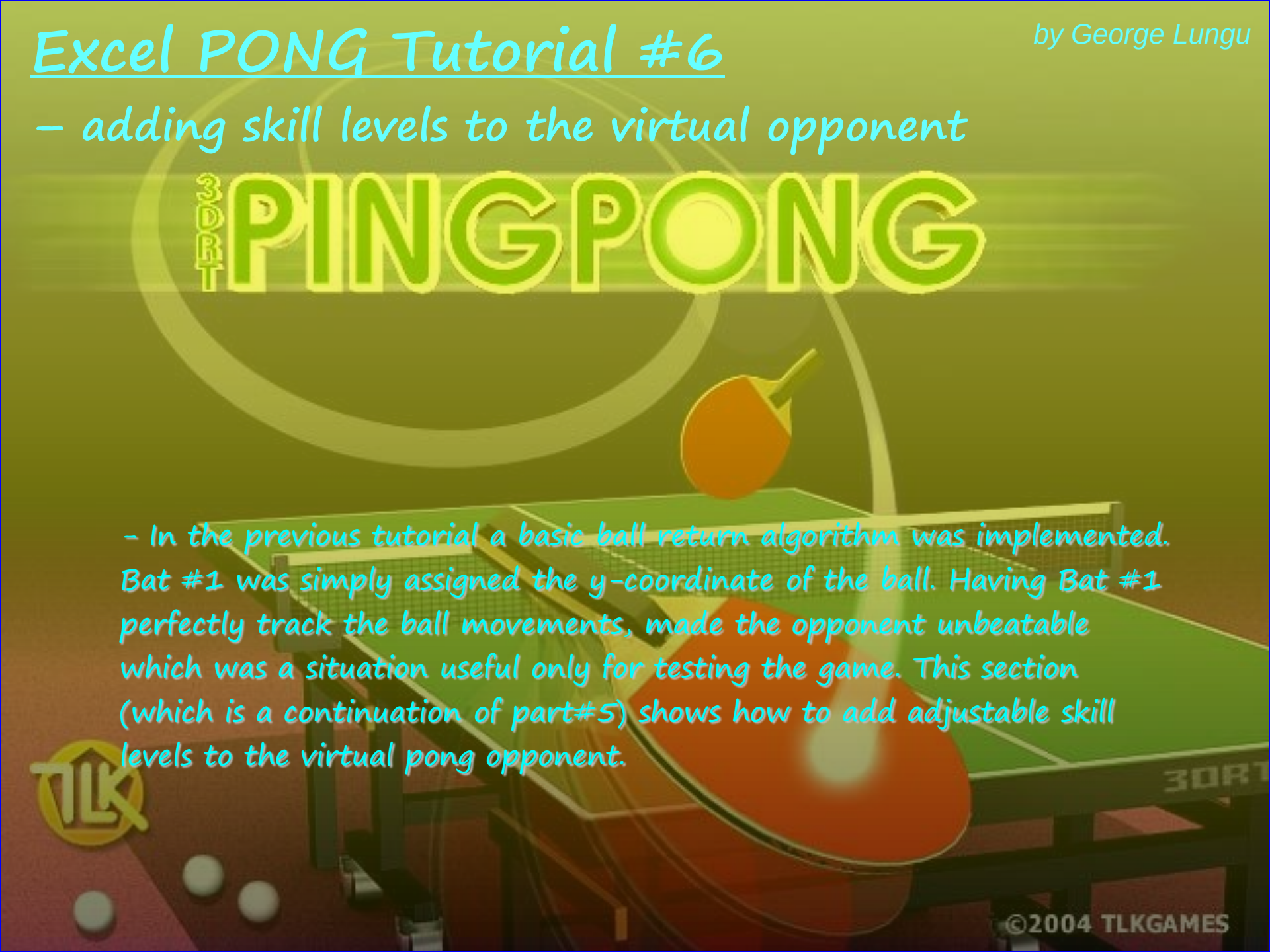
Excel PONG Tutorial #6

by George Lungu

– adding skill levels to the virtual opponent



- In the previous tutorial a basic ball return algorithm was implemented. Bat #1 was simply assigned the y-coordinate of the ball. Having Bat #1 perfectly track the ball movements, made the opponent unbeatable which was a situation useful only for testing the game. This section (which is a continuation of part#5) shows how to add adjustable skill levels to the virtual pong opponent.

