From Logo to MetaPost

Mateusz Kmiecik

Abstract

The Logo language (turtle graphics) is recommended for teaching of Information Technology at secondary schools in Poland. It is quite a primitive language so young people quickly get discouraged while programming more advanced pictures. Could MetaPost be used to this end?

1 Why have I chosen this topic?

When I was preparing for the Malopolska Computing Competition, I had to learn about many domains concerning computers. One of them was the Logo language. Before the competition I had noticed one of the tools which my dad [Jacek Kmiecik] had been using to create pictures. It was MetaPost. Dad suggested that I should compare and describe MetaPost and Logo in a BachoTEX presentation. But I had one problem with the topic of this presentation. After consideration I have chosen 'From Logo to MetaPost' because I created the same pictures in MetaPost as in Logo. Both languages are used in making graphics but I want to show that MetaPost is superior.

2 Logo

2.1 Introduction

Logo is a programming language which supports the development of creative and logical thinking as well as algorithmic abilities. In Poland, since the second half of the 1980s Logo has been used as a teaching tool in mathematics and computing lessons [1].

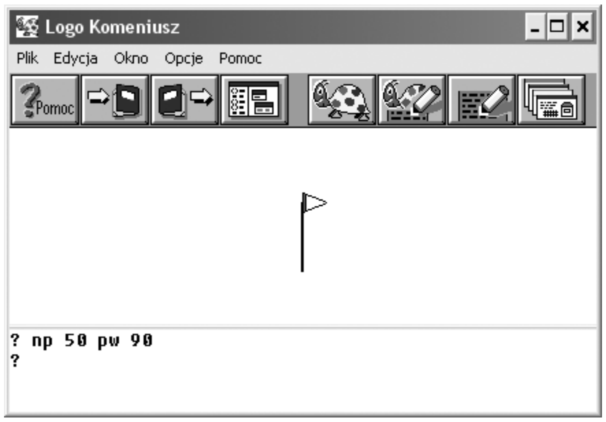
The graphic symbol of Logo is a turtle. Logo uses a specific geometry — 'turtle geometry'. Thanks to the symbol of a turtle which we can move and turn on the screen and which leaves signs, we can create pictures.

The program called Logo Komeniusz is a didactic tool which helps with using the Logo language. It has Polish commands which are very useful to help students comprehend the basic commands necessary for drawing.

Of course there are other programs to help us understand the Logo language but in my opinion Logo Komeniusz is the best one. This program is a commercial one, but its price is adjusted to the group of students who need it.

2.2 Beginnings

At first I was making easy pictures, such as a triangle or a square. Then I made my first procedure — a command creating a chosen polygon. After I absorbed the basics of Logo, I started creating more complicated figures, even whole pictures.



```
? np 50 pw 90
?
```

Figure 1: The window of Logo Komeniusz program

2.3 An example of making pictures

Commands are inserted in the bottom part of the Logo Komeniusz window. When you put commands and press `Enter` the results will be shown in the top part (fig. 1).

These commands will make the turtle go 50 steps ahead and turn around 90 degrees:

```
np 50
pw 90
```

`np` is short for `naprzód` which in English means `towards`, and `pw` is short for `prawo` which in English means `right`.

If we want to jump to another place we have to use commands `podnieś` (`lift` in English) and move the turtle to the proper place. To start drawing a picture again we have to use command `opuść` (`drop`).

```
pod   ==> podnieś
np 20 ==> naprzód 20
opu   ==> opuść
```

It is worth mentioning that Logo Komeniusz uses a Polish dialect which I believe makes using it much easier — but there are many contradictory opinions.

3 MetaPost

3.1 Introduction

When I was making pictures in Logo, I had lots of problems with it. My dad said that after the competition I should draw the same figures in MetaPost and then I would see the difference. I didn't think that it would be easier programming with paths (example: fig. 3). And there I noticed the main difference between Logo and MetaPost: Logo works by controlling an object, while MetaPost works on defining a path (opened or closed), which we can fill with color or not and put it at the proper place on the virtual "paper".

3.2 Beginnings

Everybody who starts to learn MetaPost with the original book begins with a picture of a triangle with a line (fig. 2). Not wanting to break this rule I began my MetaPost adventure with it too.

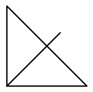
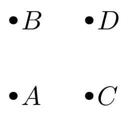


Figure 2: My first MetaPost picture (based on [3], page 4)

I was studying MetaPost by taking to pieces examples which I found on the Internet [5, 6, 7]. My dad took the role of teacher and explained many useful functions to me. I have to admit that I had quite large problems getting out of the habit of "using the turtle" (the method of working in Logo).

