Mr. Giansante

Visual Basic

Tron Light Cycles

August 2016

Tron Light Cycles

Difficulty Level:

Beginner
Intermediate
Advanced

Help References:

PictureBox Control
Graphics Commands

Design a Visual Basic program that simulates the **Light Cycles** portion of the arcade game **Tron**.

Tron was an 80's arcade game based on the 1982 Disney movie by the same name. Disney is planning on releasing Tron Legacy in December 2010.

In the original movie, the MCP (Master Control Program) wanted to control the entire system for evil purposes. User Programs (the good guys) were forced to compete in games. Programs that lost were deresed

In one of the games, devices resembling motor cycles were used to race on a grid. As they crossed the grid, they left a trail (Jet Walls) where they had been. These trails are like walls. If a Program on a Light Cycle ran into a wall, he crashed and was deresed.

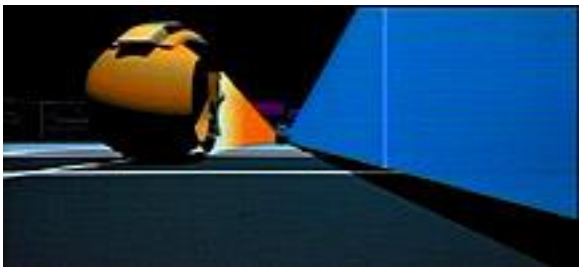
Two Light Cycles leave the playing field from opposite sides, heading for each other. One Light Cycle leaves a blue trail, and the other a gold (or yellow) trail. The goal of the game is to stay alive longer than the other player. Players die by running off the edge of the playing field, or running into the trails left by their opponent's Light Cycle or their own Light Cycle.

A running score should be kept and displayed.

Player 1 should use: [W] [A] [S] [D]
Player 2 should use the numerical keypad.

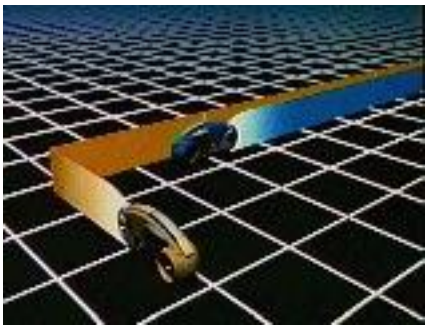
Tron Light Cycles

The following are screenshots from the Light Cycle scene of the movie Tron ...



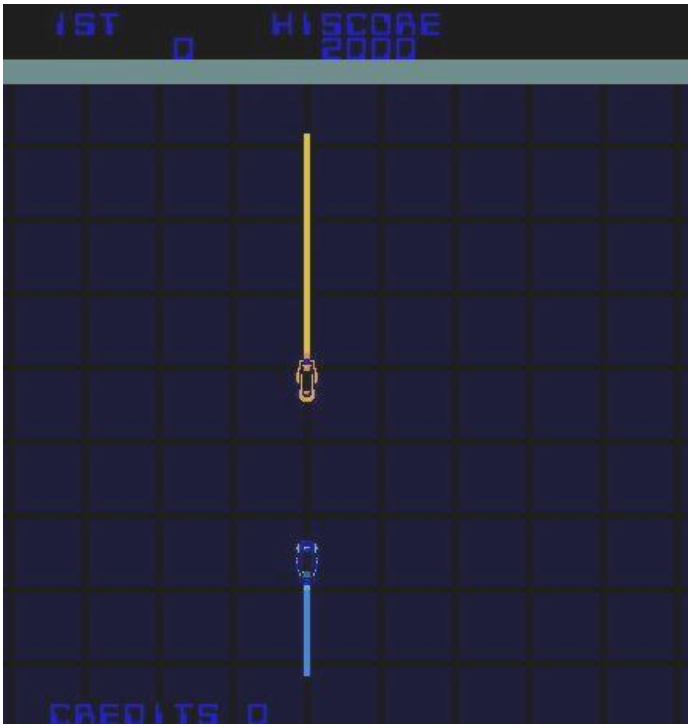
Yellow Light Cycle zips beside the trail of the blue light cycle.

Two Light Cycles collide.



Yellow Light Cycle tries to cut off blue light cycle.

The following is a screenshot from the actual arcade game Tron ...



Tron Light Cycles

Keeping Track of the "Jet Walls"

The game grid should be a PictureBox control that is 600 by 600 pixels. Note: You can choose other dimensions if you want.

To keep track of where the players have been (ie. the "Jet Walls"), we will use a two-dimensional array of Boolean ("True" or "False") variables.

The dimensions of the array will be 600 by 600 (every pixel in the game grid will have a corresponding entry in the array).

At the start of every new game, the entire array will be set to "False".

When a player goes over a pixel, the corresponding entry in the array will be set to "True".

This will allow us to keep track of what pixels contain Jet Walls and therefore we can do "collision detection".

Variations to Game

Two Key Controls
Players use only two keys to control lightcycle. One key turns them to the left and one to the right (Note: left and right will depend on the direction they are going). This requires a little bit more code, but makes the game easier and more exciting.

