Create Your Own Simulations!

MicroWorlds Pro is a multimedia programming environment that offers you the possibility to create dynamic, interactive school and Internet projects. It lets you become active web designers, not just passive web viewers. You can use MicroWorlds Pro to enhance your understanding of MicroWorlds and to get a real sense of the depth and breadth of this powerful multimedia programming environment (MicroWorlds Pro, 2000).

Among other things, MicroWorlds Pro allows you to import a picture, insert text, stamp text, create animations, create simulations, and create interactive buttons. This project will mostly focus on creating interactive computer simulations, which have proven an ideal tool for teaching and learning, especially in science education. But enough with the theory, let’s find out how people create simulations by using a multimedia environment such as MicroWorlds Pro. Follow us through the process of creating your own animations and find out for yourself the answer to this question or others that have been troubling you!

Degree of Difficulty

Experimental: Moderate
Conceptual: Moderate

Prerequisite Knowledge

This project is suggested for students that are already familiar with basic computer skills (e.g., turn on the computer, use the keyboard, use the mouse), have some experience in basic computer functions (e.g., access programs, open programs, save data), and have knowledge of basic computer programs (e.g., word processing programs, spreadsheets etc.).

Note: For programming in MicroWorlds Pro, Logo knowledge is needed. Logo is a language for computers. It has a very small number of words and grammatical rules. Where a spoken language has sentences, Logo has instructions. Where a spoken language’s rules for how to put words into sentences can be complicated with many exceptions, Logo’s rules for building instructions are much simpler but more stringent (MicroWorlds Pro, 2000). Thus, it is of particular importance to go through the MicroWorlds Pro topics (click on Help and select MicroWorlds Pro topics – see figure 1) before starting this project, where, among other things, Logo is introduced.
Particularly, read the following topics:

A. **MicroWorlds Pro Fundamentals**
   - An Overview of MicroWorlds Pro
   - Turtles
   - Graphics
   - Text Boxes
   - More About MicroWorlds Objects
   - Programming Environment

B. **Logo Programming**
   - Logo Fundamentals
   - Programming With Turtle Geometry
   - Programming With Texts
   - Programming With Numbers, Words, and Lists

C. **Interesting Concepts and Techniques**
   - Process Management
   - Variables
   - Object Manipulation Under Program Control
   - Programming Techniques
   - Sharing Projects

For the purposes of this project, we will consider all this information to be prior knowledge, including the Logo and MicroWorlds Pro vocabulary, which you can find under the Help menu. As mentioned before, the Logo and MicroWorlds Pro vocabulary consist of commands that you use to program the functions of each turtle you use.
In case you want to check whether your program’s vocabulary or syntax has any mistakes, press the F4 key and MicroWorlds will find common syntax errors. The error message will be displayed, at the same time, in the Status Bar. The errors are highlighted in sequence each time you press F4. If you press F4 when the Procedures Tab area is not displayed, it will be displayed automatically (MicroWorlds Pro, 2000).

**Objectives**

Completion of the activities should enable you:

- to learn how to create animations by using MicroWorlds Pro.
- to use MicroWorlds Pro as a tool to create simulations.
- to post your animations or simulations on the World Wide Web.

**Materials:** computer, MicroWorlds Pro (visit http://www.microworlds.com/solutions/mwpro.html for purchasing the software), the Learning MicroWorlds Pro and the MicroWorlds Pro Tips and Tricks books that are included in the package of MicroWorlds Pro.

**Useful Information:** The Right Click on your PC’s mouse is used to open pop-up menus on objects; the same functionality is obtained on a MAC by Control Click. The PC’s Shift key is used to add more than one shape to a turtle and to change the heading of the turtle; the same functionality is obtained on a MAC with the Command key.

Each MicroWorld Pro (MWP) project consists of objects and text presented on separate pages. For every project you start, there is a corresponding single page. A page is accompanied by Menus, Toolbar, Status Bar, Command Center, and Tab Areas (see figure 2).