3.3 An example of making pictures

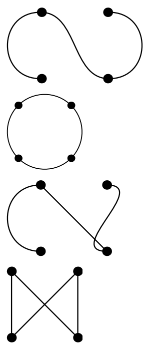
To make a picture with MetaPost, at first you have to define coordinates (x, y) of each point (*pair*) — in the example there will be four points (fig. 3):



```
pair A, B, C, D;
A = (0,0);    B = (0,10mm);
C = (10mm,0); D = (10mm,10mm);
```

Figure 3: Four points and the code to produce them

Next, I show four examples — how we can define paths between those four points and make four different shapes: fig. 4.



```
draw A..B..C..D;

draw A..B..D..C..cycle;

draw A{left}..{right}B
   --C{left}..{left}D;

draw A--B--C--D--cycle;
```

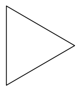
Figure 4: Examples of paths through the four points of fig. 3

To edit MetaPost files we only need a good text editor. To then create graphics we have to run the command `mpost example.mp` where `example` is the name of our input file. Then we will get the output `example.1` where the `1` refers to the number of a particular figure defined in the MetaPost file. To see the image we can put it into a LaTeX file or use the command `mptopdf` which will convert a file `example.1` to PDF.

4 Examples

In this section I'm going to show the examples which I did both in MetaPost and in Logo.

Triangle (fig. 5)



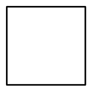
Logo version:

```
powtórz 3 [ npw 20 pw 120 ]
```

MetaPost version:

```
pair A,B,C;
A = (1cm,0);
B = A rotated 120;
C = B rotated 120;
draw A--B--C--cycle;
```

Square (fig. 6)



Logo version:

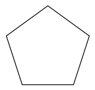
```
powtórz 4 [ np 20 pw 90 ]
```

The `powtórz` (`repeat` in English) command makes a loop; the next number defines how many times this loop will be repeated.

MetaPost version:

```
draw (0,0)--(0,u)
   --(u,u)--(u,0)--cycle;
```

Polygon (fig. 7)



Logo version:

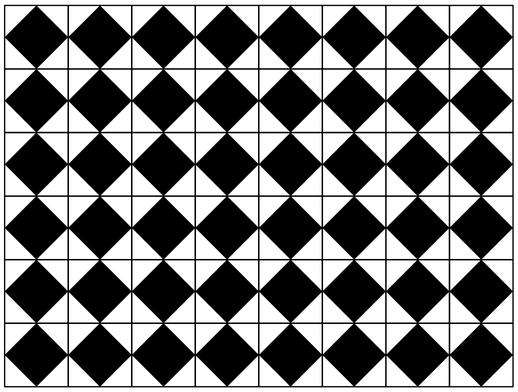
```
oto wielokąt :n :bok
powtórz :n [ np :bok pw 360/:n ]
już
```

The command `oto` defines a new procedure, here with parameters `:n` and `:bok`, then the procedure definition, ending with `już`.

MetaPost version:

```
def wielokat (expr n,b) =
 draw for i := 0 step 1 until n-1:
        1cm*up rotated (i*360/n) --
 endfor cycle;
enddef;
wielokat(5,1cm)
```

Double-squared chessboard (fig. 8)



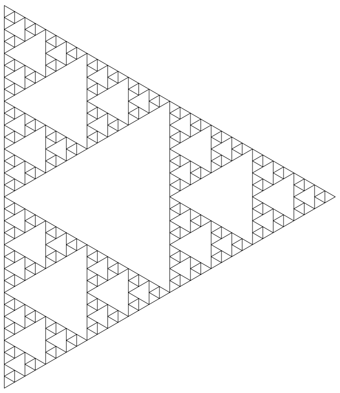
Logo version:

```
oto kw
 cs pż
 niech "a~25
 pod np 5 * :a lw 90 np 7 * :a pw 90 opu
 powtórz 6 [powtórz 8 [mkw :a pod pw 90
 np 2 * :a opu lw 90] pod
 np -2 * :a pw 90 np - 16 * :a lw 90]
 pod wróć
już

oto mkw :bok
 ugp 1 ukm 0
 pod np - :bok pw 90 np - :bok lw 90 opu
 powtórz 4 [np 2 * :bok pw 90]
 np :bok pw 45
 powtórz 4 [np :bok * pwk 2 pw 90]
 pw 45 pod np :bok lw 90
 zamaluj
już
```

MetaPost version:

```
numeric u; u := 25;
pair A,B,C,D;
A=(0,0); B=(u,0); C=(u,u); D=(0,u);

transform T;
A transformed T = 1/2[A,B];
B transformed T = 1/2[B,C];
C transformed T = 1/2[C,D];

path p; p := A--B--C--D--cycle;
path r; r := p transformed T;

picture o;
o := image (
        draw p;
        fill r withcolor black;
);
for i := 1 upto 8:
for j := 1 upto 6:
        draw o shifted (u*(i,j));
endfor;
endfor;
```

