Peer Review Activity

LEWIS COMSTIVE – MATHS FOR GAMES  
6th April 2021

|  |  |
| --- | --- |
| Author of project being reviewed: | James Greensill |
| Reviewer: | Lewis Comstive |
| Date: | 6/4/2021 |
| Does the code conform to a consistent coding standard? Note the relevant coding standard and list places where the code can be improved | The code conforms to a consistent standard.  Private variables use the ***m\_camelCase*** naming convention; m*ethods, class names, events* & *non-private variables* use the ***PascalCase*** convention.  Class files are sorted into appropriate subfolders, determining the namespace used per file.  The #region pre-processor directive is used to separate some files for easier readability. Regions for separating member variables, properties, constructors and methods can be seen in many classes.  The codebase has well-named variables, methods and classes, but could benefit in some cases from comments. (e.g., Matrix3.RotateZ(float rotation) – is the rotation in degrees or radians?) |
| Is the code well commented, easy to read and understand? List at least one area for improvement or practice you can apply to your own programming | The code is easy to read and understand, largely due to the compartmentalised methods and clear naming. |
| Does the program function as intended? Comment on the mechanics of the application. Note any variation from the brief. Does the program perform identically on different machines? | The program does function as intended.  Bullets shot from the tank wrap around to the opposing edge of the screen it collided with.  Bullets are destroyed after some time, or if they have collided with another object, such as a tree.  No observed bugs nor issues, consistent framerate.  Testing computer showed sub-70MB memory usage.  Visual Studio’s profiling tools revealed that a majority of CPU time is spent on Raylib, the external dependency used for displaying graphics.  Observations indicate that the executable functions the same on different machines. |
| Is the code well structured? List at least one area for improvement or practice you can apply to your own programming. | The code is structured well, using inheritance OOP for gameobject logic. |
| Is vector and matrix math used correctly to draw and manipulation the position and orientation of the game objects? Note any differences in how calculations are performed between this program and your own. | The maths classes provided with the solution worked as intended, correctly displaying objects using a local and global transform matrix with translation and rotation. |
| Is there anything else noteworthy? | No. |
| How would you rate the quality of this project? | 9/10 |
| What steps could be taken to resolve any quality issues? | There were no observed quality issues. |
| Was your own math class able to be used? | *After some tweaking, yes.*  Differences in variable names.  (*e.g., Vector2.m\_x vs Vector2.x*)  Rotation values were passed in as degrees, different to my library which expected radians.  Matrix implementation seemed to be in row-major whereas my library uses a column-major approach. |