Option to turn on "obstacles"
This can be accomplished with only a few lines of code since it only involves drawing rectangles (or circles) onto the PictureBox.

Four-player game
This does not require much more code (mostly cut-and-paste). In the four player game, the game should continue until only one player is left. In other words, the game should not end as soon as the first player dies. When a player dies, their lightcycle should no longer move, but their trail should remain.

Help for Tron

For each player, you must keep track of their current x and y location as well as their current direction of motion (ie. up, down, left or right).

In General | Declarations

```
Dim grid(0 To 600, 0 To 600) As Boolean

Dim x1 As Integer
Dim y1 As Integer
Dim d1 As Integer
Dim x2 As Integer
Dim y2 As Integer
Dim d2 As Integer

Const Dir_Up = 1
Const Dir_Left = 2
Const Dir_Down = 3
Const Dir_Right = 4
```

The Constants are used to make the code more readable. For example ...

```
If d1 = Dir_Up Then y1 = y1 - 5
If d1 = Dir_Down Then y1 = y1 + 5
```

(etc.)

Tron Light Cycles

Tron Pseudocode

To achieve the effect of continuous motion, you must use a **Timer** control. Every time the Timer event fires, you should update each player's position according to their direction. Virtually all of the code for this assignment will be in the Timer.
Note: Use only one Timer. You do not need separate Timers for each player.

The following code is required in the **Timer** control ...

Check direction of player 1 and update position variables (x, y) accordingly. This will require four `If-Then` statements.

```
If d1 = Dir_Up Then y1 = y1 - 5
```

Check direction of player 2 and update position variables (x, y) accordingly. This will require four `If-Then` statements.

Check if Player 1 has run off the screen. This will require four `If-Then` statements.

```
If x1 < 0 Then ...
```

Check if Player 2 has run off the screen. This will require four `If-Then` statements.

Check if Player 1's new position will be on an existing Jet Wall (ie. they lose).
This will require one `If-Then` statement along with checking the `grid` array.

```
If grid(x1, y1) = True Then ...
```

Check if Player 2's new position will be on an existing Jet Wall (ie. they lose).
This will require one `If-Then` statement along with checking the `grid` array.

Check if Player 1 and Player 2 will move to the exact same pixel (ie. both players lose and the game is a tie).
This will require one `If-Then` statement.

```
If x1 = x2 and y1 = y2 Then ...
```

Draw the players in their new position. This will require two `FillEllipse` statements.

Update the grid array to show the new Jet Walls from Player 1 ...

```
grid(x1, y1) = True
```

Update the grid array to show the new Jet Walls from Player 2 ...

Clearing the Grid Array for a New Game

At the start of each new game, the `grid` array must be "cleared" of any Jet Wall information from the previous game.

```
For i = 0 To 600
    For j = 0 To 600
        grid(i, j) = False
    Next
Next
```

Tron Light Cycles

Please fill out all the information in this column using a pen before getting this assignment marked.

_____
Name

_____
Date

_____
Class

Academic Honesty

The work I am submitting is completely my own creation and has not been copied from anyone else's work. If I have received help on this project, the names of those who have assisted are listed below.

_____
Signature

_____

_____

_____

Pre-Marking

The following people have pre-marked this assignment: (minimum of two)

_____

_____

_____

Deductions

Each ❑ which is checked indicates the item was not satisfactory and results in the loss of one mark.

Deduction for late assignment is 1 mark per day.

Deduction for program crashing is 2 marks.

User Interface

- ❑ Appropriate Text in Title Bar (Form.Text)
- ❑ Separators and Access Keys in Menu, Capitalization
- ❑ "Exit" DialogBox meets requirements
- ❑ "About" DialogBox meets requirements
- ❑ Efficient and Esthetically-Pleasing User Interface
- ❑ Appropriate controls are used
- ❑ Spelling and Grammar are correct

Code / Programming Style

- ❑ Variables are declared and logically named
- ❑ Code is commented where appropriate
- ❑ Code is indented and spaced to show
- ❑ Code is efficient

Work Ethic / Problem Solving Skills

- ❑ Problem Solving Skills are demonstrated
- ❑ Makes Productive Use of Time
- ❑ Only Seeks Help when Necessary

Program-Specific Criteria

- ❒ Game starts with all players displayed on screen
- ❒ Each player moves (on each Timer event)
- ❒ Each player moves at the same speed
- ❒ Each Player can be controlled using keyboard (with 2 or 4 keys)
- ❒ Trail is left as player moves
- ❒ Running off the screen results in player losing
- ❒ Running into a trail results in player losing
- ❒ Players cannot go through trail
- ❒ Head-to-head crash results in a tie
- ❒ Winner is displayed
- ❒ Screen is cleared after each game
- ❒ Players start new game in proper position
- ❒ Score is kept track of
- ❒ "New Game" button resets scores

Comments

_____

_____

_____

Mark _____ / 10