Visualization of Two-Dimensional Conformal Transformations with PGF/TikZ

Preprint · March 2019

1 author:

Manousos Markoutsakis
DataDirect Networks Inc.

Visualization of Two-Dimensional Conformal Transformations with PGF/Ti*k*Z

Manousos Markoutsakis

6th March 2019

Abstract

In the first part of this overview article we summarize the basics of D-dimensional global conformal transformations and two-dimensional infinitesimal conformal transformations. In the second part we suggest an elementary method for visualizing two-dimensional conformal transformations in the Euclidean plane. The practical implementation is build on the open source graphics language PGF/Ti*k*Z.

Conformal Transformations and Conformal Group. We consider the D-dimensional *Euclidean affine-linear space* $\mathbb{E}^D \equiv (\mathbb{R}^D, \delta)$ comprised of the D-dimensional affine-linear space $\mathbb{R}^D$ and the usual Euclidean metric δ_{jk} given by the constant matrix

$$\delta_{jk} = \operatorname{diag}(1, \ldots, 1) \ . \tag{1}$$

The points of the manifold $\mathbb{E}^D$ can be represented by D-tuples x^j, $j = 1, \ldots, D$. We consider transformations of the general form $x^j \mapsto x'^j(x)$ on the manifold $\mathbb{E}^D$. All transformations of the points x^j here and in the following are meant in the active sense. Simple examples are *translations* $x'^j = x^j + a^j$ and *rotations* $x'^j = R^j{}_k x^k$ which, combined, constitute the *Euclidean transformations*. The set of all Euclidean transformations on $\mathbb{E}^D$ constitutes the *Euclidean group* denoted $E(D)$. The Euclidean transformations are fully determined by the condition

$$\frac{\partial x'^l}{\partial x^j}(x)\frac{\partial x'^m}{\partial x^k}(x)\,\delta_{lm} = \delta_{jk} \ , \tag{2}$$

which, at the same time, is an expression of the invariance of the metric δ_{jk} of $\mathbb{E}^D$ under Euclidean transformations. General D-dimensional *conformal transformations* on $\mathbb{E}^D$ are defined by the condition

$$\frac{\partial x'^l}{\partial x^j}(x)\frac{\partial x'^m}{\partial x^k}(x)\,\delta_{lm} = \Psi^2(x)\,\delta_{jk} \ , \tag{3}$$

where the so-called *conformal factor* $\Psi(x)$ is a strictly positive scalar function of the coordinates. Strict positivity $\Psi(x) > 0$ is needed so that the inverse $1/\Psi(x)$ is well defined. The conformal transformations constitute a group called the *conformal group* which we denote $C(D)$. Obviously the Euclidean group is a subgroup of the conformal group and corresponds to the special case of constant $\Psi = 1$ in all transformations. In order to prove the group property for the set of conformal transformations we first note that the composition of any two conformal transformations is also a conformal transformation. More specifically consider two iterated conformal transformations F and G given by

$$\begin{aligned} F &: x \mapsto x' = F(x) \\ G &: x' \mapsto x'' = G(x') \end{aligned} \tag{4}$$

with the respective conformal factors $\Psi_F(x)$ and $\Psi_G(x)$. Then the combined transformation $G \circ F$

$$G \circ F : x \mapsto x'' = G \circ F(x) \tag{5}$$

is also a conformal transformation and its conformal factor $\Psi_{G\circ F}(x)$ is given by

$$\Psi_{G\circ F}(x) = \Psi_G(F(x))\,\Psi_F(x) \ . \tag{6}$$

The neutral element is the identity transformation and the inverse conformal transformation is given by the inverse conformal factor. Due to the last formula the associativity of conformal transformations reduces to the associativity of real numbers. Thus the conformal transformations on the Eulidean space $\mathbb{E}^D$ consitute a group. Geometrically the conformal transformations leave the angles between D-vectors unchanged. In general they do not preserve lengths or distances. The general condition for conformal transformations contains the special case where the conformal factor $\Psi(x)$ is a constant, $\Psi(x) = \text{const} = \omega > 0$. This is the case of a global, or rigid, *scale transformation* also called a *dilatation*. Hence we see that general conformal transformations can be viewed as local scale transformations.

