fixed point and root finding etc. educational software has to support deeper and wider exploration of specific and common features and behavior of the studied NLST.

---

<sup>3</sup>Cartesian coordinate system allows showing  $f: \mathbf{R} \rightarrow \mathbf{R}$  functions by duplicating space dimensions:  $x$ -axis presents the function’s Domain and  $y$ -axis – its Range. In this manner, for visualizing 2D or 3D space transformation one needs  $2 \times 2 = 4$  or  $3 \times 3 = 9$  dimensions.

The following case studies present the *VisuMatica*'s support of these pedagogical tasks. We selected these simple and popular examples to emphasize the strength of educational software as a tool for deep and wide exploration of the studied subject that allows students to discover new and surprising features “everywhere”, even in a “boring” and/or well-known math.

		One step	Continuous-time	Discrete-time
1D	$f:S \rightarrow R$ $f:R \rightarrow R$	$y = \sin x$	$\dot{x} = \sin x$	$x_{n+1} = ax_n(1 - x_n)$
2D	$f:S \rightarrow R^2$ $f:R^2 \rightarrow R^2$	$f:(x,y) \rightarrow (2(x+y), xy)$	$\dot{x} = y$ $\dot{y} = a(1 - x^2)y - x$	Orbit
2D	$f:S \rightarrow C$ $f:C \rightarrow C$	$w = z + \frac{1}{z}$	$z' = \sin z$	Orbit, Fractals
3D	$f:S \rightarrow R^3$ $f:R^3 \rightarrow R^3$	$f:(x,y,z) \rightarrow (-y,x,-zy)$	$\dot{x} = y$ $\dot{y} = a(1-x^2)y - x + b \cos cz$ $\dot{z} = c$	Orbit, Poincaré Map

Table 1

### 3. One-step NLST

#### 3.1. 1D mapping

In this simplest and easiest case, it seems that nothing special should be done in addition to regular math studies at secondary and high school. Students have been studied graphs of linear and nonlinear, e.g. quadratic, functions. They made a lot of exercises of finding function values  $f(a)$  and roots of equations  $f(x) = b$ . The mentioned activities of finding image and preimage of transformation already consolidated.

Students are familiar with the term “linear” and notions of “linear function”, “linear dependence” and manage them acceptably. All the rest of math world automatically relates in their opinion to something “nonlinear”, as its counterexamples.

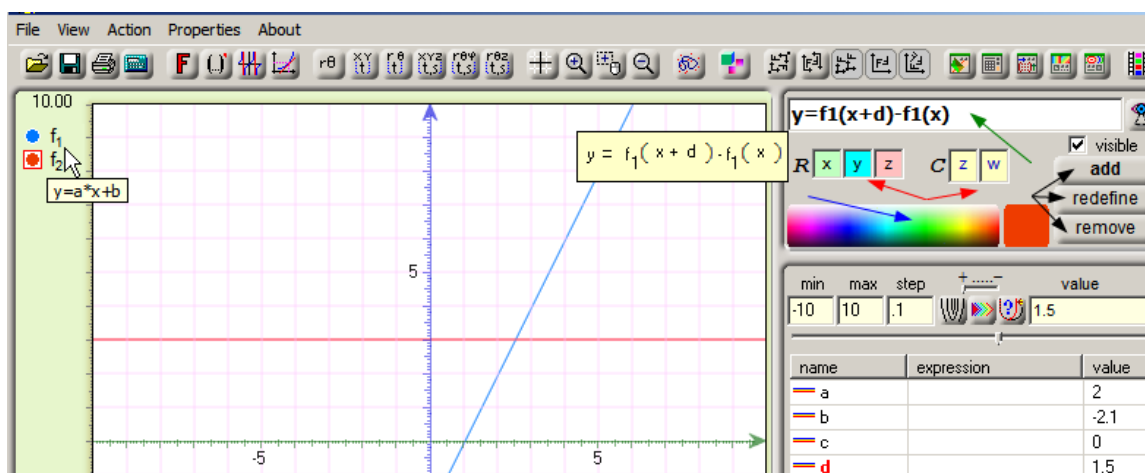


Figure 1

However, the “*nonlinearity*” is not just antonym of “*linearity*”. Their qualitative difference, its mathematical sense should be explicitly emphasized. It can be done by means of a model, that shows graphs of a general linear function  $y = ax + b$  and a dependent, difference function  $y = f_1(x + d) - f_1(x)$ .<sup>4</sup>

Playing with the values of parameters (Fig.1) students discover the fact the model was intended for: *the red line – graph of function  $y = f_1(x + d) - f_1(x)$  - always remains parallel to the x-axis*. For every linear function the difference  $f_1(x + d) - f_1(x)$  does not depend on the value of *independent* variable  $x$ !

The analytic prove of this visually discovered feature:

$$f_1(x + d) - f_1(x) = (a(x + d) + b) - (ax + b) = ad$$

brings them also to clear understanding of its value as a product  $ad$  and the already revealed independence of the difference on  $b$ .

Now we proceed to observation of the difference function’s behavior in case of nonlinear  $f_1(x)$ . Students are encouraged to override  $f_1(x)$ . Replacing  $f_1(x)$  to  $y = ax^2 + bx + c$  by pressing the “*redefine*” button while keeping the  $f_2(x)$  unchanged, they get the model, shown in Fig.2.

After completion of the following assignment

- Does the difference (red graph) depend on the value of  $x$ ?
  - How it depends on the parameters?
- When it becomes constant?
  - Is it still a quadratic function?
- Verify your conclusions analytically.

students are asked to override  $f_1(x)$  once more, with not a polynomial, say,  $y = a/x$ , and to complete similar tasks.

Thus, they become familiar with the difference between “linear” and “nonlinear” functions not only by the externally visual difference of their graphs, being straight and none straight lines; but also with much deeper exclusive linearity feature of independence of the  $f(x + d) - f(x)$  on the value of its argument.

---

<sup>4</sup>*VisuMatica* supports this notation. Students prompted on it by the Legend (see the indexed expressions “ $f_1$ ”, “ $f_2$ ” in the upper-left corner of Fig.1).

*VisuMatica* provides a common simple way to **add** mathematical objects (explicit and implicit functions, equations (including differential equations), inequalities, vector fields etc.):

- Select the proper real/complex names of variables (Fig.1, red arrows).
- Select the color from a palette (Fig.1, blue arrow).
- Type the object’s definition (expression) in the Main edit box (Fig.1, green arrow).
- Press “Enter” key on the “*add*” button (Fig.1, black arrow).

In order to **redefine** an existing object:

- Select it by clicking on its icon in the Legend (selected objects become red colored).
- Change correspondingly its variables’ names, color, and expression.
- Press the “*redefine*” button (Fig.1, black arrow).

To **remove** an object, just select it by clicking on its icon in the Legend, and press the “*remove*” button (Fig.1, black arrow).

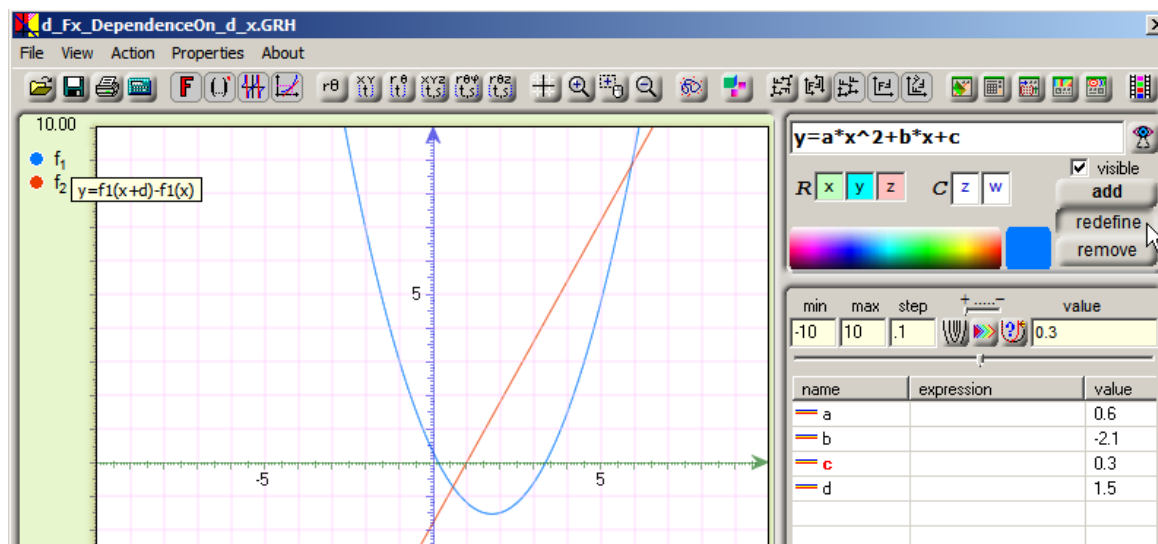


Figure 2

Going further, we can change *a little bit* the function  $y = f_1(x + d) - f_1(x)$  of absolute difference to the relative one  $y = \frac{f_1(x + d) - f_1(x)}{d}$ . With proper leading questions and concentration of students' attention to the case of small  $d$  value, the exploration of the model may serve good propaedeutic to the fundamental notion of "function derivative".

