Wild Magic 5 Update History

Last modified: October 12, 2017

Contents

The version release dates are listed here. Versions released before the current version may be obtained by email request.

- Version 5.0 posted February 4, 2010 (with *Game Physics, 2nd edition*).
- Version 5.1 posted April 18, 2010.
- Version 5.2 posted July 8, 2010.
- Version 5.3 posted September 8, 2010.
- Version 5.4 posted October 1, 2010.
- Version 5.5 posted March 3, 2011.
- Version 5.6 posted July 30, 2011.
- Version 5.7 posted October 27, 2011.
- Version 5.8 posted July 8, 2012.
- Version 5.9 posted August 5, 2012.
- Version 5.10 posted July 15, 2013.
- Version 5.11 posted January 21, 2014.
- Version 5.12 posted July 2, 2014.
- Version 5.13 posted July 9, 2014.
- Version 5.14 posted November 27, 2015.
- Version 5.15 posted May 26, 2017.
- Version 5.16 posted August 25, 2017.
- Version 5.17 posted October 12, 2017.

The updated files and related notes are provided for the versions in each of the ensuing sections. Each section has a list of changes that occurred to the version number mentioned in that section. Those changes were rolled up into the zip file that was posted for the next version. Modified files are colored gold, new files are colored green, and deleted files are colored red. Source code is colored Violet.

1 Updates to Version 5.0

April 15, 2010. Replace the include of <cstring> by Wm5CoreLIB.h to be consistent with other library files.

(5.0.1) WildMagic5/LibCore/DataTypes/Wm5Tuple.h

April 15, 2010. TransformController is a new controller class that sets the controlled object's local transform to a specified value. This was added to support blended animations. KeyframeController was derived from Controller but is now derived from TransformController. This removes an implicit dependency on the application setting channels of the mObject LocalTransform that are not filled in by the key frames during an Update() call. The streaming system is affected by this change. See Documentation/StreamingChanges.txt. The streaming version is now WM_VERSION_5_1.

(5.1.0) WildMagic5/LibGraphics/Controllers/Wm5TransformController.cpp
(5.1.0) WildMagic5/LibGraphics/Controllers/Wm5TransformController.h
(5.1.0) WildMagic5/LibGraphics/Controllers/Wm5TransformController.inl
(5.0.1) WildMagic5/LibGraphics/Controllers/Wm5KeyframeController.cpp
(5.0.1) WildMagic5/LibGraphics/Controllers/Wm5KeyframeController.h
(5.0.1) WildMagic5/LibCore/ObjectSystems/Wm5InStream.h
(5.0.1) WildMagic5/LibCore/ObjectSystems/Wm5OutStream.h
(5.0.1) WildMagic5/LibGraphics/Wm5Graphics.h
(5.0.1) WildMagic5/SampleGraphics/SkinnedBiped.cpp
WildMagic5/Data/Wmof/FacePN.wmof
WildMagic5/Data/Wmof/FacePN.be.wmof
WildMagic5/Data/Wmof/SkinnedBipedPN.wmof
WildMagic5/Data/Wmof/SkinnedBipedPN.be.wmof
WildMagic5/LibGraphics/LibGraphics_VC90.vcproj
WildMagic5/LibGraphics/LibGraphics.xcodeproj/project.pbxproj

April 15, 2010. Added functions Clamp and Saturate.

(5.0.1) WildMagic5/LibMathematics/Base/Wm5Math.h
(5.0.1) WildMagic5/LibMathematics/Base/Wm5Math.inl

April 15, 2010. The Rational class did not support conversions of denormal floating-point numbers; now it does. The ConvertTo functions had a bug in them, so small rational numbers were not correctly converted. The algorithms for the conversions have been revised, producing a significant speed up. For example, the conversion of a float to a rational is 25 times faster.