In terms of single types of transformations the conformal group contains besides the translations and rotations also the above mentioned dilatations

$$x'^k = \omega x^k \tag{7}$$

and the so-called *special conformal transformations*, abbreviated *SCT*, and given by

$$x'^k = \frac{x^k - c^k x^2}{1 - 2(c\cdot x) + c^2 x^2} \ , \tag{8}$$

where $(c \cdot x) = c_j x^j$ is the usual scalar product in $\mathbb{E}^D$. For a derivation of the SCT expression see e.g. [2]. The dilatations need one real parameter ω for their definition. The SCTs need D real parameters, represented by the vector c^k, in order to be defined. In $D \geq 2$ dimensions a general conformal transformation needs in total

$$D + \frac{D(D-1)}{2} + 1 + D = \frac{(D+2)(D+1)}{2} \tag{9}$$

real parameters to be defined. A particular type of conformal transformation is the *inversion*

$$x'^k = \frac{x^k}{x^2} \ . \tag{10}$$

Any SCT can be written as an inversion, followed by a translation by a constant vector and a subsequent inversion. We can write symbolically

$$\text{SCT}\,[\,c\,] = \text{Inversion} \circ \text{Translation}\,[\,-c\,] \circ \text{Inversion} \ . \tag{11}$$

For each single type of transformation within the conformal group there is a specific conformal factor. The conformal factor for translations and rotations is trivially equal one. For dilatations the conformal factor is

$$\Psi_{\text{Dil}}(x) = \omega \ . \tag{12}$$

For inversions the corresponding conformal factor is

$$\Psi_{\text{Inv}}(x) = \frac{1}{x^2} \ . \tag{13}$$

And finally for SCTs the conformal factor is

$$\Psi_{\text{SCT}}(x) = \frac{1}{1 - 2(c \cdot x) + c^2 x^2} \ . \tag{14}$$

The transformations we have listed above are globally defined conformal transformations. There is a small technicality we need to take into account for inversions and SCTs when defining global transformations. Inversions map the point $x = 0$ to infinity, therefore we need to use the compactified version $\mathbb{E}^D \cup \{\infty\}$ of Euclidean space. For SCTs the points x with $1 - 2(c \cdot x) + c^2 x^2 = 0$ are special. These points are mapped to infinity, so again we have to use the compactified version $\mathbb{E}^D \cup \{\infty\}$ in order to encompass all cases.

Infinitesimal Conformal Transformations and the Case $D = 2$. Instead of globally defined transformations one can consider infinitesimal transformations. Infinitesimal transformations on $\mathbb{E}^D$ have the generic form

$$x'^j(x) = x^j + \epsilon K^j(x) \ , \tag{15}$$

with the real parameter ϵ being considered small $\epsilon \ll 1$ and the vector field $K^j(x)$ being specific for the transformation at hand. For Euclidean transformations as defined in 2 the equivalent condition for the vector field $K^j(x)$ reads

$$\partial_j K_k + \partial_k K_j = 0 \ . \tag{16}$$

This is the *Killing equation* and the vector field is called the *Killing vector field*. For the case of conformal transformations we first write the conformal factor $\Psi(x)$ as

$$\Psi(x) = \exp\left(\epsilon\,\tau(x)\right) = 1 + \epsilon\,\tau(x) + \mathcal{O}(\epsilon^2) \tag{17}$$

using a real scalar function $\tau(x)$. Inserting the transformation $x'(x) = x + \epsilon K(x)$ into the condition for conformal transformations 3 we obtain

$$\partial_j K_k + \partial_k K_j = 2\,\tau \delta_{jk} \ . \tag{18}$$

Taking the trace of both sides the scalar function $\tau(x)$ is derived to be

$$\tau = \frac{1}{D}\partial_j K^j \ . \tag{19}$$

Thus we obtain the *conformal Killing equation*

$$\partial_j K_k + \partial_k K_j = \frac{2}{D}(\partial \cdot K)\delta_{jk} \ . \tag{20}$$

A vector field $K^j(x)$ fulfilling this equation is called a *conformal Killing vector field* and defines an infinitesimal conformal transformation.

Next we specialize to the case $D = 2$. The conformal Killing equation is equivalent to the following two equations

$$\left.\begin{array}{l} \partial_1 K_1 = \partial_2 K_2 \\ \partial_1 K_2 = -\partial_2 K_1 \end{array}\right\} \ . \tag{21}$$

Instead of the real Euclidean plane $\mathbb{E}^2$ we can consider complex coordinates in $\mathbb{C}$ and define a complex function $K(z)$ by

$$z \mapsto K(z) \equiv K_1(z) + \mathrm{i}K_2(z) \ , \tag{22}$$

with the function argument being $z \equiv x_1 + \mathrm{i}x_2$. Obviously the conformal Killing equation is equivalent to the Cauchy-Riemann equations, which in turn are equivalent to the holomorphy of the complex function $K(z)$ in some suited open set in $\mathbb{C}$, see e.g. [4]. This means that *holomorphic functions define infinitesimal conformal transformations*. A function $f(z)$ on $\mathbb{C}$ of the form

$$z \mapsto f(z) = z + \epsilon K(z) \tag{23}$$

is the expression of such an infinitesimal conformal transformation. Because we are foremost interested in the local behavior of functions when we study infinitesimal transformations, we can very well consider even meromorphic functions possessing certain singular points outside the open domain of interest in