### 3.2. 2D mapping

#### Real plane

Students meet the linear case of space mapping in secondary and high school. They study this concept deeper in courses of linear algebra and analytical geometry, but often with insufficient functional interpretation.

The elementary math explanation of 2D mapping can be done and fruitful already in connection with the very first nonlinear subject of quadratic equation [6].

The Viète theorem, studied at school, presents the link between roots  $x_1, x_2$  of the monic quadratic equation  $x^2 + bx + c = 0$  and its coefficients  $b, c$ . The root-to-coefficient conversion  $V: (x_1, x_2) \rightarrow (b, c)$  or  $V: (x_1, x_2) \rightarrow (- (x_1 + x_2), x_1 x_2)$  delivers an example of 2D mapping. Let us observe its model (Fig.3) and exploration very shortly (for detailed analysis of this map and its sequel see [6]).

*VisuMatica* presents the map in two separated views. The left one shows the Domain: the *space of roots*. Its coordinate axes are  $x_1$  and  $x_2$ . The right view shows the Range: the *space of coefficients*. Its coordinate axes are  $b$  and  $c$ . Color-coding of points in the range reflects the color-coding of their preimages in the domain. The color-coding of maps in *VisuMatica* is subtle because the maps are not 1-to-1 in general. As a result, points of different color in the domain that share the same image compete in imposing their colors on the image point. The application of a mapping to an object in the Domain view appears interactively in the Range view. The mouse pointer interpreted by *VisuMatica* as a point in the view.

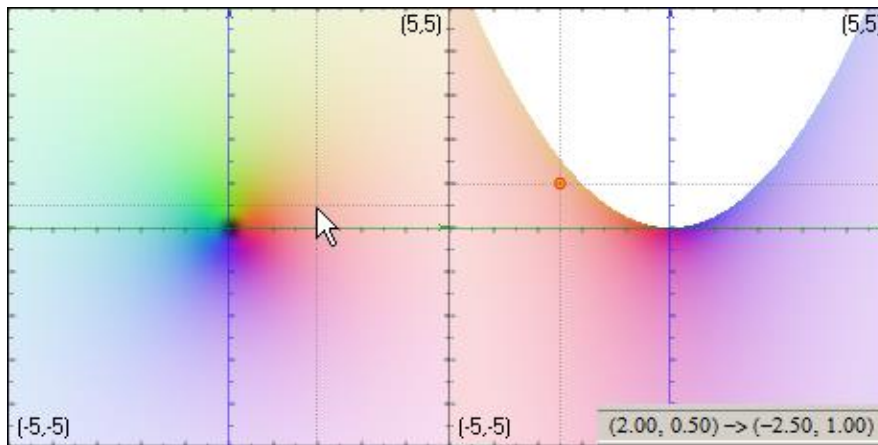


Figure 3

The moving mouse in Domain gets its automatic image in the Range view. Thus, Fig.3 shows mouse pointer located at  $(2, 0.5)$  and its image (the small red circle in the Range view) located there at  $(-2.5, 1)$ .<sup>5</sup> The correspondence also explicitly presented on the *VisuMatica*'s Status Bar (its fragment shown in the bottom-right corner of Fig.3).

*VisuMatica* automatically shows preimage/s of the point of Range that lies under moving mouse pointer. The meaning of this fact is very impressive in the students' sight: it automatically solves *every possible* quadratic equation, each point of the Range space presents. Just locate the mouse pointer at the colored point in Range view, whose coordinates are coefficients of the expression, and you immediately get solution as a preimage point/s in Domain - remains only to interpret its/their coordinates as  $x_1$  and  $x_2$ .

The simple observation of the model immediately arise many questions. Clearly, the  $V$  map hits not all points from the coefficient-space. For instance, the point  $(0, 4)$  apparently has no preimage in the root space. *Why?* The boundary between image points and non-image points seems to be a parabola whose vertex is at  $(0, 0)$ . *Is it really a parabola? What quadratic polynomial is represented by this point?*

Students discover interesting "behavior" of preimages of the mouse while moving it in the Range view. - *There are two points-preimages symmetric with respect to bisector of the 1<sup>st</sup> and 3<sup>rd</sup> quadrant, if mouse pointer located at an internal point of the colored area. These two points coincide, when mouse located at the border of this area. They disappear when mouse pointer enters the white area.*

Mouse pointer in Fig.4 located at  $(2, -3)$  and its preimages are  $(-3, 1)$  and  $(1, -3)$ . Students easily explain this fact and are able to generalize the explanation. Introduction of bisector  $x_2 = x_1$  confirms correctness of the observation. The image of bisector is the boundary of colored area in the Range view. With reference to the origin of this curve, students are hinted to replace equal  $x_1$  and  $x_2$  in the Range expression  $(-(x_1 + x_2), x_1 x_2)$  with  $x$ . ... Yes, it is a parabola and they got its equation!

<sup>5</sup>Points closed around the mouse pointer and its image have the same color tint

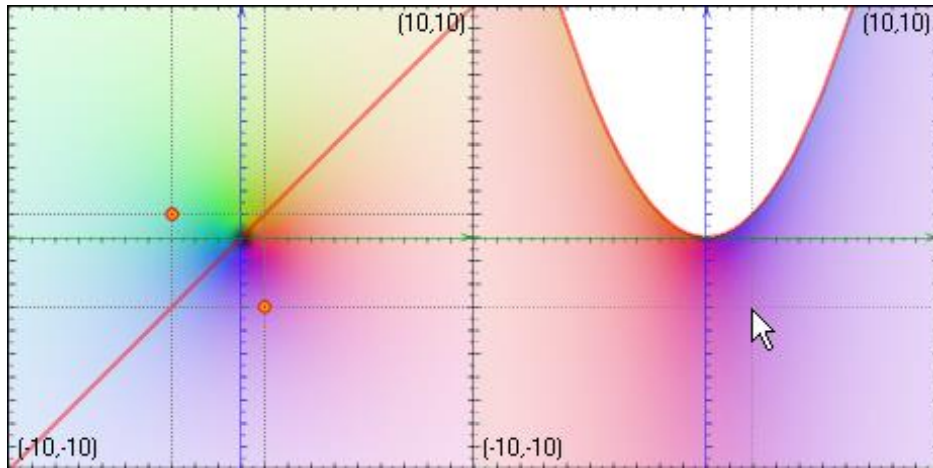


Figure 4

*VisuMatica* provides a rich set of *tools* for featuring and exploration of 2D mapping by means of *mapping dialog box* (Fig.5 shows its tabbed panels) in both cases of  $\mathbf{R}^2$  and  $\mathbf{C}$  Domain. The following study of transformation  $V$  shows a few examples of educational potential of these tools.