Sierpiński's Triangle (fig. 9)



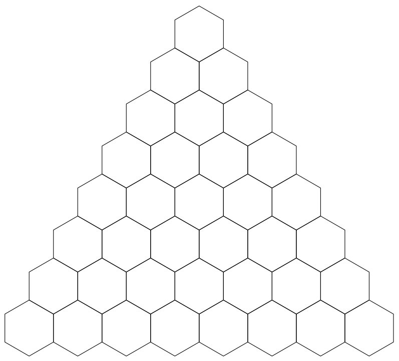
Logo version:

```
oto sierpinski :a :n
 sż opu
 jeśli :n = 0 [stop]
 trojkat :a
 sierpinski ( :a / 2 ) ( :n - 1 )
 np ( :a / 2 )
 sierpinski ( :a / 2 ) ( :n - 1 )
 np ( :a / 2 ) pw 120 np ( :a / 2 )
 sierpinski ( :a / 2 ) ( :n - 1 )
 np ( :a / 2 ) pw 120 np :a pw 120
już
```

MetaPost version:

```
def sierpinskiN (expr a, b, n) =
 if n = 0:
 draw a--(b rotatedabout(a,60))--b--cycle;
 else:
  sierpinskiN(a, 0.5[a,b], n-1);
  sierpinskiN(0.5[a,b], b, n-1);
  sierpinskiN(0.5[a,b rotatedabout(a,60)],
       0.5[a~rotatedabout(b,-60),b], n-1);
 fi;
enddef;
z0 = origin;
z1 = z0 shifted (0,-500);
sierpinskiN (z0, z1, 5);
```

Honeycomb (fig. 10)



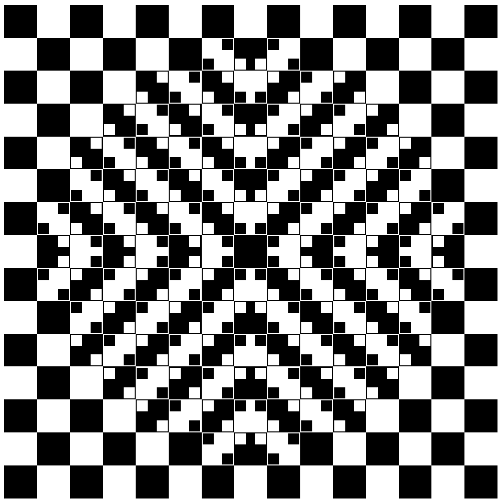
Logo version:

```
oto plaster :n
 jeśli :n < 1 [stop]
 cs sż
 niech "bok 300 / 8
 pod np ( - ( 0.75 * :n - 1 ) * :bok )
 pw 90 np ( - 0.5 * ( :n - 1 ) * :bok
 * pwk 3 ) lw 90 opu
 plastek :n :bok
już

oto plastek :n :bok
 jeśli :n = 0 [stop]
 powtórz :n [szesc :bok pod pw 90
 np :bok * ( pwk 3 ) lw 90 opu]
 pod np 1.5 * :bok pw 90
 np ( - ( :n - 0.5 ) * :bok * ( pwk 3 ) )
 lw 90 opu
 plastek :n - 1 :bok
już
```

MetaPost version:

```
numeric n,w; n := 6; w := 25;
path _s; _s :=  for i := 0 upto n-1:
  up rotated (i*360/n) -- endfor
  cycle;

def plast (expr n, b) =
 if n<>0:
  for i := 1 upto n:
   pair zz;
   xpart zz = ( (i-1)*(sqrt3) -
           ((sqrt3)/2) * n ) * b;
   ypart zz = -3/2*n * b;
   draw _s scaled b shifted zz;
  endfor;
 plast ((n-1), b);
 fi;
enddef;
```

Magic chessboard (fig. 11)



Mateusz Kmiecik

Logo version:

```
oto iluzja
 sż
 cs
 niech "bok 50
 niech "mbok 10
 skok 7 * :bok ( - 7 * :bok )
 powtórz 7 [powtórz 7           >>
  [kwadratcz skok 0 (2*:bok)]]
 kwadratcz skok - :bok (-:bok)
 powtórz 7 [kwadratcz skok 0 (-:bok*2)]
 skok - :bok :bok]
 powtórz 7 [kwadratcz skok 0 (2*:bok)]
 kwadratcz
 skok 7 * :bok ( - 7 * :bok )
 ; male
 skok :bok / 2 ( - :bok / 2 )
 niech "bok 10
 powtórz 4                      >>
  [kwadraciki pw 90 skok 25 ( - 25 )]
już
```

This code is wrong because of a restriction of the Logo Komeniusz editor. I used the symbol `>>` because the command `powtórz` and its syntax must be written in the same line. And this is a drawback of making procedures in Logo — regardless of the resulting code's legibility, some commands must be written all on one line.