$\mathbb{C}$. Hence we can write down the Laurent series for $K(z)$, e.g. around the point $z = 0$. The infinitesimal conformal transformation $f(z)$ reads then

$$f(z) = z + \epsilon \sum_{n \in \mathbb{Z}} \kappa_n \cdot (-z^{n+1}) \ . \tag{24}$$

The minus sign in front of z^{n+1} is just conventional in conformal field theory. We see that one needs countably infinite many parameters, the coefficients κ_n, in order to define a 2-dimensional infinitesimal conformal transformation. This is a speciality of *local* conformal transformations in two dimensions. If one wants to narrow down to *globally* defined conformal transformations, then the number of parameters in two dimensions is equal $(D+2)(D+1)/2 = 6$ again, as given in the previous section. These are two parameters for translations, one angle parameter for rotations, one scale parameter for dilatations and two parameters for the SCTs.

Visualization of Conformal Transformations with PGF/Ti*k*Z. Conformal transformations are important for various branches of theoretical physics, in engineering and in computer vision. However, in the literature one encounters rather seldom graphical visualizations of conformal maps. At the same time it is enlightening to actually see the effect of certain conformal transformations. In the following a simple recipe is suggested how to produce high-quality graphical representations of conformal maps. The practical implementation is based on the free and open source graphics language PGF/Ti*k*Z. For description of this vector graphics software see [3]. We will present the conceptual idea, concrete examples of conformal maps and the corresponding code within the PGF/Ti*k*Z language.

The initial basic idea is to consider the *active transformation* of a regular coordinate grid, for instance a Cartesian coordinate grid, under a conformal transformation. We concentrate on the case $D = 2$ throughout. One starts with a set of curves describing the regular coordinate grid. In the case of a Cartesian coordinate grid these curves would be the set of lines

$$\left.\begin{array}{r} x = c \\ y = d \end{array}\right\} \ . \tag{25}$$

Any point in the Euclidean plane is specified by its Cartesian coordinates (x, y). The real parameters c and d take their values within a suited, discrete set of equidistant values. Under the active conformal transformation

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} u(x,y) \\ v(x,y) \end{pmatrix} \tag{26}$$

each point (x, y) is mapped to the point (u, v). The constant line $x = c$ is transformed to a parametric curve in the plane of the form $u = u(c, y)$ and $v = v(c, y)$, where the variable y takes all real values within the domain considered. In the same way, the constant line $y = d$ is transformed to a parametric curve

$u = u(x, d)$ and $v = v(x, d)$, where the variable x takes all real values within the domain of interest. The set of all these parametric curves visualizes the effect of deformation of the regular Cartesian grid under the conformal transformation. What we visualize in this way is actually the conformal geometry as seen from our usual Euclidean geometry point of view.

Let us give a first example. We start with the regular Cartesian grid as shown in figure 1 with its domain being already chosen suited for the example.

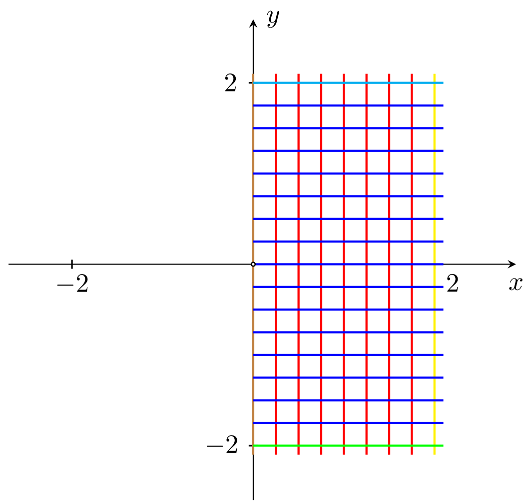

Figure 1: Initial Cartesian grid, occupying two quadrants

The concrete case considered is the conformal map

$$z \mapsto z^2 \ , \tag{27}$$

with z taking values in $\mathbb{C} \setminus \{0\}$. Written with real Cartesian coordinates in $\mathbb{R}^2 \setminus \{0\}$ this conformal map is given by

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} x^2 - y^2 \\ 2xy \end{pmatrix} \ . \tag{28}$$