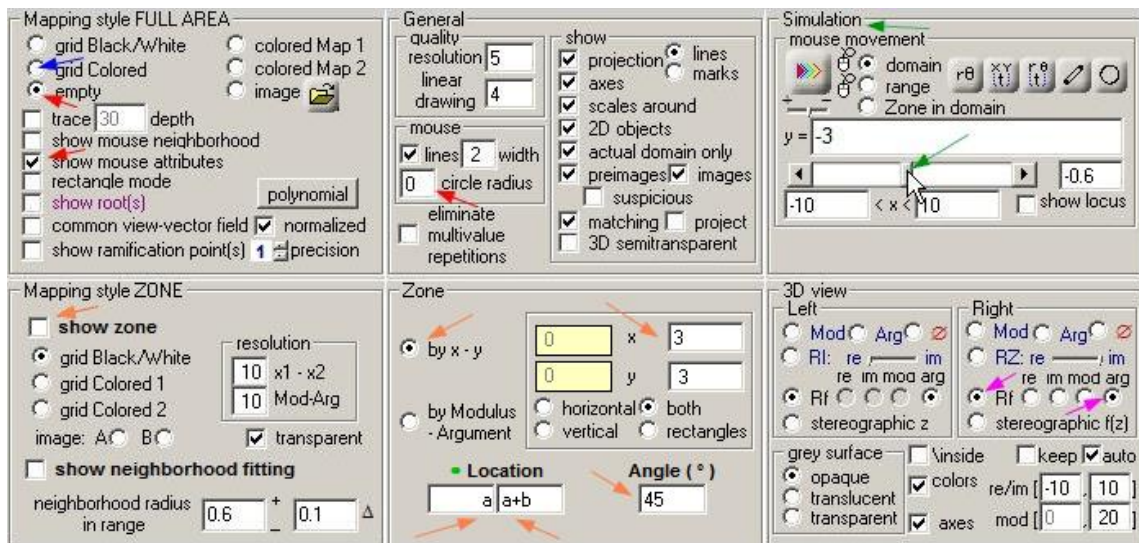


Figure 5

To verify the nonlinearity of our mapping we select the “*grid colored*” mapping style (see the blue arrow in Fig.5). It brings up the view, shown in Fig.6. Surprisingly, the images of horizontal and vertical grid lines also are straight lines.

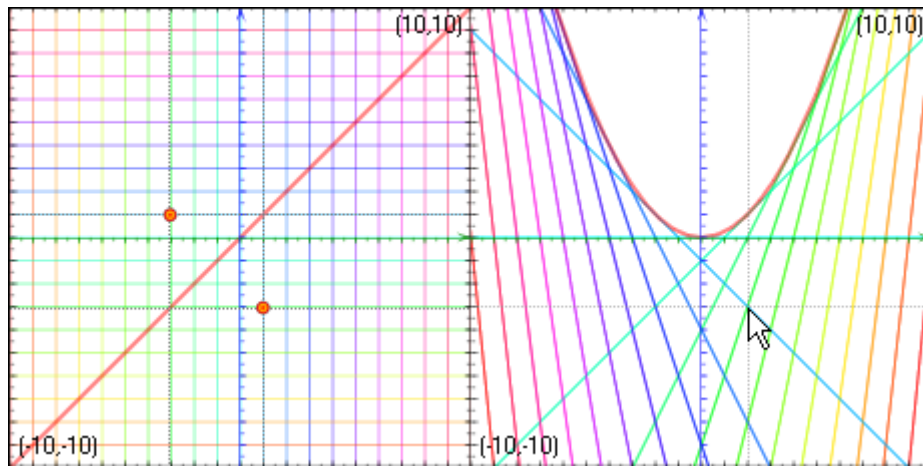


Figure 6

It is rather difficult to link the definite grid line with its image and get some deeper insight about their dependence, although both of them have the same color. Furthermore, it is desirable to check if our conclusion about images of the grid lines remains correct for each horizontal and vertical line in Domain. To make it clear one can set the mapping style as “empty” and enable option “show mouse attributes” with zeroed “circle radius” (see the red arrows in Fig.5).

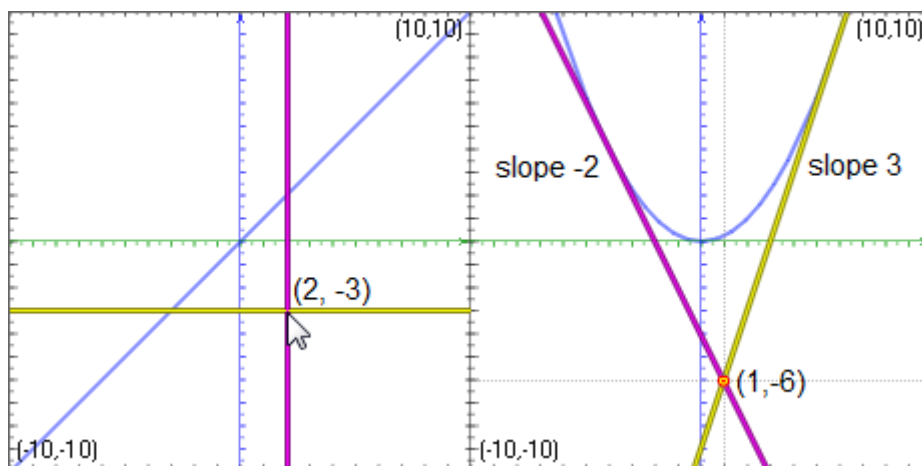


Figure 7

Moving mouse pointer in the Domain view brings up an enlightening dynamic scene. Our model shows horizontal and vertical lines, passing through the point of mouse location, and their images in Range view - straight lines, that intersect at the mouse image and are tangent to parabolic border. Fig.7 presents the case of mouse, positioned at  $(2, 3)$ . Clearly, its image located at  $(1, -6)$ . *But why the slopes' values are opposite to the roots 2 and -3? Why the image lines are tangent to parabola? If so, what will happen while relocating mouse along the yellow line, along the purple one?* We can answer the last question by application of the “simulation” tool that simulates the mouse movement in exact manner instead on manual repositioning (Fig.5, green arrows). Nevertheless, *is the mapping linear?* Although we have an example of transforming straight line  $x_1 = x_2$  to the parabola-shaped curve, we will check the suspicion that  $V$  is nonlinear by the feature of *non-keeping the difference of the V-values.*



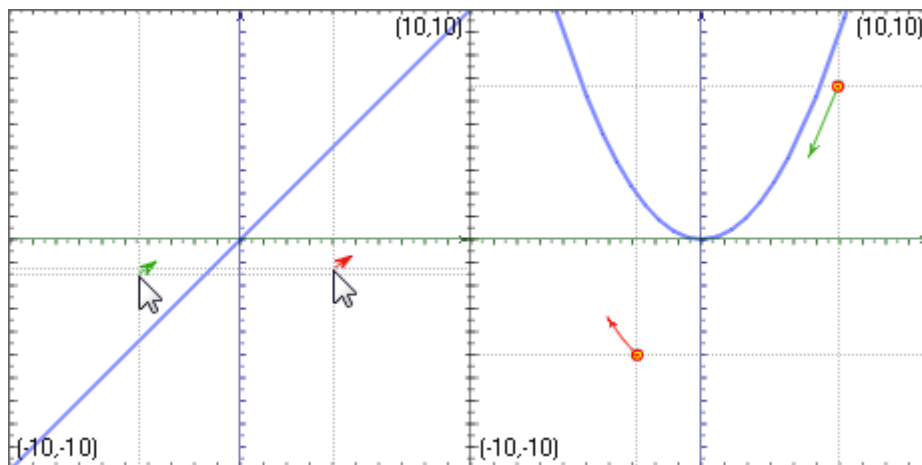


Figure 8

The model consists of a segment-arrow, that connects a free point  $P$  with a point, defined as  $Q(P.x + a, P.y + b)$ . Parameters  $a$  and  $b$  define the difference  $(a, b)$  of independent Domain's variable (the segment). Moving manually point  $P$  in the Domain view we convict, that the difference of the points' images changes drastically: the image of an arrow not obviously remains a segment, it changes (length and direction). Thus, Fig.8 presents two different positions of the dragged segment (colored in red and green), and their correspondent images. It “proves” the nonlinearity of  $V$  mapping.

