jakesgordon corrected cameraDepth/fieldOfView calculations

0ab4d16 on Jun 23, 2012

1 contributor

Raw | Blame | History

314 lines (266 sloc) | 14.7 KB

```
<!DOCTYPE html>

<html>
<head>
  <title>Javascript Racer - v1 (straight)</title>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
  <link href="common.css" rel="stylesheet" type="text/css" />
</head>

<body>

  <table id="controls">
    <tr>
      <td colspan="2">
        <a href='v1.straight.html'>straight</a> |
        <a href='v2.curves.html'>curves</a>     |
        <a href='v3.hills.html'>hills</a>       |
        <a href='v4.final.html'>final</a>
      </td>
    </tr>
    <tr><td id="fps" colspan="2" align="right"></td></tr>
    <tr>
```

```
      <th><label for="resolution">Resolution :</label></th>
      <td>
        <select id="resolution" style="width:100%">
          <option value='fine'>Fine (1280x960)</option>
          <option selected value='high'>High (1024x768)</option>
          <option value='medium'>Medium (640x480)</option>
          <option value='low'>Low (480x360)</option>
        </select>
      </td>
    </tr>
    <tr>
      <th><label for="lanes">Lanes :</label></th>
      <td>
        <select id="lanes">
          <option>1</option>
          <option>2</option>
          <option selected>3</option>
          <option>4</option>
        </select>
      </td>
    </tr>
    <tr>
      <th><label for="roadWidth">Road Width (<span id="currentRoadWidth"></span>) :<
      <td><input id="roadWidth" type='range' min='500' max='3000' title="integer (50
    </tr>
    <tr>
      <th><label for="cameraHeight">CameraHeight (<span id="currentCameraHeight"></s
      <td><input id="cameraHeight" type='range' min='500' max='5000' title="integer
    </tr>
    <tr>
      <th><label for="drawDistance">Draw Distance (<span id="currentDrawDistance"></
      <td><input id="drawDistance" type='range' min='100' max='500' title="integer (
    </tr>
    <tr>
      <th><label for="fieldOfView">Field of View (<span id="currentFieldOfView"></sp
      <td><input id="fieldOfView" type='range' min='80' max='140' title="integer (80
    </tr>
    <tr>
      <th><label for="fogDensity">Fog Density (<span id="currentFogDensity"></span>)
      <td><input id="fogDensity" type='range' min='0' max='50' title="integer (0-50)
    </tr>
  </table>

  <div id='instructions'>
    <p>Use the <b>arrow keys</b> to drive the car.</p>
  </div>

  <div id="racer">
```

```
71      <canvas id="canvas">
72        Sorry, this example cannot be run because your browser does not support the &l
73      </canvas>
74      Loading...
75    </div>
76
77    <audio id='music'>
78      <source src="music/racer.ogg">
79      <source src="music/racer.mp3">
80    </audio>
81    <span id="mute"></span>
82
83    <script src="stats.js"></script>
84    <script src="common.js"></script>
85    <script>
86
87      var fps            = 60;                      // how many 'update' frames per sec
88      var step           = 1/fps;                   // how long is each frame (in secon
89      var width          = 1024;                    // logical canvas width
90      var height         = 768;                     // logical canvas height
91      var segments       = [];                      // array of road segments
92      var stats          = Game.stats('fps');       // mr.doobs FPS counter
93      var canvas         = Dom.get('canvas');       // our canvas...