That this is a conformal transformation, as demanded by the defining equation 3, can be inspected directly. The corresponding conformal factor $\Psi(x, y)$ is

$$\Psi(x, y) = 2\left(x^2 + y^2\right)^{1/2} \ . \tag{29}$$

Each vertical straight line $x = c$ is mapped to the parametric curve $u = c^2 - y^2$ and $v = 2cy$. Each horizontal straight line $y = d$ is mapped to the parametric curve $u = x^2 - d^2$ and $v = 2xd$. Transforming the complete domain of interest, defined by the values $0 \leq x \leq 2.1$ and $-2.1 \leq y \leq 2.1$, one obtains the conformally deformed grid lines as depicted in figure 2. All local angles are preserved, with the origin being the only exception where the angle is doubled. Globally the total Cartesian grid is stretched and bent in such a way that the initial brown line segment $x = 0$ with $y \in (0, 2.1]$ is bent "downward" and the initial brown line segment $x = 0$ with $y \in [-2.1, 0)$ is bent "upward" so that the two resulting lines coincide with the line segment $y = 0$ with $x \in [-4.41, 0)$.

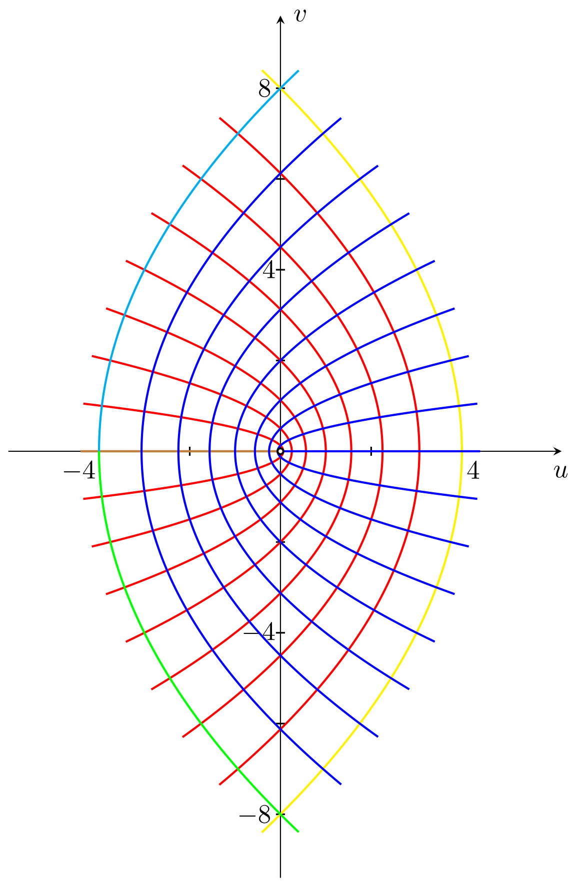

Figure 2: Conformally transformed grid under $z \mapsto z^2$

In full analogy one can consider the conformal map

$$z \mapsto z^4 \tag{30}$$

in $\mathbb{C} \setminus \{0\}$. Written in $\mathbb{R}^2 \setminus \{0\}$ this conformal map is given by

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} (x^2 - y^2)^2 - 4x^2y^2 \\ 4(x^2 - y^2)xy \end{pmatrix} . \tag{31}$$

The corresponding conformal factor $\Psi(x, y)$ is

$$\Psi(x, y) = 4\left(x^2 + y^2\right)^{3/2} . \tag{32}$$

We can start with a regular Cartesian grid in the first quadrant, as in figure 3.

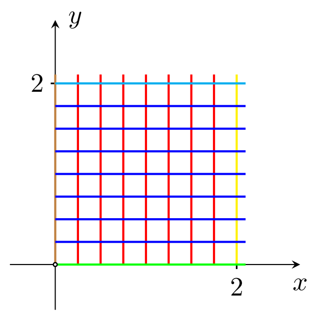

Figure 3: Intial Cartesian grid, occupying one quadrant

The original grid is conformally transformed with all local angles being preserved except at the origin where the angle is multiplied by four. In the large the Cartesian grid is transformed in such a way that the initial brown line segment $x = 0$ with $y \in (0, 2.1]$ is bent along the positive rotation direction sweeping the angle between $\varphi = \pi/2$ and $\varphi = 2\pi$ so that the resulting line coincides with the line segment $y = 0$ with $x \in (0, 19.4481]$. The result of this conformal transformation is visualized by the diagram 4, whos scale has been reduced by a factor of ten compared to the original diagram.