(5.0.1) WildMagic5/LibMathematics/Rational/Wm5Rational.h
(5.0.1) WildMagic5/LibMathematics/Rational/Wm5Rational.inl
(5.0.1) WildMagic5/LibMathematics/Wm5MathematicsLIB.h

April 15, 2010. Added accessors SetApplicationTime and GetApplicationTime. This time is the last time the controller was updated. This gives you the ability to change the internal application time without calling Update(time). The Update(time) function no longer tests whether the current application time is different from the last update time. The update occurs only when Active is true. The SetObject function was made public. ControlledObject needs to call this during AttachController, but so do derived classes that manage sets of controllers.

(5.0.1) WildMagic5/LibGraphics/Controllers/Wm5Controller.h
(5.0.1) WildMagic5/LibGraphics/Controllers/Wm5Controller.cpp
(5.0.1) WildMagic5/LibGraphics/Controllers/Wm5Controller.inl

April 15, 2010. Removed a commented out function in the interface and the commented out bodies in the inline file. Added new member functions InverseTransform and Invert3x3. The Inverse() function was incorrect. Invert3x3 now does the general inversion for a 3×3.

(5.0.1) WildMagic5/LibGraphics/DataTypes/Wm5Transform.h
(5.0.1) WildMagic5/LibGraphics/DataTypes/Wm5Transform.cpp
(5.0.1) WildMagic5/LibGraphics/DataTypes/Wm5Transform.inl

April 15, 2010. The type for RGB565 needed to be GL_UNSIGNED_SHORT_5_6_5_REV.

(5.0.1) WildMagic5/LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLMapping.cpp

April 15, 2010. Added conditional compilation WM5_USE_TEXT_DISPLAY_LIST for the display list construction and destruction. This had already been done in the Renderer text-drawing function.

(5.0.1) WildMagic5/LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLRendererData.h
(5.0.1) WildMagic5/LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLRendererData.cpp
(5.0.1) WildMagic5/LibGraphics/Renderers/OpenGLRenderer/Wm5WglRenderer.cpp

April 15, 2010. KEY_SHIFT and KEY_CONTROL were initialized with the Win32 mouse shift/control flags (MK_SHIFT and MK_CONTROL). They needed to be initialized with the keyboard shift/control (VK_SHIFT and VK_CONTROL). OnSpecialKeyDown and OnSpecialKeyUp now report when shift/control is pressed for special keys. Multisampling caused the application to crash. The input.mWindowHandle needed to be set to the new window handle. Also, the DestroyWindow call causes the WM_DESTROY message, which posts a quit message. The application terminated as a result. I fixed this, but it is not clear why Wild Magic 4 does not suffer the same problem.

(5.0.1) WildMagic5/LibApplications/WinApplication/Wm5WinApplication.cpp

April 15, 2010. Modified the way shift, control, alt, and command keys are detected and processed. They are now correctly handled.

(5.0.1) WildMagic5/LibApplications/GlxApplication/Wm5GlxApplication.cpp

April 15, 2010. The keyboard event handlers were not configured to process shift, control, alt, and command keys. They are now, but unfortunately something is still amiss. When the SHIFT key is pressed, the keyboard handler detects this and calls OnSpecialKeyDown(KEY_SHIFT,...). With the key pressed, the operating system still generates a release message, which leads to a call OnSpecialKeyUp(KEY_SHIFT,...). I was unable to figure out how to prevent the key-released event (when the key was not even released).

(5.0.1) WildMagic5/LibApplications/AglApplication/Wm5AglApplication.cpp

April 15, 2010. Removed the post-build copy of Assert/*.inl, because there are no such files to copy.

WildMagic5/LibCore/LibCore_VC90.vcproj

April 15, 2010. New class to blend two transform controllers. This is used for animation blending.

(5.1.0) WildMagic5/LibGraphics/Controllers/Wm5BlendTransformController.h
(5.1.0) WildMagic5/LibGraphics/Controllers/Wm5BlendTransformController.cpp
(5.1.0) WildMagic5/LibGraphics/Controllers/Wm5BlendTransformController.inl
WildMagic5/LibGraphics/LibGraphics_VC90.vcproj
WildMagic5/LibGraphics/LibGraphics.xcodeproj/project.pbxproj

April 15, 2010. Added a new application for blending animations for a biped (idle, walk, and run cycles). Uses a real artist-generated data set.