**Figure 2**
It is also important to know about Logo’s punctuation and markers meaning. According to MicroWorlds Pro (2000):

<table>
<thead>
<tr>
<th>Punctuation and Markers</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot; Quotation mark</td>
<td>Indicates that the next word is to be taken literally. Note that we are only talking about one quote that is placed at the beginning of the word.</td>
</tr>
<tr>
<td>[ ] Square brackets</td>
<td>Surround a list.</td>
</tr>
<tr>
<td>: Colon</td>
<td>Sometimes referred to as dots. Indicates that the next word is the name of a variable. Refers to contents of the container.</td>
</tr>
<tr>
<td>( ) Parentheses</td>
<td>Group things in ways that Logo ordinarily would not or varies the number of inputs for a primitive.</td>
</tr>
<tr>
<td></td>
<td>Vertical bars</td>
</tr>
</tbody>
</table>

**Part A: Animations involving both movement and motion**

**Procedure**

Before creating our first simulation, let us introduce animation and, more specifically, explain how movement and motion in an animation can be performed. Let’s suppose that you want a car to move across your page. Follow the next steps:

1. Create a turtle by selecting 🐢 and click anywhere on the page. Click on the Graphics tab, click on the car shape, and then click on the turtle (or you can type setshape “car in the Command Center). What do you observe?

2. Right-click on the car and from the pop-up menu select animate. What do you observe? Now, press the Shift button on your keyboard and drag the car towards the left or right (or you can type seth 270 in the Command Center – 0 corresponds to north, 90 corresponds to East, 180 corresponds to south, 270 corresponds to west). Click on the car and describe what you observe. To stop the car, click on it (or click on the stop button on the toolbar). Try this for different angles (i.e., 30°, 45°, 56°, 23.5°) and explain your observations.

3. Right-click on the car and select Edit. In the pop-up dialog box, you will see the commands that were inserted automatically due to the animations we have already selected. Change the forward input from 5 to 10. What do you observe? Change the wait input from 1 to 5. What do you observe?
4. Repeat step 1. Create a button by selecting click on the page, and type in the dialog box **forward 5 wait 1**. Click on the **OK** button of the dialog box and then click on the button created on your page. Compare it with step 3. How can you create a button without selecting ?

5. Repeat step 1. Create a slider by selecting click on the page, and type in the dialog box

![Slider Dialog Box]

Create a turtle, right-click on it and select Edit to open its dialog box. Type **fd slider1** as the turtle’s instruction and set it to Many Times. Click on the turtle to start it. Use the slider to change the speed. Compare it with steps 3 and 4. How can you create a slider without selecting ?

6. Repeat steps 1 to 3 and add two more cars on your page (or you can just right-click on the already existing car, select copy, and then click on the page and select paste). Type **everyone [clickon]** in the Command Center and press the enter button on your keyboard. Then, type **everyone [clickoff]** in the Command Center and press the enter button on your keyboard. What do you observe? Now, type **talkto [car1 car2 car3] fd 20** in the Command Center and press the enter button on your keyboard. What do you observe? Compare it with the **everyone [clickon]** command.

7. Now, let’s consider programming a turtle to look like a dog running. In the Graphics Tab area, locate the two dog shapes. The first one is named **dog 1** and the second **dog 2**. Create a turtle, right-click on it and select edit. In the pop-up dialog box, name the turtle **dog** and in its instructions type **setsh “dog1 wait 5 setsh “dog2 wait 5** (where **setsh** is an abbreviation for **setshape**, either of the two commands works fine). Click the **OK** button of the dialog box and then click on the turtle. Use different numbers and explain your observations.