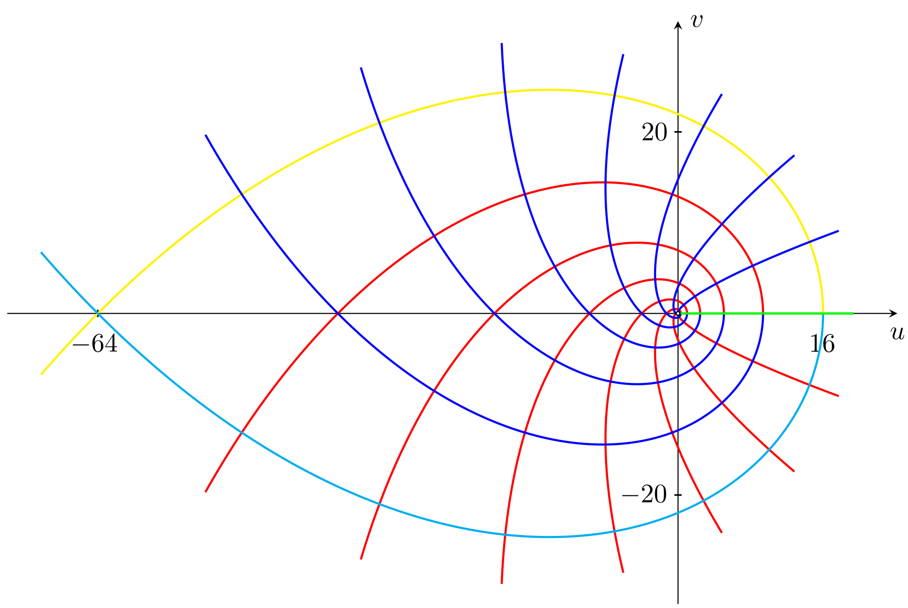

Figure 4: Conformally transformed grid under $z \mapsto z^4$

As a final example we consider the conformal map

$$z \mapsto \frac{1}{z} \ , \tag{33}$$

again defined in $\mathbb{C} \setminus \{0\}$ leaving out the singular point at the origin. Written in $\mathbb{R}^2 \setminus \{0\}$ this conformal map is expressed as

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} u \\ v \end{pmatrix} = \frac{1}{x^2+y^2} \begin{pmatrix} x \\ -y \end{pmatrix} \ . \tag{34}$$

The conformal factor $\Psi(x, y)$ of this transformation is easily calculated to be

$$\Psi(x, y) = \sqrt{2}\frac{(y^2 - x^2)}{(y^2 + x^2)^2} \ . \tag{35}$$

Once again we consider initially a regular Cartesian grid, in this case occupying all four quadrants, as shown in figure 5. We leave out the line segments crossing the coordinate origin to avoid the singular point $(x, y) = (0, 0)$ in the inversion.

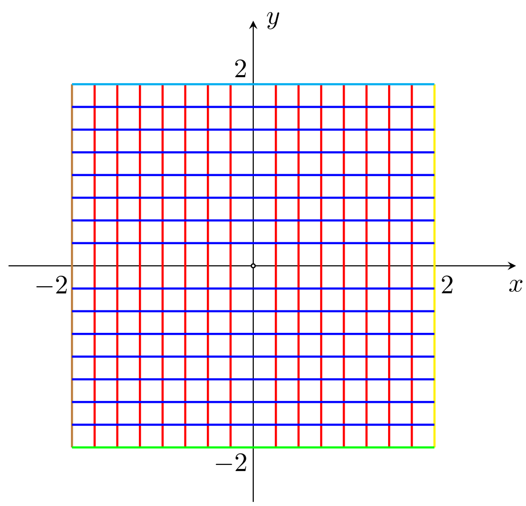

Figure 5: Initial Cartesian grid, occupying all four quadrants

The inversion operation applied on this Cartesian grid produces the picture 6. Again, all local angles are preserved, the resulting red and blue lines meet always at a right angle. The initially outer line segments, with the brown, yellow, green and cyan colors, are mapped to the inner part of the resulting conformal diagram. The initially innermost line segments are mapped to the outer part of the conformal diagram.

