Detailed Game Specification:  
TRON-Lightcycle

Course: COMP 2659, Winter 2017  
Instructor: Paul Pospisil  
  
Author(s): Jaden McConkey, Phillip Renwick  
Last Modified: Jan 15th 2017

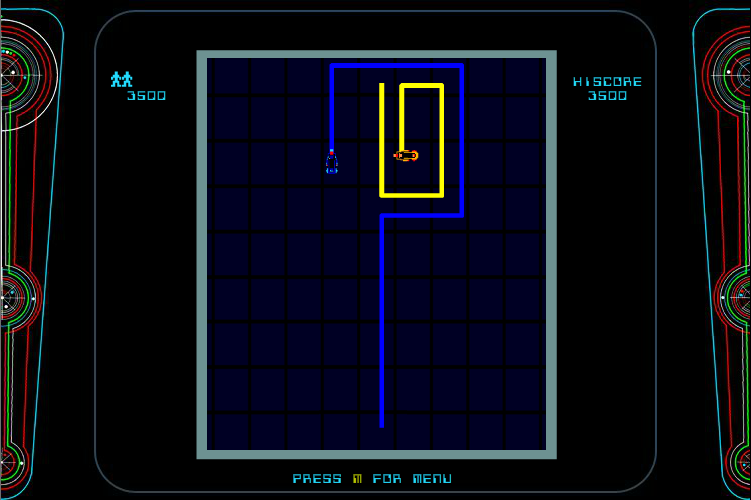
# 1. General Game Overview

Tron – light cycle is a 2D territory claiming skill based racing game. The aim is to survive longest within the confines of the playing field while also interfering with the survival/play of the competing player.

Play is controlled via the WASD keys or the arrow keys (user preference) each player directly controls a “lightcycle” that can move up,down,left and right within the rectangular playing field. For both players any movement is trailed by a “light wall” with collision, if either player comes into contact with either the field wall, their own light wall or the opposing players light wall, they have died and the opposing player win.

Gameplay is primarily two player, however there will also be a single person timed training mode to allow for practice in this highly skill based game.

Example Gameplay Screen(sample taken from the Tron arcade Game ) (will be greyscale)



# 2. Game Play Details for Core 1-Player Version

Player must navigate a circuit to reach the finish state.

As levels progress, the circuit will become more confining and longer, increasing the challenge to reach the goal.

-stretch: may attempt to allow for a survival mode via opening and closing "gates" in the circuit to allow an unending path to be available and ever changing.

## Objectives and Rules

Players seek to eliminate the opponent via trapping them.

Movement is 4-directional, up down, left, right, with turns being instantaneous, and oriented to the current direction of travel. Players may alter their velocity via up or down arrow respectively, but may not stop.

Coming into contact with a wall or enemy trail will result in their destruction.

## Objects

|  |  |  |  |
| --- | --- | --- | --- |
| Object or Object Type Name | Properties | Behaviours | Graphical Image |
| Light Cycle | * position (integer pair) * speed (integer) * direction | * Accelerate * Turn * Collision(death) |  |
| LightCycle Trail | * past path * Obstacle, fatal | * Laid by cycles * lasts until match end |  |

[Note: for some games, it may not be appropriate or realistic to include images for all object types. However, this is recommended if possible. Simple bitmap-style images (e.g. 8×8, 4×4, 16×16, etc.) are best for many game types. There are free tools to help you construct such images.]

[Note: for some games a particular object may have >1 associated image, depending on its current state (e.g. Mario walking left vs. Mario walking right).]

## Physics

Velocity of the lightcycles is 4 pixels per tick.

Acceleration is instantaneous. Pressing forwards adds 50% speed, back subtracts 50%.

Passing into a cell occupied by either an opponent or trail is considered to be a collision.

## Asynchronous (Input) Events

[Note: keyboard input is required of all games. Additional mouse input is optional for core game play.]

|  |  |  |
| --- | --- | --- |
| Event Name | Triggering Input Event | Description |
| Accelerate | Up arrow | Speed increased by 50%(+2px/sec)  Increase pitch of "engine" tone |
| Decelerate | Down Arrow | Speed decreased by 50%(-2px/sec)  Decrease pitch of "engine tone |
| Turn Left | Left Arrow | Turns left, relative to orientation |
| Turn Right | Right Arrow | Turns right, relative to orientation |

## Synchronous (Timed) Events

[Note: on the Atari ST, one easy option will be to make use of a 70 Hz timer (i.e. 70 ticks per second). So, it may be easiest to base timed events on multiples of 1/70th of a second.]

[Note: the trigger for a synchronous event is typically based on a clock. But, some synchronous events are also triggered by the occurrence of other synchronous events. E.g. a ship may move forward 1 pixel every 1/10th of a second, but it might as a result collide with an obstacle – these are two distinct events, one triggered directly by clock ticks and the other conditionally triggered by the original movement.]

|  |  |  |
| --- | --- | --- |
| Event Name | Trigger Timing | Description |
| Movement phase | 1/5 ticks  0.0714 sec. | * Lightcycle positions updated * Collisions checked |

## Condition-Based (Cascaded) Events

[Note: some events may trigger other events, conditionally. This is called event “cascading”. The triggering event(s) may themselves be asynchronous, synchronous, or other condition-based events.]

|  |  |  |
| --- | --- | --- |
| Event Name | Triggering Condition | Description |
| Init | Match Start | Cycles are placed at opposite ends of the grid, oriented towards each other, and placed in motion |
| Release | Init | Control over the cycles is given to the players. |
| Collision | Cycle has moved into cell occupied by wall or trail. | This is fatal, other player is victorious, round ends. |

## Hypothetical Gaming Session

Players begin at opposite ends, in motion towards each other.

Players maneuver to cut off the other player, and to try and trap them.

Inevitably, one player will find themselves trapped by some combination of the walls and the trails left by both players. Collision with either a wall or trail is inevitable, the only question is who will lose first.

# 3. Game Play Details for Core 2-Player Version

Gameplay is unchanged from the one player version, except that input for the other player comes not from scripted "ai", but from the other player instead.

Each player sees the map as if they played single player. At game start, both will see precisely the same viewpoint, each starting from the bottom, moving upwards, each seeing the other player as the darker colored player.

As players provide input, that input will be transmitted to the other players, enabling their client to know what actions the other player has taken, and to update its state accordingly.

It is necessary to send the player's actions in order to update the other client.

[2-player sample screenshot goes here.]

# 4. Sound Effects

|  |  |  |
| --- | --- | --- |
| Sound Effect Name | Brief Description | Event which Triggers Playback |
| Theme | Tron Themed tonal | Menu |
| Chime | Match start count down beeper | Init |
| Engine low | Lower pitch engine tone | Player decelerates |
| Engine mid | Standard engine tone | Player not (de/ac)celerate |
| Engine high | Higher pitch engine tone | Player accelerates |
| Collision | White noise "explosion" | Collision |

[Optional: brief description of background music. Music will be required by assignment 3.]

# 5. Additional Features (Time Permitting)

-toggling gates to enable different circuits

-More than two cycles in play. Potentially players.

-Fixed ai patterns to battle against with a total of 12 stages with increasing dificulty

-mode for playing on the same keyboard, mapping wasd for one player, arrows for the other.