8. Combine steps 3 and 5 to program the dog to continuously change shapes and move forward.

The question that is raised at this point is how can we program a turtle to move in different directions, move towards another object or turtle, follow a certain path etc. The command we are looking for is **towards**, which turns the current turtle towards a specified turtle or object (in case we want to know the distance between the two, we have to use the command **distance**, i.e., `t1 show distance "t2`). The next step includes examples of the **towards** command:

10. Create 3 turtles and spread them out on your page. Type `t1, towards "t2`, in the Command Center. What do you observe? How about, if you type `t1, towards "t3`? Now, name your t1 turtle *dog* (you can replace the turtle with one of the dog shapes in the Graphics, if you like). Type in the Command Center **everyone [ht]** dog, st (the **everyone [ht]** command hits all turtles on the page and dog, st allows dog to reappear on the page). Write the following procedure on the Procedures Tab area:

```plaintext
to follow
dog,
towards "t2
fd distance "t2 wait 5
towards "t3
fd distance "t3 wait 5
end
```

Finally, type in the Command Center **dog, follow** and press enter. Explain your observations. Does the program run without the **wait** command? Does the program run if we use **glide** instead of **fd**? What can you do to have a turtle to run around an irregular track?

### Activities

1. Create a table where you give the corresponding explanations of the following Microworlds Pro or Logo programs both before and after you try the suggested program variations (note that the programs that are marked CC must be typed in the Command Center area, and the PT programs must be typed in the Procedures Tab area – do not include the symbols CC or PT in your programs):

<table>
<thead>
<tr>
<th>Programs</th>
<th>Program Variations</th>
</tr>
</thead>
<tbody>
<tr>
<td>CC:</td>
<td></td>
</tr>
<tr>
<td><code>forward 50</code></td>
<td>Change the forward number</td>
</tr>
<tr>
<td><code>show heading</code></td>
<td>Change the orientation of the turtle by using your mouse (see step 1)</td>
</tr>
<tr>
<td>CC:</td>
<td>Change the series of the numbers.</td>
</tr>
<tr>
<td><code>show -4 + 10</code></td>
<td>Change the signs of the numbers.</td>
</tr>
<tr>
<td>CC:</td>
<td>Change the series of the letters.</td>
</tr>
<tr>
<td><code>show &quot;science (for this you have to open a text box on your page. Use the A button from your toolbar)</code></td>
<td>Change the word.</td>
</tr>
<tr>
<td>PT:</td>
<td>Vary all numbers, one at a time. Disregard the wait command.</td>
</tr>
<tr>
<td>-----</td>
<td>------------------------------------------------------------------</td>
</tr>
<tr>
<td>to square :size repeat 4 [forward :size right 90 wait 10] end</td>
<td>Note: Both CC and PT commands must be entered for the program to run.</td>
</tr>
<tr>
<td>CC: t1, square 100</td>
<td></td>
</tr>
<tr>
<td>CC: repeat 3 [print &quot;science&quot;] (you need to open a text box)</td>
<td>Change number and word.</td>
</tr>
</tbody>
</table>
| CC: make "message [print " science] repeat 3 :message | Change number and word. Compare it with repeat 3 [print "science]
| CC: if ycor > 100 [setc "blue"] (after you press enter, position the turtle at the top of your page and press enter again) | Change the number, the symbol from > to <, the name of the color. |
|  forever [forward 1 wait 0.1] forever [ifelse ycor > 0 [setc "green] [setc "blue]] | Change the numbers (one at a time), the symbol from > to <, the name of the colors. |
| CC: pd forward 50 right 55 forward 50 | Change the numbers (one at a time). |
| CC: pd repeat 5 [forward 100 right 144] | Change right to left. |
| CC: pd repeat 360 [forward 1 right 1] | Change forward to back. Disregard pd. |
| CC: pd t1, arc 10 18 | Note: Run the programs separately. They are included together for comparison reasons. |
| PT: to arc 10 18 repeat 5 [forward 50 right 10] end | |
| Create a text box and write six words in it, each word at a separate line. | Create a text box. |
| CC: show textcount "text1 | Change the number from 3 to 1 in the show textitem 3 "text1. |
| CC: show textitem 3 "text1 | Note: Run the programs separately |
| CC: show textpick "text1 | |
| Create a text box. | Change the word. |
| CC: printdown "Lucent | Note: Both CC and PT commands must be entered for the program to run. |
| PT: to printdown :" Lucent | |
| if empty? :" Lucent [stop] print first :" Lucent printdown butfirst :" Lucent end | |
| CC: | Change the numbers (one at a time). |
to between? :7 :1 :12
output not or (:7 < :1) (:7 > :12)
end
PT:
show between? 7 1 12