Observe Fig.4 and Fig.6 once more. *Why the color palette of the Range view in Fig.4 does not include the whole spectrum of Domain?* Convince that there is 1-to-1 correspondence of the palette under bisector and the whole Range area. Counting the lines in Fig.6 brings another surprise: half of lines disappeared under the mapping. *Why?*

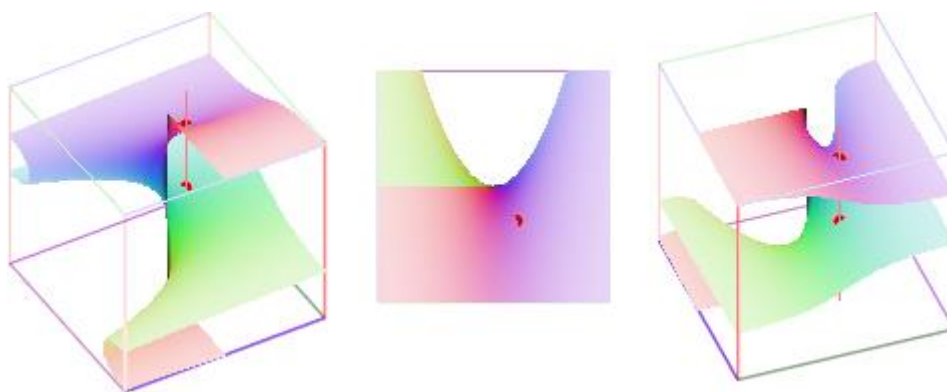


Figure 9

One can get some additional visual insight on what happens in  $V$  mapping by selection of 3D view with Riemann function of Domain's argument (see [6] for details) from the *mapping dialog box* (the magenta arrows in Fig.5). The surface is "synchronized" with mouse and keyboard activities and emphasizes the *overlapping effect of colors and lines* (Fig.9 shows different surface views; the second one is a self-explanatory top view).

The last experiment in the presented series refers to the only example of deforming bisector. Is it really an exclusive case? The simplest way to inspect the problem is to define a line by two free points in Domain. Now remains to move manually these points or the whole line<sup>6</sup> while paying attention to the changing line's image. These activities lead students to few important conclusions:

- The line images "mostly" are looking as parabolas.
- The image becomes straight line not only in case of horizontal or vertical orientation of the original line, but also in case of this line is parallel to the bisector of 2<sup>nd</sup> and 4<sup>th</sup> quadrant.
- The shape of image remains unchanged while dragging the initial line.
- Each one of the images has and only one common point with the parabolic boundary of the colored area, definitely, except of the bisector's image.

The following experiment will help students to understand an analytic explanation of these facts. We have discovered two special directions of bisectors and the constancy of image's shape while dragging the line-preimage. Fortunately, *VisuMatica* includes a special moving "zone" tool (see orange arrows in Fig.5) for 2D mapping exploration that fits father study<sup>7</sup>. Let us orient zone along the suspicious direction of 45° and drag it (Fig.10).

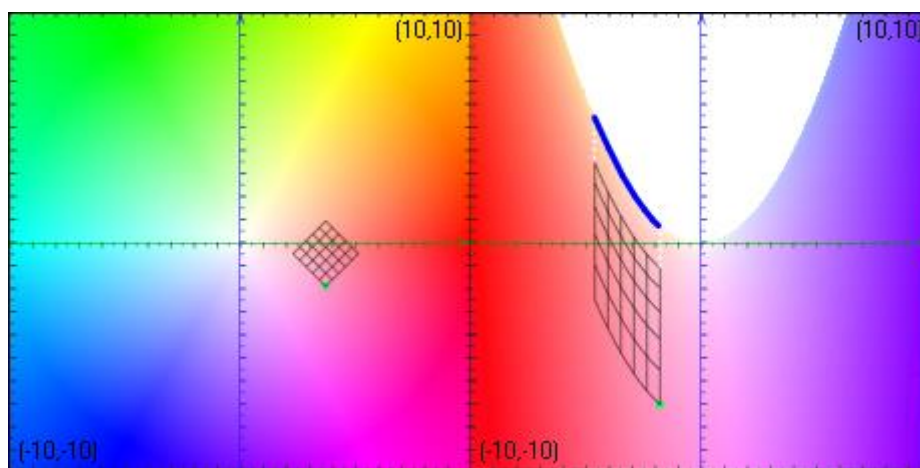


Figure 10

One can notice that the images of grid segments, that are parallel to the bisector of 2<sup>nd</sup> and 4<sup>th</sup> quadrant, are distributed equidistantly (being vertical segments), and their distance remains unchanged. Whereas, the images of grid segments, that are parallel to the bisector of 1<sup>st</sup> and 3<sup>rd</sup> quadrant, all congruent to the blue fragment (drawn by hand) and can be considered as result of its translation in a vertical direction. Accordingly, all the vertical segments in the image have the same length.

Taking into account the *Cavalieri principle* and all these notices we conclude that the area of the zone's image remains the same while moving zone in parallel to the bisector of 1<sup>st</sup> and 3<sup>rd</sup> quadrant. One can enforce this movement by definition of the zone location coordinates parametrically, say

<sup>6</sup> *VisuMatica* keeps the slope of a manually dragged line unchanged.

<sup>7</sup> The idea of mapping examination by observation of the image of some area of the Domain space has a long history in mathematics science, especially in Complex analysis, and education.


$(a, a + b)$  – see orange arrows of “Zone” panel in Fig.5 - and changing  $a$  value by slider or by clicking the “animation” button .

Fig.11.a presents a combination of images of zone, moving in parallel to bisector.

In general, we have gained knowledge on an important feature of  $V$  mapping: it is *conservative for any figure moving in parallel to the bisector* (Fig.11.b presents combination of images of a triangle, which moves in parallel to bisector).

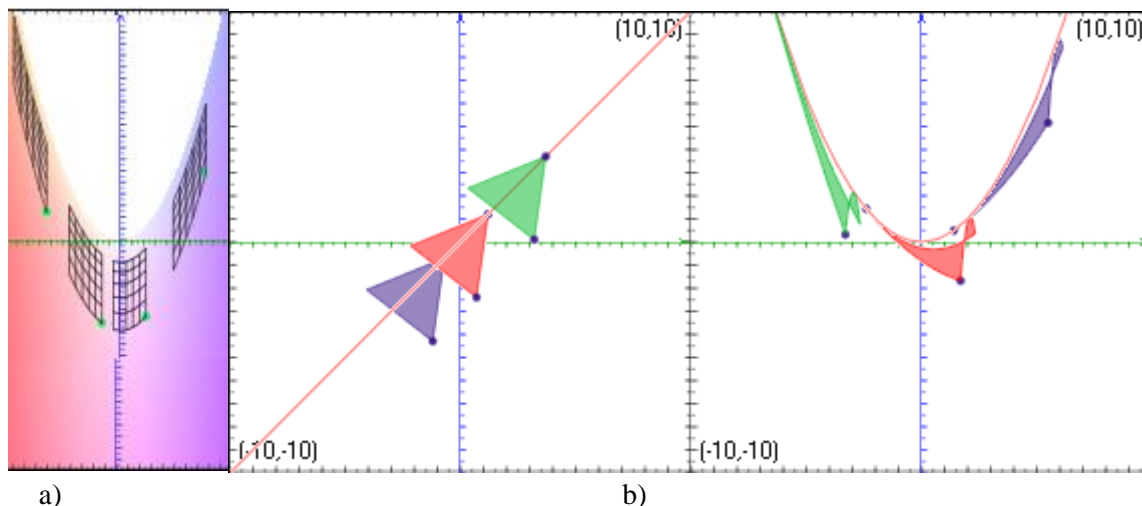


Figure 11

## Complex plane

The theory of functions of complex variable deals mainly with NLST of  $f: \mathbf{C} \rightarrow \mathbf{C}$ . All the elements of *VisuMatica*'s mapping resources are applicable in complex case.

Let's start study the complex Domain from creating a very simple tool for checking linearity by analogy with the  $f: \mathbf{R}^2 \rightarrow \mathbf{R}^2$  case. There are at least three options to construct such tool.

First way: we define two complex numbers  $c_1$  and  $c_2$ , say  $c_1 := a + b*i$ ,  $c_2 := c + d*i$  and the function as  $w = c_1*z + c_2$ . It remains to add a segment-arrow to our model in the same manner as in the  $V$ -mapping case, and the tool is ready. Dragging the  $P$  point with the mouse brings up an opposite result to the one shown in Fig.8. The image of arrow remains the same. After any change of parameters  $a, b, c, d$  the dragging does not change its direction and length. So, our transformation of complex plane  $w = c_1*z + c_2$  is linear. Replacing the expression by some other, say  $w = z^2$ , causes instability of the dragged arrow's image.

The second way is just to enable the “*show mouse neighborhood*” option (see “*Mapping style full AREA*” panel in Fig.5). Initially, it is intended to check the conformality of complex mapping, but can serve our needs too.