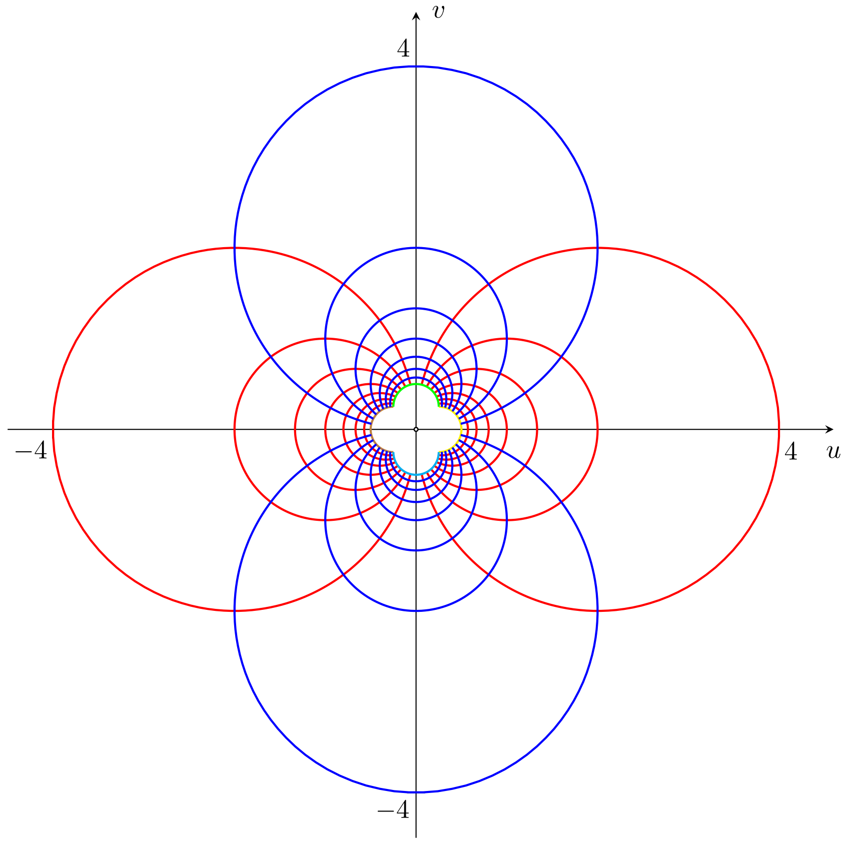

Figure 6: Conformally transformed grid under $z \mapsto z^{-1}$

PGF/Ti*k*Z Code. In the following we display and discuss the PGF/Ti*k*Z code leading to the output of the first example $z \mapsto z^2$. The code used reads

```
\begin{tikzpicture} [scale=0.6] % Overall scale of graphics
%-------------------------------------
% Coordinate system
\draw [>=stealth,->] (-6.0,0)--(6.2,0); % u-axis
\draw [>=stealth,->] (0,-9.4)--(0,9.6); % v-axis
\node at (6.18,-0.45) {$u$}; % u-axis
\node at (0.44,9.6) {$v$}; % v-axis
%-------------------------------------
% Marks and labels (scaled according to graphics size)
\draw [line width=0.6] (-4.0,0.1)--(-4.0,-0.1); % mark
\draw [line width=0.6] (-2.0,0.1)--(-2.0,-0.1); % mark
\draw [line width=0.6] (2.0,0.1)--(2.0,-0.1); % mark
```

```
\draw [line width=0.6] (4.0,0.1)--(4.0,-0.1); % mark
\draw [line width=0.6] (-0.1,-8.0)--(0.1,-8.0); % mark
\draw [line width=0.6] (-0.1,-6.0)--(0.1,-6.0); % mark
\draw [line width=0.6] (-0.1,-4.0)--(0.1,-4.0); % mark
\draw [line width=0.6] (-0.1,-2.0)--(0.1,-2.0); % mark
\draw [line width=0.6] (-0.1,2.0)--(0.1,2.0); % mark
\draw [line width=0.6] (-0.1,4.0)--(0.1,4.0); % mark
\draw [line width=0.6] (-0.1,6.0)--(0.1,6.0); % mark
\draw [line width=0.6] (-0.1,8.0)--(0.1,8.0); % mark
\node at (-4.45,-0.44) {$-4$}; % label
\node at (4.25,-0.42) {$4$}; % label
\node at (-0.58,-8.025) {$-8$}; % label
\node at (-0.485,-4.0) {$-4$}; % label
\node at (-0.255,4.0) {$4$}; % label
\node at (-0.35,8.0) {$8$}; % label
%-------------------------------------
% Plots using the parametric equations
%-------------------------------------
% The x = const curves
% the function variable \x takes values for the y-direction
% and varies from \x=-2.1 to \x=+2.1
%-------------------------------------
% x = 0.0
\draw [ domain= -2.1:2.1, samples=200, color=brown, line width=0.8 ]
plot ({ 0.0 - ( \x )^2 } , { 2.0 * 0.0 * ( \x ) });
% x = 0.25
\draw [ domain= -2.1:2.1, samples=200, color=red, line width=0.8 ]
plot ({ 0.0625 - ( \x )^2 } , { 2.0 * 0.25 * ( \x ) });
% x = 0.5
\draw [ domain= -2.1:2.1, samples=200, color=red, line width=0.8 ]
plot ({ 0.25 - ( \x )^2 } , { 2.0 * 0.5 * ( \x ) });
% x = 0.75
\draw [ domain= -2.1:2.1, samples=200, color=red, line width=0.8 ]
plot ({ 0.5625 - ( \x )^2 } , { 2.0 * 0.75 * ( \x ) });
% x = 1.0
\draw [ domain= -2.1:2.1, samples=200, color=red, line width=0.8 ]
plot ({ 1.0 - ( \x )^2 } , { 2.0 * 1.0 * ( \x ) });
% x = 1.25
\draw [ domain= -2.1:2.1, samples=200, color=red, line width=0.8 ]
plot ({ 1.5625 - ( \x )^2 } , { 2.0 * 1.25 * ( \x ) });
% x = 1.5
```

```
\draw [ domain= -2.1:2.1, samples=200, color=red, line width=0.8 ]
plot ({ 2.25 - ( \x )^2 } , { 2.0 * 1.5 * ( \x ) });
% x = 1.75
\draw [ domain= -2.1:2.1, samples=200, color=red, line width=0.8 ]
plot ({ 3.0625 - ( \x )^2 } , { 2.0 * 1.75 * ( \x ) });