Note: Both CC and PT commands must be entered for the program to run.

2. Use the **Map, Replace, Sort, Which, Intersect, Subset**, and **Union** commands in programs that show their function (Hint: See Help Menu’s Programming: Useful Tools).

3. Use the commands of activity 1 in programs that show their function. Try to combine different commands within one program.

4. Use the commands of activity 1 to expand on the program you created in procedure step 8. Try to combine different commands within one program.

5. Type the following program in the Command Center: **setshape [bird1 bird2] repeat 20 [fd 2 wait 2] repeat 20 [fd 10 wait 2] glide 100 5.**
   - Before running the program, predict what will happen.
   - Run the program and explain any discrepancies between your observations and your predictions.
   - What other command can you use to substitute **glide**? Can we use different commands and have the same result?
   - Write a program which shows the bird moving back and forth on the horizontal level of your page (Hint: read pages 49-50 from the MicroWorlds Pro Learning by Tom Lough).

6. Create an animation that shows a square changing size. Repeat the same for a rectangle, triangle, and hexagon. (For the purposes of this activity the figures must be created by you. To create a new shape, double click on one of the empty shapes in the shapes palette, and then use the paint tools to design the shapes.)

7. Repeat activity 2 but this time use a different color each time there is a change in size.

8. Repeat step 9, by using the **dolist** command (use the information given in MicroWorlds Pro Tips and Tricks on page 35). Use this type of program to show a bee moving on the perimeter of different path shapes, such as squares, triangles, hexagons, octagons etc. Set different times of waiting at the tips of the path shapes.

9. Use stick figures of at least 5 consecutive frames showing simple animation of some sort (i.e., a woman running). Do the activity by using at first only one figure and then add a second one (the two figures should have some sort of interaction). To create a new shape, double click on one of the empty shapes in the shapes palette, and then use the paint tools to design the shapes (hint: use procedure steps 3 and 5).
10. Create your own movie! Use stick figures of at least 5 consecutive frames showing simple animation of some sort (i.e., a man playing with a ball). First, start a new page and draw your first figure. Second, select Duplicate Page from the Pages menu to create an identical copy of your already existing page. Third, change the graphics on your second page. Fourth, continue this process until you complete the whole set of animation frames you had in mind to create for your movie. In case you want to include sound in your project, you can either create your own melody (select and follow the directions given in the help menu under the topic MicroWorlds Objects: Melodies) or record your own music or sounds (select and follow the directions given in the help menu under the topic Recording Sounds). Finally, in the Procedures Tab area, type the following commands and run the program:

```lisp
  to flip
dolist [i pagelist] [getpage :i wait 5]
end
```

**Note:** The `dolist` command runs a list of instructions using a range of variable values. In this program, it creates a variable `i` and cycles through the `pagelist`, assigning each page name to `i` in turn. As `i` assumes that value, `dolist` runs the list of instructions `[getpage :i wait 5]`. This displays each page in sequence (Lough, 2000).

**Part B: Interactive Simulations**

**Procedure**

After completing the introductory activities of Part A, we are now ready to proceed into creating interactive simulations. But, what is an interactive simulation? It is a virtual representation of a phenomenon, which can be studied through a series of observations as we change the variables that control its behavior. An example of such a simulation would be an incline that has a piece of metal sliding towards the ground and you have the possibility of changing its angle, the mass of the metal, the friction coefficient between the two objects, etc.