A map is conformal if it preserves angles. *VisuMatica* visualizes the feature of conformality by a star in the Domain view and its “image”<sup>8</sup> in the Range view. The original star has a center at mouse

<sup>8</sup> The star-image is symbolic: its segments are always shown as linear. They visualize exclusively direction and the ratio of scale in this direction in the neighborhood of  $f(z)$ , when mouse points to  $z$  and not the result of mapping the star-tool as any other geometric figure.

pointer's location and twelve equal segments colored differently. Angles between adjacent segments are equal. The starting red segment directed to the right and ends with an arrow. Mapping is conformal if and only if the “image” of the star is also a star with equal angles between adjacent segments. The image star may be differently rotated and equally scaled (optionally) - the rotation angle and the scaling ratio can be different for different mouse pointer's locations.

The *map is linear* if and only if the shape of image of the star remains the same, just moves in The Range view, no matter of mouse pointer's position in Domain (*proof of this statement is a challenging task*). Actually, its shape will be a *constantly scaled and/or rotated original star* in case of linear mapping (Fig.12 shows a star around mouse pointer and its image in the Range, located at the barbel's tip of the mapped butterfly for  $w = c_1 * z + c_2$ <sup>9</sup>).



Figure 12

The third way is the simplest one. It made up of adding a new mapping as  $w = f_1(z + c_3) - f_1(z)$ , where  $c_3$  is some complex number, and observing its behavior with different values of  $c_3$ .

The result of mapping with linear  $f(z)$  will be an empty Range view with a single small red point – the common image of the whole complex Domain's plane (Fig.13) does not matter where the mouse pointer is located. Explanation of this fact as “*the difference  $w$  is the same for every  $z$* ” and as an “*IFF condition that the  $f(z)$  mapping is linear*” becomes a fruitful task. This interpretation becomes “more visible” if we will enable an additional, common Domain + Range view<sup>10</sup> and permit the option of common view as vector field in the “*Mapping style*” panel (Fig.5). The resulting picture (Fig.13 bottom-left) shows regularly distributed preimages connected with their images by vectors. These vectors have a common end *IFF* the mapping is linear. Bottom-right area of Fig.13 shows the common view with enabled option of presenting vector field as “*normalized*” – all the arrows directed to the same point. Naturally, the common view displays the link between  $z$  mouse points to and its image emphasized by a dotted black line.

<sup>9</sup> It can be an interesting visual study of dependence of the angle, scaling ratio and the origin's image location on values of parameters  $a, b, c, d$ .

<sup>10</sup> This - sometimes useful in transformation's studies view - is, mainly, only one view of DGS.

As an example of how software helps in making a complicated subject clear, we will consider Zhukovsky<sup>11</sup> transformation function  $z = z + \frac{1}{z}$ . He had studied this subject while looking for the ideal shape of the aerodynamic profile.

Our model includes the moving zone as set of concentric colored circles (in “Zone” panel Fig.5 was selected the zone definition “by Modulus-Argument”) properly located in the Domain view. Its image (Fig.14) has two remarkable features: the outer red circle got the airfoil shape and images of the rest of circles “cover” the Range space but mostly concentrated very close under the airfoils bottom. Dynamically moving zone with mouse, students find out that these features are not “stable” and a similar picture appears very seldom.

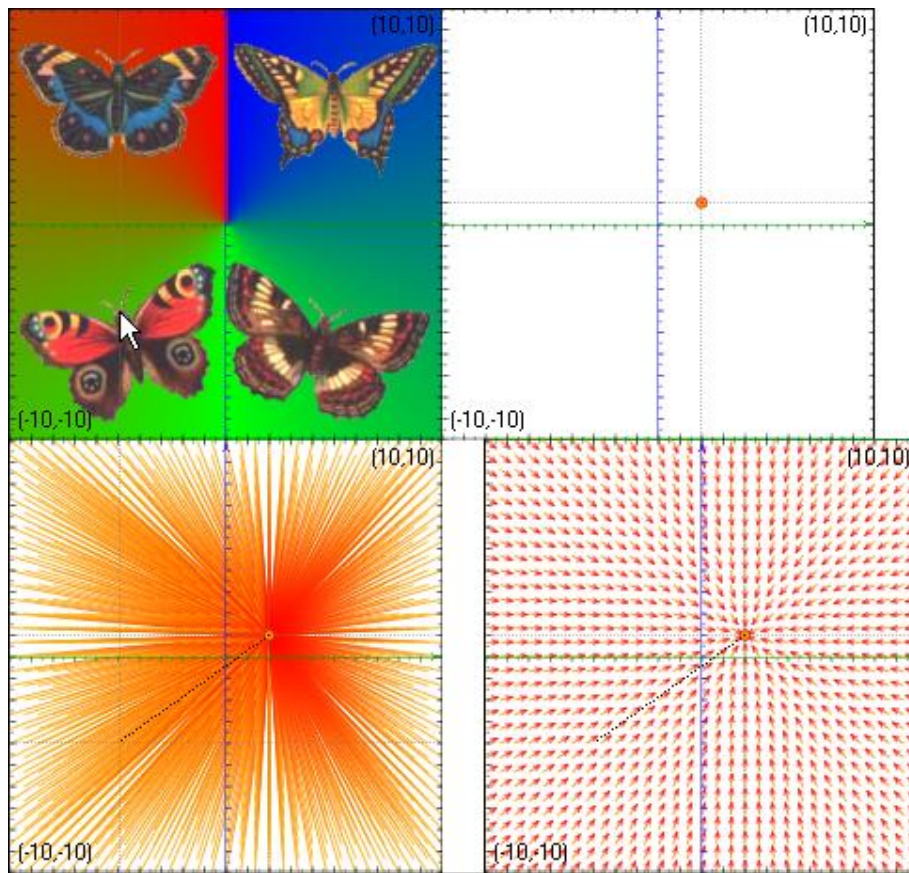


Figure 13

<sup>11</sup> Zhukovsky Nikolai Egorovich (1847 - 1921) was a Russian mathematician and a founding father of modern aero- and hydrodynamics.

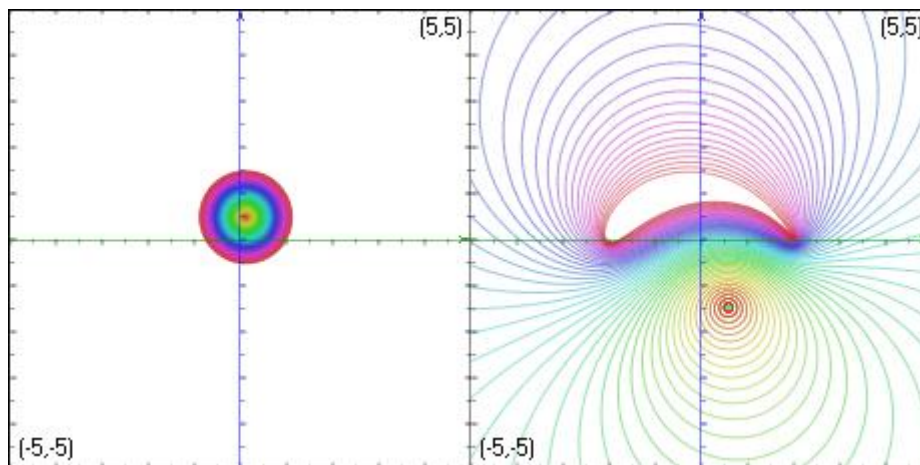


Figure 14

How Zhukovsky could imagine more than a century ago such features that are deeply hidden even for computerized visualization? We will focus on the first feature: the airfoil shape of the circle's image. Foremost, let us look at the map itself. The mapping Range (Fig.15) includes two "strange" points - complex numbers:  $c_1 = -2 + 0i$  and  $c_2 = 2 + 0i$ . They are really "special": by moving mouse pointer in the Range view we can see, that the pointed complex numbers have two preimages. However, when we approach these two points the two preimages become closer and seem to coincide if the mouse points "exactly" to  $c_1$  or  $c_2$ . Keeping in mind that preimages of any complex number  $c$  are roots of equation  $f(z) = c$  students solve equations  $z + \frac{1}{z} = c_i$  and validate their observation that the coincided points in the Domain are  $r_1 = -1 + 0i$  and  $r_2 = 1 + 0i$ . What is so special in values of  $r_1$  and  $r_2$ ? – They are critical points of the function and the function is singular at these points. Really, a point  $z_0$  is critical if the derivative  $f'(z_0) = 0$ ; and it is easy to check that  $r_1, r_2$  are roots of the equation  $f'(z) = 0$ .