94      var ctx            = canvas.getContext('2d'); // ...and its drawing context
95      var background     = null;                    // our background image (loaded bel
96      var sprites        = null;                    // our spritesheet (loaded below)
97      var resolution     = null;                    // scaling factor to provide resolu
98      var roadWidth      = 2000;                    // actually half the roads width, e
99      var segmentLength  = 200;                     // length of a single segment
100     var rumbleLength   = 3;                       // number of segments per red/white
101     var trackLength    = null;                    // z length of entire track (comput
102     var lanes          = 3;                       // number of lanes
103     var fieldOfView    = 100;                     // angle (degrees) for field of vie
104     var cameraHeight   = 1000;                    // z height of camera
105     var cameraDepth    = null;                    // z distance camera is from screen
106     var drawDistance   = 300;                     // number of segments to draw
107     var playerX        = 0;                       // player x offset from center of r
108     var playerZ        = null;                    // player relative z distance from
109     var fogDensity     = 5;                       // exponential fog density
110     var position       = 0;                       // current camera Z position (add p
111     var speed          = 0;                       // current speed
112     var maxSpeed       = segmentLength/step;      // top speed (ensure we can't move
113     var accel          =  maxSpeed/5;             // acceleration rate - tuned until
114     var breaking       = -maxSpeed;               // deceleration rate when braking
115     var decel          = -maxSpeed/5;             // 'natural' deceleration rate when
116     var offRoadDecel   = -maxSpeed/2;             // off road deceleration is somewhe
117     var offRoadLimit   =  maxSpeed/4;             // limit when off road deceleration
118
```

```
    var keyLeft       = false;
    var keyRight      = false;
    var keyFaster     = false;
    var keySlower     = false;

    //=========================================================================
    // UPDATE THE GAME WORLD
    //=========================================================================

    function update(dt) {

      position = Util.increase(position, dt * speed, trackLength);

      var dx = dt * 2 * (speed/maxSpeed); // at top speed, should be able to cross f

      if (keyLeft)
        playerX = playerX - dx;
      else if (keyRight)
        playerX = playerX + dx;

      if (keyFaster)
        speed = Util.accelerate(speed, accel, dt);
      else if (keySlower)
        speed = Util.accelerate(speed, breaking, dt);
      else
        speed = Util.accelerate(speed, decel, dt);

      if (((playerX < -1) || (playerX > 1)) && (speed > offRoadLimit))
        speed = Util.accelerate(speed, offRoadDecel, dt);

      playerX = Util.limit(playerX, -2, 2);     // dont ever let player go too far o
      speed   = Util.limit(speed, 0, maxSpeed); // or exceed maxSpeed

    }

    //=========================================================================
    // RENDER THE GAME WORLD
    //=========================================================================

    function render() {

      var baseSegment = findSegment(position);
      var maxy        = height;

      ctx.clearRect(0, 0, width, height);

      Render.background(ctx, background, width, height, BACKGROUND.SKY);
      Render.background(ctx, background, width, height, BACKGROUND.HILLS);
```

```
    Render.background(ctx, background, width, height, BACKGROUND.TREES);

    var n, segment;

    for(n = 0 ; n < drawDistance ; n++) {

      segment        = segments[(baseSegment.index + n) % segments.length];
      segment.looped = segment.index < baseSegment.index;
      segment.fog    = Util.exponentialFog(n/drawDistance, fogDensity);

      Util.project(segment.p1, (playerX * roadWidth), cameraHeight, position - (se
      Util.project(segment.p2, (playerX * roadWidth), cameraHeight, position - (se

      if ((segment.p1.camera.z <= cameraDepth) || // behind us
          (segment.p2.screen.y >= maxy))          // clip by (already rendered) se
        continue;

      Render.segment(ctx, width, lanes,
                     segment.p1.screen.x,
                     segment.p1.screen.y,
                     segment.p1.screen.w,
                     segment.p2.screen.x,
                     segment.p2.screen.y,
                     segment.p2.screen.w,
                     segment.fog,
                     segment.color);

      maxy = segment.p2.screen.y;
    }

    Render.player(ctx, width, height, resolution, roadWidth, sprites, speed/maxSpe
                  cameraDepth/playerZ,
                  width/2,
                  height,
                  speed * (keyLeft ? -1 : keyRight ? 1 : 0),
