Milestone 2 - C4 Character Controller

Spring 2010

Due: 02/19/2010 11:55:00 PM

Assignment Type: *Work individually.* Discussing algorithms, design approaches, bugs, etc., is perfectly fine. Sharing code is NOT acceptable. Please document sources of help in your README, especially if you felt the help was significant. The source of help could be a student, newsgroup, web page, etc.

Late Policy:

2^(n+1) late policy point penalty. The details are the same as in M1.

Description:

For this assignment, you will be creating a character controller and model for a single-wheeled vehicle (unicycle/unibike/monobike/etc.). See bottom of document for some concept designs that you can use for inspiration, however you should be creating a much simpler design for M2. The unicycle controller will respond to user input by causing your custom model to interact with the world. The unicycle will be able to traverse ground surfaces, jump, change speed, animate via a programmatic rolling animation, and make sounds triggered by various events such as collisions or rolling. YOU WILL NOT BE USING PHYSX OR ANY OTHER PHYSICS LIBRARY.

We highly recommend that you make your controller by carefully studying the provided GTDemoChar code. Also, please be sure to read all provided documentation.

Your controller will be tested and graded in the "ring.wld" map, however most any map with a spawn point should work as well.

For the audio requirements, we recommend that you use sounds already available in the virtual data directory structure. The Demo data directory contains many sounds under the sound directory as well as the model directories for the weapons.

Requirements:

You custom controller needs to implement the following requirements:

1.) Your code compiles and the binary loads with the map, "ring.wld" *(10 points)*
2.) Create a textured unicycle model (in the C4 world editor), which your controller loads and uses *(10 points)*
   a. *The unicycle model should consist of a wheel that rolls and a base that remains stationary relative to the wheel rotation.*

   b. *The unicycle should be created using primitive shapes within the C4 editor*

3.) Your unicycle is viewed in the 3rd person by a chase camera *(5 points)*

4.) Unicycle can be moved around in the world via keyboard and mouse *(5 points)*

5.) A keyboard/mouse button mapped to "%jump;" causes the unicycle to jump ONCE and then fall back to the ground before jumping again (e.g. if you hold down the jump button the unicycle should not continue to fly into the air). Note: keys are mapped in the provided GTDemoChar engine.cfg file. Please don't modify unless you really feel you need to. *(5 points)*

6.) A keyboard/mouse button mapped to "%sped;" causes the unicycle to travel around at an increased speed. See 5. for key mapping details. *(10 points)*

7.) The unicycle controller must support collision with geometry and be stopped from penetrating. A sound should be played when a collision is hard enough. (See GTDemoChar for an example.) *(15 points)*

8.) The unicycle wheel animates (rolls) in the appropriate direction when moving and is based on the size and speed of the wheel, as well as the ground surface it is rolling on. Assume the wheel rolls perfectly with no sliding. *(15 points)*

9.) The unicycle should lean smoothly according to acceleration/deceleration and turning. There should be no sudden "snapping" to different orientations, but instead gradual leaning that is animated over time. This animation does NOT need to be physically accurate (i.e. it can be exaggerated or cartoon-like). (*10 points)*

10.) When the unicycle wheel is rolling, it should make a sound. When not rolling, it should make no sound. The volume of the sound should become louder as the model moves faster. While not required, varying pitch slightly as well gives improved results. *(10 points)*

11.) Submit your assignment as a ZIP file with the following details:

   a. Your name should appear on the HUD of your game when it is running. (if modifying GTDemoChar, simply change the "stuName" string param passed to the DemoController's constructor)
   b. ZIP file name: <last_name><first_initial>_m2.zip
   c. **Readme file**: <last_name><first_initial>_m2_readme.txt
      i. Full name, email, and prism account name
      ii. Detail which requirements you have completed, which are incomplete, and which are buggy (be specific)
      iii. Detail any special install instructions the grader will need to be aware of for compiling and running your code
         1. Don't forget to let us know if you changed kGroundProbeDepth in C4Character.cpp as detailed in the GTDemoChar_README.doc
      iv. Detail sources of help as mentioned in "assignment type" section of this document
      v. Be very specific! (You will probably want to try following your own instructions to make sure they are accurate.)
   d. Model file: <last_name><first_initial>_m2_model.mdl
   e. Model *world* file: <last_name><first_initial>_m2_model.wld

        i. If you used a 3rd party modeling program instead of C4's world editor, provided the original model, and collada file as well

f. Your source files (most likely UnicycleChar.cpp and UnicycleChar.h, but could include more)

g. Any additional media files needed (textures, sounds, etc.)

        i. Don't forget instructions about which virtual directory path to use

h. Any special instructions to grader about testing your extra credit submission. Extra credit won't get graded if you don't inform the grader via your write-up!

(5 points)

11.) Be sure to save all your files in the state they were at submission time in case we have any problems with grading (such as forgetting to submit a file we need)!