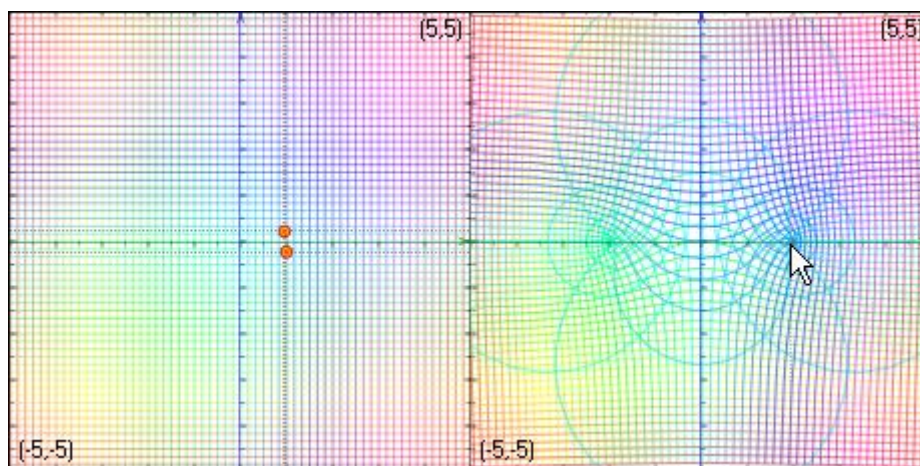


Figure 15

The study of singularity focuses on the function's behavior in the closed neighborhood of critical point. Mainly, on the study of a simple closed (also called Jordan) curve around the point and the

curve's image. We will use a circle in this role. Specifically, we will utilize the familiar “*show mouse attributes*” feature; but redefine it by way of disabling “*lines*” in the “*mouse*” pane of the “*General*” tabbed panel of the Mapping dialog box and setting the “*radius*” equal to the  $a$  parameter. Thus we get a circle of radius  $a$ , with center at the position of moving mouse pointer. By assigning different values to  $a$  parameter and moving the mouse pointer close to  $r_1$  and  $r_2$  (and not only) one can take note of two different types of the circle's image:

- A closed curve with **one self-intersection**, if one and only one of the critical points outlined by the circle (Fig.16 a, b). The curve becomes a lying “8” eight-shaped when the circle has points on both sides of the imaginary axis (Fig.16 b).
- A **simple** closed curve otherwise (Fig.16 (c) – both critical points enclosed with the circle, (d) – both critical points lie outside the circle).

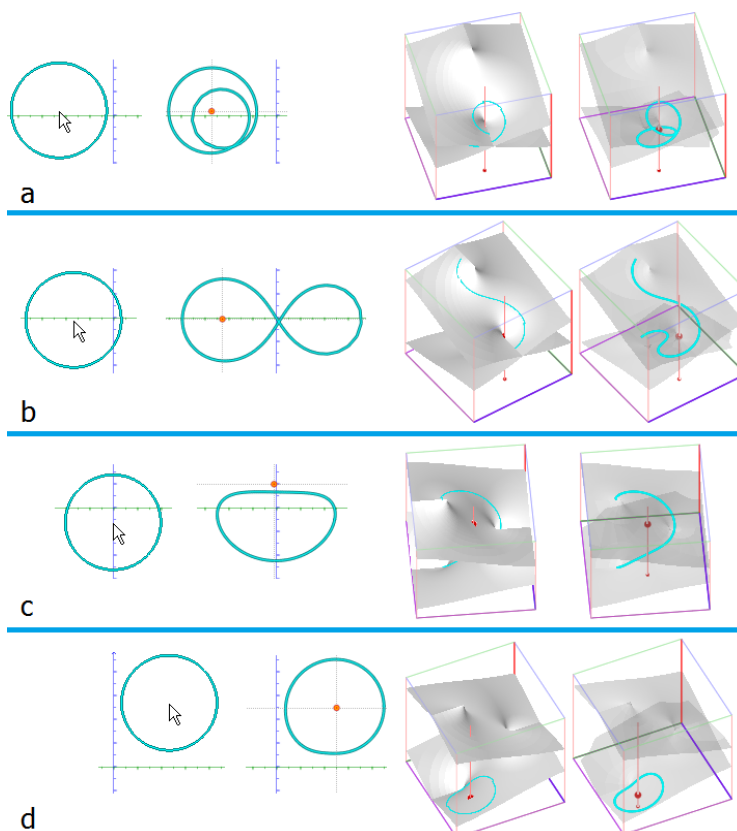


Figure 16

Two columns on the right present the Riemann surface with circle image on it. We added the column with semitransparent surface to help reading curve's behavior. Vertical projection of the curve onto the cube's bottom is always the circle's image in the Range view. One can discover a lot about the behavior of the image-curve from these 3D pictures. However, as always in 3D – you have to rotate the cube with the Riemann surface to perceive its behavior and on this basis the cause of curve-image behavior in each of the four cases.

We will avoid detailed discussion: comparison of the circle's images in the Range view is sufficient to grasp the idea as follows:

On the way from one type of image to another, while moving the mouse pointer, we could see the moment of the image becoming airfoil shaped. Insight comes on the “common

boundary” of mentioned two cases: the transition happens when the critical point *lies on circumference*. Here comes the **cusp** we were looking for!

To verify the correctness of our assumption we will enforce the mouse pointer to move around critical point on a distance of radius  $a$ , that is along a circle of radius  $a$  with a center in the critical point. The “simulation” tool of the Mapping dialog fits our intent.

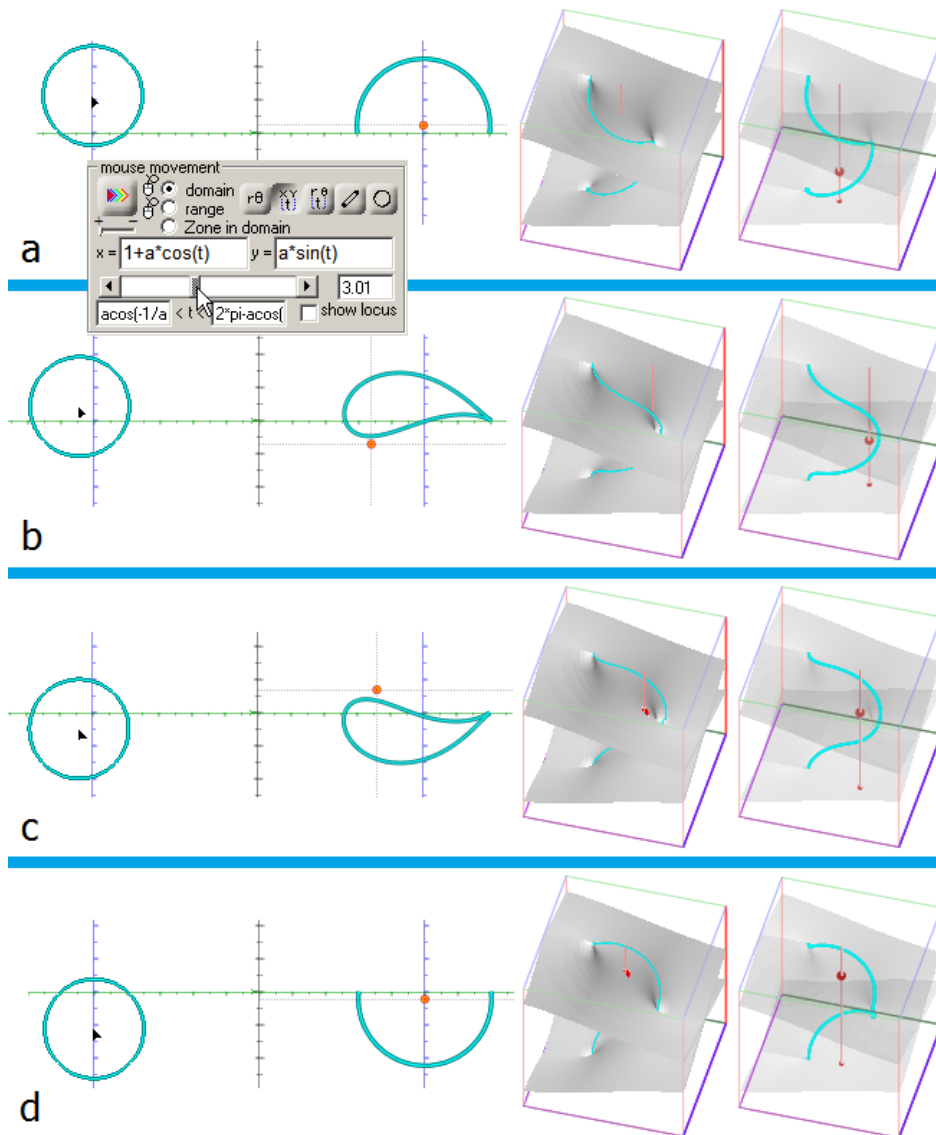


Figure 17

Fig.17 presents simulation dialog’s settings for the enforced movement of mouse pointer with circle of radius  $a = 1.5$  around  $r_2 = 1 + 0i$  and four in-between results.