The entire program takes 70 lines, and is on my web page: `www.mateusz.kmiecik.pl`

MetaPost version:

```
numeric u, k; u := 25; k := 0.3;
path fullsquare;
fullsquare :=
  unitsquare shifted -center unitsquare;

picture A, B, C, D, E, F, G, H;
path p, q;
p := fullsquare scaled u;
q := p scaled 0.33;

color Wh, Bl;
def Wh = white enddef;
def Bl = black enddef;

A := image (fill p withcolor black;);
B := image (fill p withcolor white;);

C := image (draw A;
fill q shifted (k*u*(-1,1)) withcolor Wh;
fill q shifted (k*u*(1,-1)) withcolor Wh;
);
D := image (draw B;
fill q shifted (k*u*(-1,1)) withcolor Bl;
fill q shifted (k*u*(1,-1)) withcolor Bl;
);
E := image (draw A;
fill q shifted (k*u*(1,1))  withcolor Wh;
fill q shifted (k*u*(1,-1)) withcolor Wh;
);
F := image (draw B;
fill q shifted (k*u*(1,1))  withcolor Bl;
fill q shifted (k*u*(1,-1)) withcolor Bl;
);

G := image (draw E
       rotatedaround (center E,180););
H := image (draw F
       rotatedaround (center F,180););

for i := 1 upto 7:
for j := 1 upto 8:
 draw
 if ( (j=1) and (i<>7) ):
   if odd(i): H else: G fi
 elseif ((i+j) = 8) and (i <>7):  D
 elseif ((i+j) = 6):              D
 elseif ((i+j) = 4):              D
 elseif ((i+j) = 9) and (i>2) and (i<6):
                                  C
 elseif ((i+j) = 7):              C
 elseif ((i+j) = 5):              C
 elseif ((i+j) = 3):              C
 else:
  if odd(j):
   if odd(i):    B
   else:         A
   fi
  else:
    if odd(i-1): B
    else:        A
    fi
  fi
 fi
 shifted(u*(i,j));
endfor;
endfor;

picture szach;
szach := currentpicture;
currentpicture:=nullpicture;

draw szach shifted (0,-u);
draw A;
draw currentpicture rotated 90;
draw currentpicture rotated 180;
```

5 Summary

Although Logo Komeniusz uses Polish commands, drawing some figures is still very troublesome. In MetaPost we make images by defining proper paths and points, which is very useful. Another difference is the approach to drawing figures: MetaPost doesn't control a pencil (or turtle) or anything like that, but defines proper paths and puts them on a virtual sheet of paper, filled with some color. The drawback of MetaPost is that we must have installed TEX and we must know how to operate it. Logo Komeniusz is a program which doesn't require any installation. After turning on it we can start working.

I think that for learning algorithms in secondary schools Logo Komeniusz is very valuable, while MetaPost can be introduced for additional lessons, perhaps in computing or mathematics.

References

[1] Jadwiga Orłowska-Puzio: *Dydaktyczne zastosowania programowania w LOGO*, `http://www.jorlowska.prv.pl/rdydakt/logo.html`, wersja PDF: `http://www.jorlowska.prv.pl/rdydakt/docs/DYDAKTYCZNE_ZASTOSOWANIA_PROGRAMOWANIA_W_LOGO.pdf`

[2] *Pierwszy dokument w MetaPost* [First document with MetaPost], `http://www.gust.org.pl/doc/1stdocument/metapost`

[3] John D. Hobby: *A user's manual for MetaPost*, `http://www.tug.org/docs/metapost/mpman.pdf`

[4] Hans Hagen: *Metafun*, `http://www.pragma-ade.com/general/manuals/metafun-p.pdf`

[5] *Métapost: exemples*, `http://tex.loria.fr/prod-graph/zoonekynd/metapost/metapost.html`

[6] André Heck: *Learning MetaPost by Doing*, `http://staff.science.uva.nl/~heck/Courses/mptut.pdf`

[7] Documentation, tutorials and examples at `http://www.tug.org/metapost.html`

◇ Mateusz Kmiecik
Gimnazjum nr 16 im. Króla Stefana Batorego, Kraków
mateusz (at) kmiecik dot pl
http://www.mateusz.kmiecik.pl