Extra Credit:

1.) Add a particle system that creates a shower of sparks or debris at the point of a collision. Feel free to make use of the SimpleBall example. (+3 points)
2.) Add a gun that is attached to the unicycle base that shoots (when you press fire) in the direction the camera is facing. The gun should rotate to face the direction of camera view at all times. (Up to 5 points)
3.) Add a "jump pad" to your game DLL that allows placement of jump pads with the game editor. There is an example in the Game.dll, however yours should only add velocity to the Z axis rather than direct the player to a specific location. You should also create a demo map with jump pads present (name it <last_name><first_initial>_m2_extra.wld). (Up to 5 points)

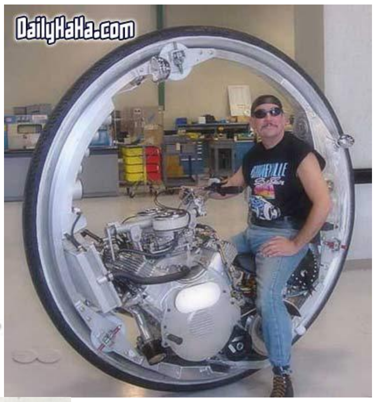
Submission:

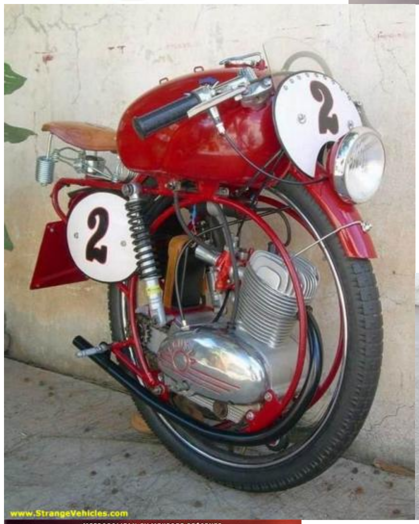
You must submit via T-Square. If submitting late, you should still use T-Square. The site will be configured to accept late assignments. If there is any technical problem with T-Square at submission time, you may attempt to email your assignment. If email is necessary, please email your zip file to the TA and instructor(s) with a subject that starts with "CS4455 M<current_assignment #> Submission".

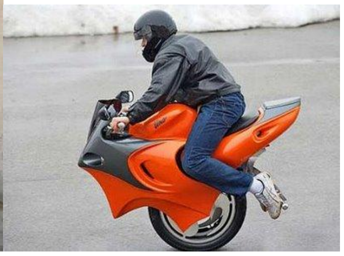
Tips:

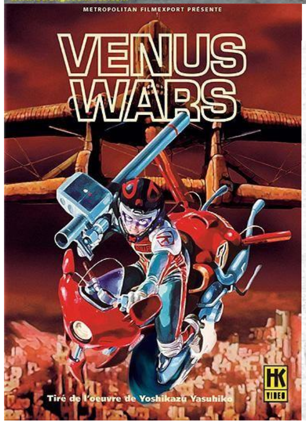
The C4 Wiki has several tutorials that will help you with this assignment. The example code we are providing should be quite helpful as well. *Be sure to read the other documents and source code comments in the support materials we are providing.*

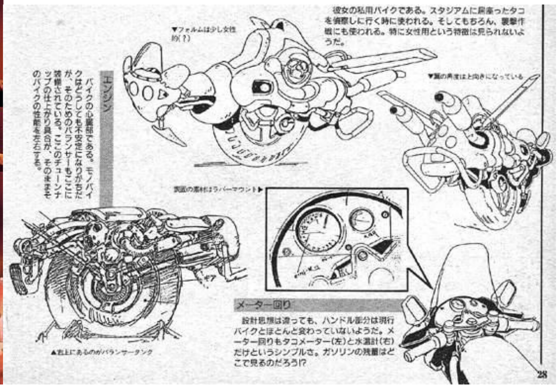
Some visual examples of single-wheeled vehicles:








METROPOLITAN FILMEXPORT PRÉSENTE
VENUS WARS
Tiré de l'oeuvre de Yoshikazu Yasuhiko
HK VIDEO



彼女の私用バイクである。スタジアムに居座ったタコを偵察しに行く時に使われる。そしてもちろん、襲撃作戦にも使われる。特に女性用という特徴は見られないようだ。

▼フォルムは少し女性的(?)

▼翼の角度は上向きになっている

エンジン

バイクの心臓部である。モノバイクはどうしても不安定になりがちだが、そのためのバランサーもここに装備されている。ここのチューンナップの仕上がり具合が、そのままそのバイクの性能を左右する。

表面の素材はラバーマウント▶

▲右上にあるのがバランサータンク

メーター回り

設計思想は違っても、ハンドル部分は現行バイクとほとんど変わっていないようだ。メーター回りもタコメーター(左)と水温計(右)だけというシンプルさ。ガソリンの残量はどこで見るのだろう!?

28

© 2007 Michael Fichtenmayer www.fichtenfoo.com

Rocket Roadster
panda
USA
PRECISION
panda