Common to all these states is the fact that curve starts and ends on the two-leveled “point” that corresponds to  $r_2$  and lies on both layers. It bends around the second critical point (case b, c) producing airfoil shape or passes through this point and degenerates to an arc: projections of both parts of the curve, which lie on different levels of the surface, coincide (case a, d).

Lastly, we must say a word about visualization of multivalued functions’ mapping. Being a problem for professional computer algebra systems, e.g. *Mathematica* (Wolfram Research) [10] it often has



an adequate presentation in *VisuMatica* (see details in [7]). Fig.18 presents *Mathematica* and *VisuMatica* views of  $f(z) = \sqrt{z}$ .

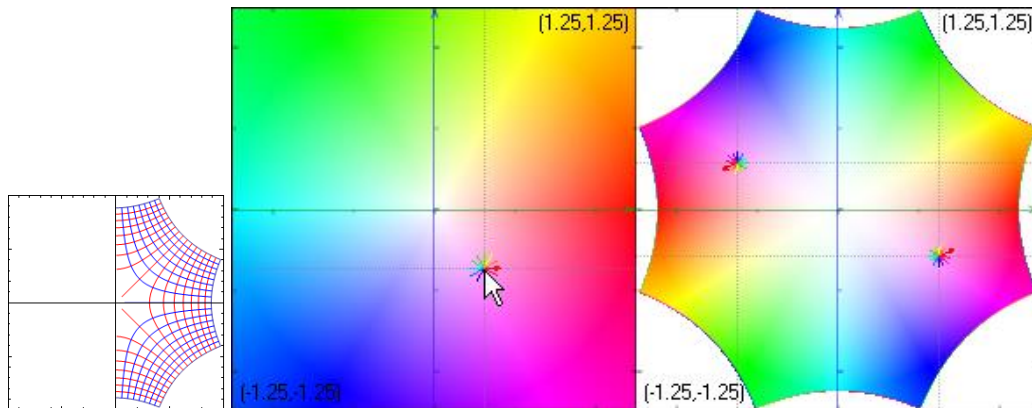


Figure 18

### 3.3. 3D mapping

The specifics of 3D transformations lie in impossibility to visualize the “whole” space or its sub volume as a “filled continuum” like we did in case of 2D by coloring or texturing the “whole” space – Domain view. The reasonable way to deal with the problem is to put some objects (surfaces, curves, polyhedrons and other 3D and 2D figures) into the Domain and observe their images. Relocating, in particular - dragging objects in Domain view and watching the changes of their images helps understand the mechanism of 3D transformation.

*VisuMatica* manages transformations of the whole space in 3D case, including NLST, in contrast to transformations of isolated object(s)<sup>12</sup>, as it is typical in DGS. *VisuMatica* identifies definition of the entered transform and expresses it by two icons: dimensions (2D/3D) and linearity (NLST/linear). Fig.19 shows the “*TRANSFORMATION*” panel of “*Geometry*” dialog with definition of a 3D NLST, where  $g(x) := 3 \sin x$ .

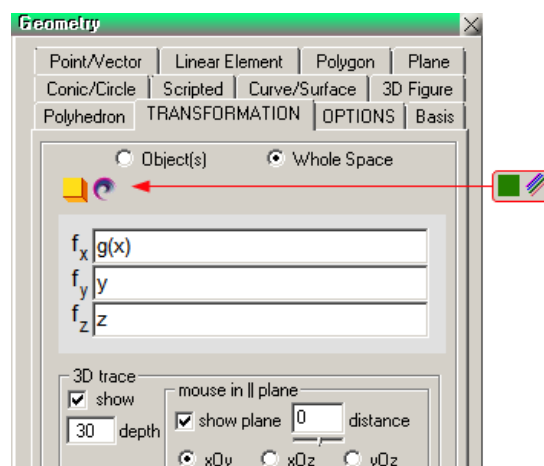


Figure 19

<sup>12</sup> *VisuMatica* supports these transformations too.

Fig.20 (a, b) shows transformation of a parabolic hyperboloid and a sphere (in two different locations) in separated *Domain-Range view*. Fig.20 c) presents cube and sphere (before and after dragging) with their images in a *common view* under  $f(x) = -\frac{yz}{10}, f(y) = \frac{xz}{10}, f(z) = z$  transform.

It is worth to note a simple way of giving dynamics and animations to one-step transformations, by defining them with parameters. Of course, in that case, we will deal with families, rather than one transformation, but the way of constructing and processing such animations may be educationally valuable.

Consider the classical task of deforming a square to a torus. Let the square be perpendicular to the  $y$ -axis, axis-aligned with side equal to 20; and the torus with center at the origin and the major radius  $r$  in  $XOY$  plane has a minor radius  $rr$ . Their parametric definitions look as follows:

$$\begin{cases} x_{square} = t \\ y_{square} = r \text{ and} \\ z_{square} = s \end{cases} \begin{cases} x_{torus} = (r + rr + rr \cos(-s \frac{\pi}{10} + \frac{\pi}{2})) \cos(-t \frac{\pi}{10} + \frac{\pi}{2}) \\ y_{torus} = (r + rr + rr \cos(-s \frac{\pi}{10} + \frac{\pi}{2})) \sin(-t \frac{\pi}{10} + \frac{\pi}{2}) \\ z_{torus} = rr \sin(-s \frac{\pi}{10} + \frac{\pi}{2}) \end{cases}, \text{ where } t, s \in [-10, 10]$$

It is easy to define morphing of the square to torus by linear homotopy

$$\begin{cases} x(a) = (1+a)x_{square} + ax_{torus} \\ y(a) = (1+a)y_{square} + ay_{torus} \\ z(a) = (1+a)z_{square} + az_{torus} \end{cases}, \text{ with } a \in [0, 1]. \quad (1)$$