                  0);
  }

  //=========================================================================
  // BUILD ROAD GEOMETRY
  //=========================================================================

  function resetRoad() {
    segments = [];
    for(var n = 0 ; n < 500 ; n++) {
      segments.push({
          index: n,
          p1: { world: { z:  n   *segmentLength }, camera: {}, screen: {} },
```

```
        p2: { world: { z: (n+1)*segmentLength }, camera: {}, screen: {} },
        color: Math.floor(n/rumbleLength)%2 ? COLORS.DARK : COLORS.LIGHT
      });
    }

    segments[findSegment(playerZ).index + 2].color = COLORS.START;
    segments[findSegment(playerZ).index + 3].color = COLORS.START;
    for(var n = 0 ; n < rumbleLength ; n++)
      segments[segments.length-1-n].color = COLORS.FINISH;

    trackLength = segments.length * segmentLength;
  }

  function findSegment(z) {
    return segments[Math.floor(z/segmentLength) % segments.length];
  }

  //=========================================================================
  // THE GAME LOOP
  //=========================================================================

  Game.run({
    canvas: canvas, render: render, update: update, stats: stats, step: step,
    images: ["background", "sprites"],
    keys: [
      { keys: [KEY.LEFT,  KEY.A], mode: 'down', action: function() { keyLeft   = t
      { keys: [KEY.RIGHT, KEY.D], mode: 'down', action: function() { keyRight  = t
      { keys: [KEY.UP,    KEY.W], mode: 'down', action: function() { keyFaster = t
      { keys: [KEY.DOWN,  KEY.S], mode: 'down', action: function() { keySlower = t
      { keys: [KEY.LEFT,  KEY.A], mode: 'up',   action: function() { keyLeft   = f
      { keys: [KEY.RIGHT, KEY.D], mode: 'up',   action: function() { keyRight  = f
      { keys: [KEY.UP,    KEY.W], mode: 'up',   action: function() { keyFaster = f
      { keys: [KEY.DOWN,  KEY.S], mode: 'up',   action: function() { keySlower = f
    ],
    ready: function(images) {
      background = images[0];
      sprites    = images[1];
      reset();
    }
  });

  function reset(options) {
    options       = options || {};
    canvas.width  = width  = Util.toInt(options.width,          width);
    canvas.height = height = Util.toInt(options.height,         height);
    lanes                  = Util.toInt(options.lanes,          lanes);
    roadWidth              = Util.toInt(options.roadWidth,      roadWidth);
    cameraHeight           = Util.toInt(options.cameraHeight,   cameraHeight);
```

```
    drawDistance           = Util.toInt(options.drawDistance,   drawDistance);
    fogDensity             = Util.toInt(options.fogDensity,     fogDensity);
    fieldOfView            = Util.toInt(options.fieldOfView,    fieldOfView);
    segmentLength          = Util.toInt(options.segmentLength,  segmentLength);
    rumbleLength           = Util.toInt(options.rumbleLength,   rumbleLength);
    cameraDepth            = 1 / Math.tan((fieldOfView/2) * Math.PI/180);
    playerZ                = (cameraHeight * cameraDepth);
    resolution             = height/480;
    refreshTweakUI();

    if ((segments.length==0) || (options.segmentLength) || (options.rumbleLength))
      resetRoad(); // only rebuild road when necessary
  }

  //=========================================================================
  // TWEAK UI HANDLERS
  //=========================================================================

  Dom.on('resolution', 'change', function(ev) {
    var w, h, ratio;
    switch(ev.target.options[ev.target.selectedIndex].value) {
      case 'fine':   w = 1280; h = 960;  ratio=w/width; break;
      case 'high':   w = 1024; h = 768;  ratio=w/width; break;
      case 'medium': w = 640;  h = 480;  ratio=w/width; break;
      case 'low':    w = 480;  h = 360;  ratio=w/width; break;
    }
    reset({ width: w, height: h })
    Dom.blur(ev);
  });

  Dom.on('lanes',          'change', function(ev) { Dom.blur(ev); reset({ lanes:
  Dom.on('roadWidth',      'change', function(ev) { Dom.blur(ev); reset({ roadWidt
  Dom.on('cameraHeight',   'change', function(ev) { Dom.blur(ev); reset({ cameraHe
  Dom.on('drawDistance',   'change', function(ev) { Dom.blur(ev); reset({ drawDist
  Dom.on('fieldOfView',    'change', function(ev) { Dom.blur(ev); reset({ fieldOfV
  Dom.on('fogDensity',     'change', function(ev) { Dom.blur(ev); reset({ fogDensi

  function refreshTweakUI() {
    Dom.get('lanes').selectedIndex = lanes-1;
    Dom.get('currentRoadWidth').innerHTML      = Dom.get('roadWidth').value      =
    Dom.get('currentCameraHeight').innerHTML   = Dom.get('cameraHeight').value   =
    Dom.get('currentDrawDistance').innerHTML   = Dom.get('drawDistance').value   =
    Dom.get('currentFieldOfView').innerHTML    = Dom.get('fieldOfView').value    =
    Dom.get('currentFogDensity').innerHTML     = Dom.get('fogDensity').value     =
  }

  //=========================================================================
```

```
311     </script>
312
313 </body>
```