This part is of particular importance, because it not only challenges your programming skills in MicroWorld Pro, but also challenges your knowledge in the subject you are simulating. For the purposes of this project, the subject we will emphasize is physics (this does not mean that you can not create simulations for other subjects).

Let’s say we want to create a simulation that shows relative velocity in one dimension. First, we have to select the content and concepts we want to introduce through the simulation, and then decide the best way to simulate them. In the case of relative velocities, we need to know the following:

- Velocity is a vector and is defined as the rate of change of displacement. It is not an absolute quantity but is measured relative to other objects. For example, when
you hear a car is traveling 90 km/h, you normally assume that it means a velocity relative to the road.

- To measure any object’s velocity we must specify the coordinate system or reference frame in which the measurement is to be made. Usually the origin of the coordinate system is fixed in some other body. In the example with the car, the road was the other body. But, what about driving down the highway when another car passes you with a slightly higher velocity? Although both cars run rapidly down the road, the faster car appears to overtake you very slowly. This is because when two objects are traveling along the same line, the relative velocity is obtained simply by ordinary subtraction (assuming the speeds are small compared to the speed of light):

\[ V_{AC} = V_{AB} - V_{BC} \]

Having in mind the aforementioned content and concepts, we are now ready to select the main features that our simulation must include:

- Since velocity is a vector and is defined as the rate of change of displacement, there must be a controlled movement of objects/subjects and an option for deciding which direction to follow (at least for one of the objects/subjects).
- There must be relative movement among the objects/subjects.
- There must be a specified coordinate system or reference frame in which the measurement is to be made.

Finally, we have to select the phenomenon/activity we will simulate to show all the aforementioned. This is again a very serious issue and you have to consider it very carefully, based on the following criteria:

- The phenomenon/activity must clearly show the content and concepts you want to simulate.
- The creation of the phenomenon/activity on your MicroWorlds Pro pages must be possible. It is suggested that you use already existing shapes, icons, pictures etc. (remember that you can import pictures in MicroWorlds Pro from other programs, the internet, etc.). Creating them from scratch is time consuming and sometimes impossible.
- Before starting, make sure you have a clear plan that includes a diagram of what you want to create, the program/vocabulary you will use, and the control equipment you will use (i.e., buttons, sliders, text boxes etc.)

Based on these criteria, some of the phenomena/activities that qualify for the purposes of this project are a person jogging on a running road (gym equipment), a boat moving on running water, and a person walking on a train or plane.

In this project we will create a program that shows a swan moving on running water. Both the velocities of the water and the swan will be controlled through sliders, as well as, the boat’s direction (it will be programmed to move both in the same direction the water is moving and in the opposite direction).
Now that we have specified the desired parameters of our simulation, we are ready to proceed with the creation of the simulation:

1. Create the background of your simulation by importing pictures or shapes from the Graphics area (you can also import pictures or shapes from elsewhere, i.e., internet), or by using the available painting tools of Graphics. Include a sky with some clouds (so it can be distinguished from the river), a river (use darker blue than the blue you used for the sky), and a shore (use light brown for ground). An example of such a background could be the following:

![Background Example](image)

2. Create a turtle and turn it to a swan (there is a swan shape in the Graphics area) and put it on the river, as follows

![Turtle Turned to Swan](image)
3. At this point we have a problem to solve. The way our background is designed, it is impossible for us to show the flow of the water. We cannot program the blue color of the river to flow, since it is not a turtle. Thus, we have to come up with an idea that will show the flow of the water. This can be done, for example, with a piece of wood (i.e., a log) that floats on the surface of the river and is carried along by the water (we assume that the wood will have the velocity of the stream). The piece of wood that will be added to our simulation must be a turtle, thus create one and turn it into a piece of wood. Our picture, now, changes to

4. Name the swan, debbie, and the piece of wood, wood.

5. Create two sliders. Type in the first slider dialog box, swan, and set the minimum value to zero and the maximum value to 6. Type in the second slider dialog box, river, and set the minimum value to zero and the maximum value to 4.