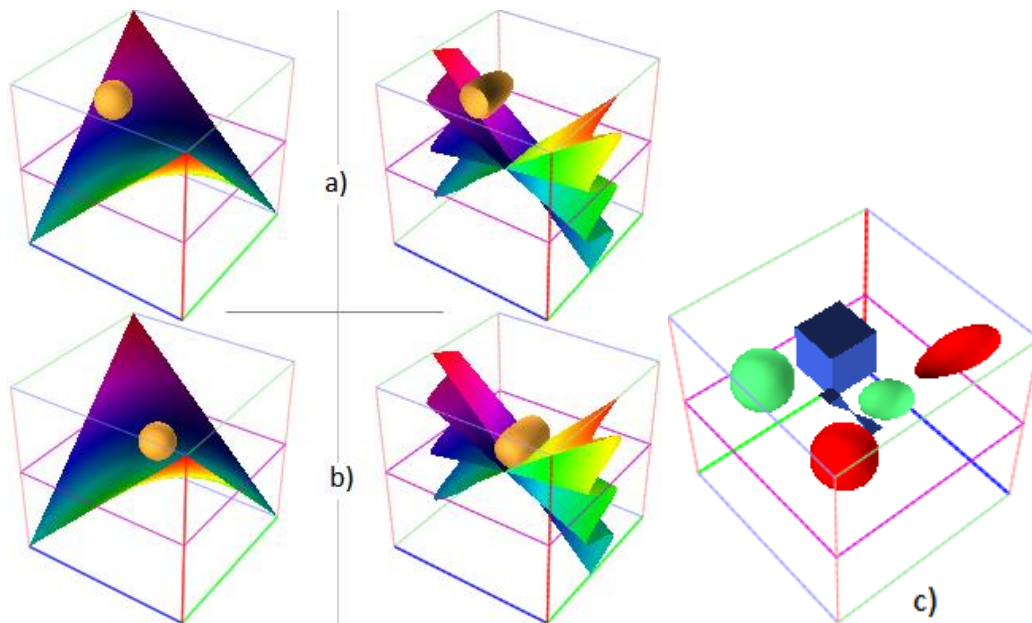


Figure 20

The *surface*, defined in (1) parametrically and dependent on  $a$  parameter, becomes a square when  $a = 0$ , a torus, when  $a=1$  and looks as in-between for  $0 < a < 1$ . Implemented in VisuMatica it produces a nice animation. However, we are looking for whole space, not only surface transformation. In order to get its definition we replace  $t$  to  $x$  and  $s$  to  $z$  in formulas of the square and torus definition, apply them to (1) and enter the resulting strings as  $f_x, f_y, f_z$  into the “*TRANSFORMATION*” panel. From now, the transformation is active. We define our square parametrically as  $(t, r, s)$  with  $t, s \in [-10,10]$ , and play with value of  $a$  parameter. It turns! -It does what it made from, but *how the transformation behaves towards other objects?* After adding a sphere, we discover a not expected result: its image is a projection of a circular disk onto the image of the square for each value of  $a$  parameter. The volume of the sphere’s image always equals to zero. ...*Aha!* We have lost the  $y$  coordinate. It was not necessary because of special orientation of our square ( $y$  is constant). ...*Small fix:* add  $y - r$  to  $f_y$  expression - it equals to zero concerning the square, whose  $y = r$  by definition, but allows breathing to images of other objects. Their volume becomes positive.

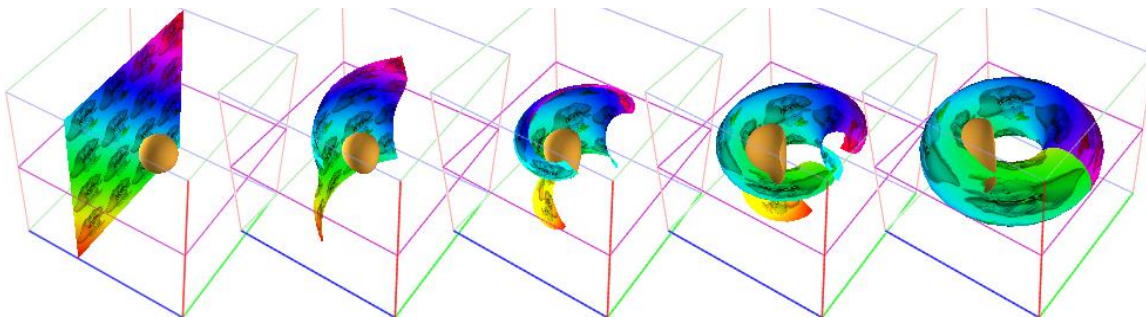


Figure 21

Fig.21 shows our scene with different values of  $a$  (the square defined with a spectrum of colors and an image of Dumbo-Disney’s baby elephant to ease seeing the correspondence).

One can express any space transformation as

$$\begin{aligned} \begin{cases} f_x(x, y, z) \\ f_y(x, y, z) \\ f_z(x, y, z) \end{cases} &= \begin{cases} x + (f_x(x, y, z) - x) \\ y + (f_y(x, y, z) - y) \\ z + (f_z(x, y, z) - z) \end{cases} = \begin{cases} x \\ y \\ z \end{cases} + \begin{cases} (f_x(x, y, z) - x) \\ (f_y(x, y, z) - y) \\ (f_z(x, y, z) - z) \end{cases} \\ &= (x, y, z) + v(x, y, z), \text{ where } v(x, y, z) \\ &= \begin{cases} (f_x(x, y, z) - x) \\ (f_y(x, y, z) - y) \\ (f_z(x, y, z) - z) \end{cases} \end{aligned}$$

and thus to clarify it geometrically as addition of vector  $v$  to point  $P(x, y, z)$ :  $f: (x, y, z) \rightarrow (x, y, z) + v(x, y, z)$ .

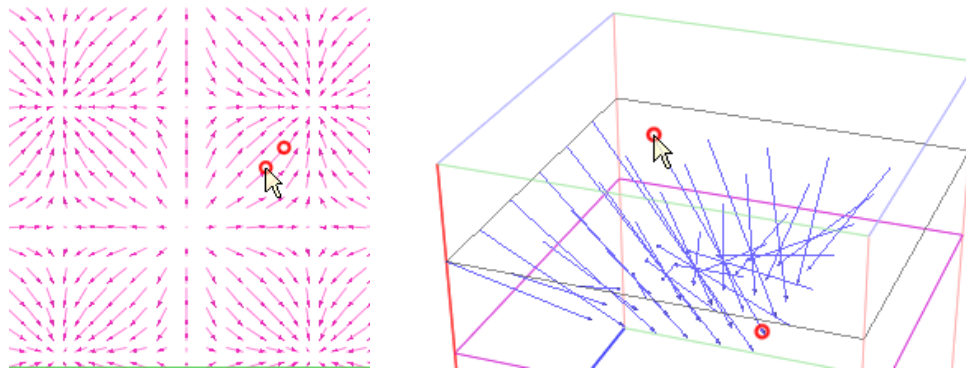


Figure 22

Considering vector  $v$  as a function  $v: (x, y, z) \rightarrow \begin{cases} (f_x(x, y, z) - x) \\ (f_y(x, y, z) - y) \\ (f_z(x, y, z) - z) \end{cases}$ , defined on the whole space  $\mathbf{R}^3$ ,

we deal with a *vector field*. It gives an idea to take advantage of the *VisuMatica*'s vector fields supporting mechanism in our exploration. The notion of *vector field* is crucial for describing differential equations and dynamical systems. Vectors there interpreted as velocity. Mainly, only their slope is taken into account while the magnitude is ignored or scaled to suit the visualization needs. So, considering this circumstance, the interface must be explicitly configured on proper vectors' portray. (Fig.22 shows interpretation of the correspondence in 2D and 3D nonlinear mapping by means of vector fields).

The following two papers present instructive capabilities of *VisuMatica* in studies of continuous- and discrete-time dynamical systems. With mentioned settings of vector's drawing, their appropriate parts can be read as an extension of current 3D mapping analysis.

#### 4. Conclusions

This paper demonstrates a new way in software support of teaching and learning mathematics, in particular, nonlinear space transformations.

Unlike applets and specific models in CAS and DG systems, here creation and setting mechanism have a generic nature. This approach allows emphasize and exploit the multidisciplinary and interdisciplinary links, the unity of mathematics, and thus to ease study of different mathematical subjects.

Distinguishing features of *VisuMatica*:

- Absence of special syntax for definition and configuration of mathematical objects,
- Intensive use of the powerful potential of user interface,
- Attainment of dynamicity by manual manipulations with mouse and sliders,
- Information visualization by colored coding

help students to explore complicated mathematical notions and get insight about their properties and relationships.

## References

- [1] Arnold V.I. Ordinary differential equations. The MIT Press, 1978
- [2] Bertuglia C. S., Vaio F. Nonlinearity, chaos & complexity. The dynamics of natural and social systems. Oxford University Press, 2005
- [3] Devaney R. L. The Fractal Geometry of the Mandelbrot Set.  
[http://is.muni.cz/el/1456/jaro2009/PMAPEM/The\\_Fractal\\_Geom/The\\_Fractal\\_Geometry\\_Of\\_The\\_Mandelbrot\\_Set.html](http://is.muni.cz/el/1456/jaro2009/PMAPEM/The_Fractal_Geom/The_Fractal_Geometry_Of_The_Mandelbrot_Set.html)
- [4] Enns R. H. It's a Nonlinear World, Springer, 2010
- [5] Glendinning P. Stability, Instability and Chaos. An Introduction to the Theory of Nonlinear Differential Equations. Cambridge University Press, 1994
- [6] Katz G., Nodelman V. The Shape of Algebra in the Mirrors of Mathematics, World Scientific, 2012
- [7] Katz G., Nodelman V. Software Tools for Visualizing Multivalued Function. Research Journal of Mathematics & Technology, Vol. 2, Num. 1, 2013
- [8] Marotto F. R. Introduction to Mathematical Modeling Using Discrete Dynamical Systems. Thomson Brooks/Cole, 2006
- [9] Wagon S. New Visualization Ideas for Differential Equations.  
<http://www.cecm.sfu.ca/organics/papers/wagon/paper/html/wagon.html#wagon15>
- [10] Weisstein, Eric W. Conformal Mapping.  
<http://mathworld.wolfram.com/ConformalMapping.html>