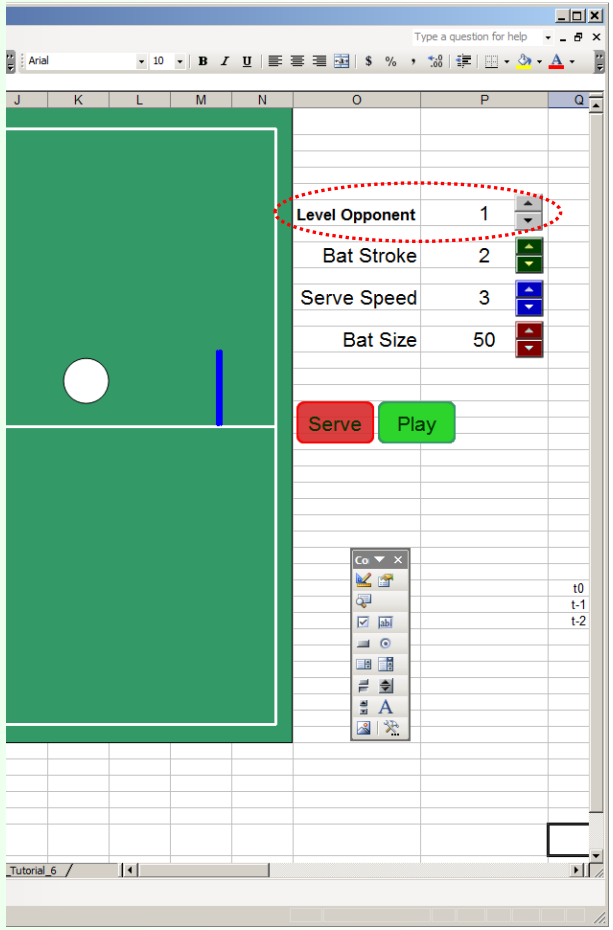
37. Create a new copy of the worksheet

- Copy the last worksheet into a new one. Rename the new worksheet “Pong_Tutorial_6”.
- For this section of the presentation keep the same macros from Module3.

38. Insert a new entry in the worksheet called Level Opponent

- As set up in the last tutorial, Bat #2 perfectly tracks the ball y-coordinate. We will build a new setup in which we will have the same bat follow the ball with an adjustable speed value, proportional to the opponent’s skill level.
- Insert the label “Level Opponent” in cell O6 with font size 14 bold.
- Create a spin button: View => Toolbars => Control Toolbox => click Design Mode in the control toolbox => drag draw a spin button => right click Properties.
- Modify properties: change name to Level_Opponent, change min: 1 and max: 20.
- Double click the button and change the new code you see there into the following:

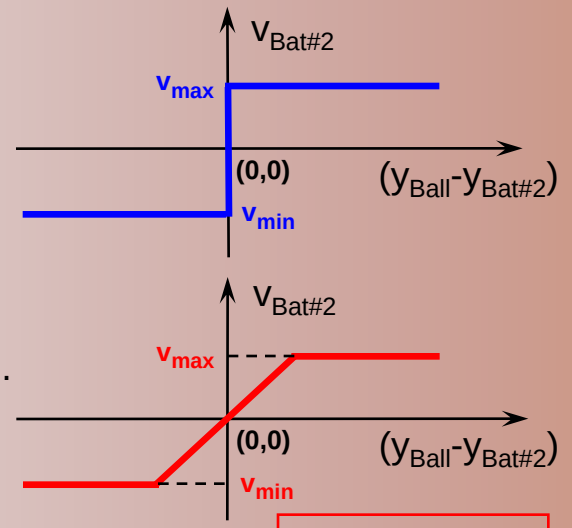
```
Private Sub Level_Opponent_Change()
Range("P6") = Level_Opponent.Value
End Sub
```



Make sure to exit the design mode before using the macro by clicking the “Design Mode” icon in the Control Toolbox !

39. Bat #1: y-speed formula and y-coordinate formula

- Instead of tracking the coordinate of the ball let's have Bat #1 move towards the ball y-coordinate with a speed proportional to the skill level of the virtual opponent (v_{min} and v_{max} are proportional to the skill level of the opponent).
- The problem with this algorithm is that Bat #2, once gotten near the ball y-coordinate will start bouncing up and down since it overshoots the ball position.
- A better algorithm will be the almost identical to the first one but around the origin makes a smooth linear passage between v_{min} and v_{max}



A simple implementation of the formula is given below (K_1 and K_2 are constants).

$$v_{Bat\#2} = sign(y_{Ball} - y_{Bat\#2})\min[K_1 \cdot abs(y_{Ball} - y_{Bat\#2}), K_2] \cdot Level_{skill}$$

$$K_1 = \frac{v_{max}}{Skill_{level}} \qquad K_2 = \frac{slope}{Skill_{level}}$$

40. Excel implementation

Bat y-speed and y-coordinate:

W27: "=SIGN(S28-V28)*MIN(0.05*ABS(S28-V28),0.5)*P6"

By experimentation, K1 and K2 were chosen to 0.05 and 0.5 respectively

V27: "=V28+W27*Y23"

$$y_{Bat\#2\_current} = y_{Bat\#2\_previous} + v_{y\_Bat\#2\_current} \cdot \Delta t$$

41. Testing the new model

-Using various values for serve speed, bat size and opponent skill level verify the functionality of the model

- Adjust K2 to your preference. A larger K2 means a faster opponent and a lower K12 means a slower opponent.