6. Create two buttons. Name the first one turn and the second one start (in both cases set the button to Many times).

7. At this point your page should look as follows,
8. Write the following program (only the part that is under the program’s column, not the part in explanation’s column) in the Procedures Tab area:

<table>
<thead>
<tr>
<th>Program</th>
<th>Explanation</th>
</tr>
</thead>
</table>
| to go wood, fd river 
debbie, 
if heading = 90 [ down_stream ] 
if heading = 270 [up_stream] 
end | It sets the wood to go forward in the river and defines for debbie what is considered to be downstream (towards left) and what is considered to be upstream (towards right). |
| to down_stream 
setsh “swan1” 
fd swan + river 
end | It sets debbie’s shape as it moves downstream and defines the total velocity as the sum of the individual velocities of the swan and the river. “swan1” refers to the shape of the swan that faces towards left. You can find the shape under Graphics. |
| to up_stream 
setsh “swan2” 
fd swan - river 
end | It sets debbie’s shape as it moves upstream and defines the total velocity as the difference of the individual velocities of the swan and the river. “swan2” refers to the shape of the swan that faces towards the right. You can find the shape under Graphics. |
| to turn 
debbie, ifelse shape = | It refers to the function of the turn button. Each time you press it, it turns 180 degrees |
<table>
<thead>
<tr>
<th>Activities</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Create a series of simulations where you have a car (or any other type of vehicle) accelerating, decelerating, and moving with constant velocity.</td>
</tr>
<tr>
<td>2. Combine the accelerating, decelerating, and constant velocity vehicle in one simulation.</td>
</tr>
<tr>
<td>3. Create a simulation to show the phenomenon of buoyancy. Assume that you have 3 objects of different densities (much less than that of water) at the bottom of a glass filled with water. Show how they will behave when you release them. Make the density of each object changeable by using sliders, but always less than that of the water.</td>
</tr>
<tr>
<td>4. Repeat activity 3, but this time allow the density of the objects to become bigger than that of water, as well. (Hint: make use of the if and if else statements.)</td>
</tr>
<tr>
<td>5. Create a similar simulation to the simulation of part B to show the same phenomenon. This time use a different scenario (i.e., a man walking on a ship).</td>
</tr>
<tr>
<td>6. Expand the simulation of part B by including a third object moving in the river (i.e., a boat). Follow the same procedure we followed in part B.</td>
</tr>
<tr>
<td>7. Change the ranges of the sliders and explain what will happen if we set their minimum value below zero.</td>
</tr>
<tr>
<td>8. Create a simulation where you have a car with some initial velocity sliding on the road and stopping due to friction. (Hint: Use a program similar to the one in part B.)</td>
</tr>
<tr>
<td>9. Create a simulation combining part A and part B. Use the three dolphin shapes given in the Graphics area, which means the dolphin will be changing shape as it moves.</td>
</tr>
</tbody>
</table>

```
“swan2 [setsh “swan1][setsh “swan2] rt 180 end
```

and faces in the opposite direction.

```
to startup set "button2 "on? "true end
```

It refers to the function of the start button. Each time you press it, the program starts running.

9. Run the program, by clicking the start button. Vary the velocities of the swan and the river. What do you observe? What is happening when the swan is moving opposite the flow of the river and it has the same velocity (magnitude) as the river? What do you observe when you press the turn button?
Final Project/Report

Create a series of physics simulations. It does not have to be in kinematics. You can select any physics area you like. Include a report explaining the reasoning behind your selection (see part B) and an analytical step by step description of your programs.

References