% x = 2.0
\draw [ domain= -2.1:2.1, samples=200, color=yellow, line width=0.8 ]
plot ({ 4.0 - ( \x )^2 } , { 2.0 * 2.0 * ( \x ) });
%------------------------------------
% The y = const curves
% the function variable \x takes values for the x-direction
% and varies from \x=0.0 to \x=+2.1
%------------------------------------
% y = 2.0
\draw [ domain= 0.0:2.1, samples=200, color=cyan, line width=0.8 ]
plot ({ ( \x )^2 - 4.0 } , { 2.0 * ( \x ) * 2.0 });
% y = 1.75
\draw [ domain= 0.0:2.1, samples=200, color=blue, line width=0.8 ]
plot ({ ( \x )^2 - 3.0625 } , { 2.0 * ( \x ) * 1.75 });
% y = 1.5
\draw [ domain= 0.0:2.1, samples=200, color=blue, line width=0.8 ]
plot ({ ( \x )^2 - 2.25 } , { 2.0 * ( \x ) * 1.5 });
% y = 1.25
\draw [ domain= 0.0:2.1, samples=200, color=blue, line width=0.8 ]
plot ({ ( \x )^2 - 1.5625 } , { 2.0 * ( \x ) * 1.25 });
% y = 1.0
\draw [ domain= 0.0:2.1, samples=200, color=blue, line width=0.8 ]
plot ({ ( \x )^2 - 1.0 } , { 2.0 * ( \x ) * 1.0 });
% y = 0.75
\draw [ domain= 0.0:2.1, samples=200, color=blue, line width=0.8 ]
plot ({ ( \x )^2 - 0.5625 } , { 2.0 * ( \x ) * 0.75 });
% y = 0.5
\draw [ domain= 0.0:2.1, samples=200, color=blue, line width=0.8 ]
plot ({ ( \x )^2 - 0.25 } , { 2.0 * ( \x ) * 0.5 });
% y = 0.25
\draw [ domain= 0.0:2.1, samples=200, color=blue, line width=0.8 ]
plot ({ ( \x )^2 - 0.0625 } , { 2.0 * ( \x ) * 0.25 });
% y = 0.0
\draw [ domain= 0.0:2.1, samples=200, color=blue, line width=0.8 ]
plot ({ ( \x )^2 - 0.0 } , { 2.0 * ( \x ) * 0.0 });
% y = -0.25
```

```
\draw [ domain= 0.0:2.1, samples=200, color=blue, line width=0.8 ]
plot ({ ( \x )^2 - 0.0625 } , { - 2.0 * ( \x ) * 0.25 });
% y = -0.5
\draw [ domain= 0.0:2.1, samples=200, color=blue, line width=0.8 ]
plot ({ ( \x )^2 - 0.25 } , { - 2.0 * ( \x ) * 0.5 });
% y = -0.75
\draw [ domain= 0.0:2.1, samples=200, color=blue, line width=0.8 ]
plot ({ ( \x )^2 - 0.5625 } , { - 2.0 * ( \x ) * 0.75 });
% y = -1.0
\draw [ domain= 0.0:2.1, samples=200, color=blue, line width=0.8 ]
plot ({ ( \x )^2 - 1.0 } , { - 2.0 * ( \x ) * 1.0 });
% y = -1.25
\draw [ domain= 0.0:2.1, samples=200, color=blue, line width=0.8 ]
plot ({ ( \x )^2 - 1.5625 } , { - 2.0 * ( \x ) * 1.25 });
% y = -1.5
\draw [ domain= 0.0:2.1, samples=200, color=blue, line width=0.8 ]
plot ({ ( \x )^2 - 2.25 } , { - 2.0 * ( \x ) * 1.5 });
% y = -1.75
\draw [ domain= 0.0:2.1, samples=200, color=blue, line width=0.8 ]
plot ({ ( \x )^2 - 3.0625 } , { - 2.0 * ( \x ) * 1.75 });
% y = -2.0
\draw [ domain= 0.0:2.1, samples=200, color=green, line width=0.8 ]
plot ({ ( \x )^2 - 4.0 } , { - 2.0 * ( \x ) * 2.0 });
%------------------------------------
% Origin is left out (scaled according to graphics size)
\fill (0,0) circle (2.4pt);
\fill [color = white] (0,0) circle (1.2pt);
%------------------------------------
\end{tikzpicture}
```

We describe briefly the primary commands in the above code. The PGF/Ti*k*Z code is initialized and completed by the command pair

```
\begin{tikzpicture}
...
\end{tikzpicture}
```

Comment lines or notes start with the % symbol. The definition and labeling of the coordinate system above is obvious. The plotting of the parametric curves defined by $x = c$ is achieved by a sequence of commands of the general form

```
% x = (VALUE)
\draw [ domain= -2.1:2.1, samples=200, color=red, line width=0.8 ]
plot ({ (VALUE)^2 - ( \x )^2 } , { 2.0 * (VALUE) * ( \x ) });
```

These commands encode the set of parametric curves

$$\left.\begin{aligned} u &= c^2 - y^2 \\ v &= 2cy \end{aligned}\right\} . \tag{36}$$

The variable \x takes 200 discrete equidistant values in the domain between the points -2.1 and 2.1 along the y-axis. One can use a higher number of samples to produce a smoother curve. The command is carried out separetely for each curve defined by the discrete `VALUE` of x

$$x \in \{0.0,\, 0.25,\, 0.5,\, 0.75,\, 1.0,\, 1.25,\, 1.5,\, 1.75,\, 2.0\} . \tag{37}$$

The plotting of the parametric curves defined by $y = d$ is analogous. The commands

```
% y = (VALUE)
\draw [ domain= 0.0:2.1, samples=200, color=blue, line width=0.8 ]
plot ({ ( \x )^2 - (VALUE)^2 } , { 2.0 * ( \x ) * (VALUE) });
```

correspond to the set of parametric curves

$$\left.\begin{aligned} u &= x^2 - d^2 \\ v &= 2xd \end{aligned}\right\} . \tag{38}$$

The variable \x takes 200 discrete equidistant values in the domain between the points 0.0 and 2.1 along the x-axis. The command is again carried out separetely for each curve defined by the discrete `VALUE` of y

$$\begin{aligned} y \in \{&-2.0, -1.75, -1.5, -1.25, -1.0, -0.75, -0.5, -0.25, \\ &0.0,\, 0.25,\, 0.5,\, 0.75,\, 1.0,\, 1.25,\, 1.5,\, 1.75,\, 2.0\} . \end{aligned} \tag{39}$$

The other examples of conformal transformations visualized in this article can be treated in an analogous fashion. The language PGF/Ti*k*Z provides not only polynomial and rational functions but also transcendental functions. Hence it is possible to generate graphics for a large variety of conformal transformations. For a description of the PGF/Ti*k*Z language the thorough manual [3] should be consulted.

Conclusion. In the first part of this article we provided the basic facts of the theory of conformal transformations and the conformal group acting on Euclidean space. In the second part we showed how it is possible, by using elementary means, to generate high-quality visualizations of two-dimensional conformal transformations and the corresponding geometries. The practical implementation is based on the open source graphics language PGF/Ti*k*Z and, despite its manual character, proves to be effective for the purpose.

References

[1] Ralph Blumenhagen, Erik Plauschinn: *Introduction to Conformal Field Theory*. Springer, 2009

[2] Philippe Di Francesco, Pierre Mathieu, David Senechal: *Conformal Field Theory*. Springer, 1997

[3] Till Tantau: *The TikZ and PGF Packages*. Universität zu Lübeck, 2015

[4] Michael T. Vaughn: *Introduction to Mathematical Physics*. Wiley-VCH, 2007