(5.1.0) WildMagic5/SampleGraphics/BlendedAnimations/BlendedAnimations.cpp
(5.1.0) WildMagic5/SampleGraphics/BlendedAnimations/BlendedAnimations.h
(5.1.0) WildMagic5/SampleGraphics/BlendedAnimations/BipedManager.cpp
(5.1.0) WildMagic5/SampleGraphics/BlendedAnimations/BipedManager.h
(5.1.0) WildMagic5/SampleGraphics/BlendedAnimations/BipedManager.inl
WildMagic5/SampleGraphics/BlendedAnimations/Data/*
WildMagic5/SampleGraphics/BlendedAnimations_VC90.vcproj
WildMagic5/SampleGraphics/makefile.wm5
WildMagic5/SampleGraphics/BlendedAnimations.xcodeproj/*

April 15, 2010. Converted the castle data set from WM4 format to raw data for WM5. Added most of the lighting that was part of the original 3DS Max art. (The art work still needs help and was unfinished.)

(5.1.0) WildMagic5/SampleGraphics/Castle/Castle.cpp
(5.1.0) WildMagic5/SampleGraphics/Castle/Castle.h
(5.1.0) WildMagic5/SampleGraphics/Castle/CreateMeshes.cpp
(5.1.0) WildMagic5/SampleGraphics/Castle/DLitMatTexEffect.cpp
(5.1.0) WildMagic5/SampleGraphics/Castle/DLitMatTexEffect.h
(5.1.0) WildMagic5/SampleGraphics/Castle/LoadData.cpp
WildMagic5/SampleGraphics/Castle/Castle_VC90.vcproj
WildMagic5/SampleGraphics/Castle/Geometry/*
WildMagic5/SampleGraphics/Castle/Shaders/*
WildMagic5/SampleGraphics/Castle/Textures/*

April 15, 2010. Added a new application for a morphing face. Uses a real artist-generated data set.

(5.1.0) WildMagic5/SampleGraphics/MorphFaces/CubicInterpolator.h
(5.1.0) WildMagic5/SampleGraphics/MorphFaces/CubicInterpolator.inl
(5.1.0) WildMagic5/SampleGraphics/MorphFaces/MorphFaces.cpp
(5.1.0) WildMagic5/SampleGraphics/MorphFaces/MorphFaces.h
WildMagic5/SampleGraphics/MorphFaces/Data/*
WildMagic5/SampleGraphics/MorphFaces_VC90.vcproj
WildMagic5/SampleGraphics/makefile.wm5
WildMagic5/SampleGraphics/MorphFaces.xcodeproj/*

April 15, 2010. Added support for a command-line option: `WmfxCompiler -a MyFile.fx` The option says that the shaders were already compiled, so WmfxCompiler should not compile them and simply load them and create the *.wmfx file. This allows manual compilation for profiles not currently supported in Wild Magic's Shader/Profile system.

(5.0.1) WildMagic5/Tools/WmfxCompiler/WmfxCompiler.cpp
(5.0.1) WildMagic5/Tools/WmfxCompiler/FxCompiler.cpp
(5.0.1) WildMagic5/Tools/WmfxCompiler/FxCompiler.h

April 15, 2010. A simple Microsoft Windows application to convert 24-bit color images to gray-scale images. I used this for gray-scale images in *Game Physics, 2nd edition*.

(5.1.0) WildMagic5/Tools/BmpColorToGray/BmpColorToGray.cpp
(5.1.0) WildMagic5/Tools/BmpColorToGray/BmpColorToGray.h
WildMagic5/Tools/BmpColorToGray/BmpColorToGray.vcproj

April 15, 2010. A simple Microsoft Windows application to convert a 24-bit BMP image to the WM5 raw texture EMTF format.

(5.1.0) WildMagic5/Tools/BmpToWmtf/BmpToWmtf.cpp
(5.1.0) WildMagic5/Tools/BmpToWmtf/BmpToWmtf.h
WildMagic5/Tools/BmpToWmtf/BmpToWmtf.vcproj

April 15, 2010. Source code to import .obj and .mtl files. The importer does not implement all OBJ features, but it is usually sufficient to load OBJ/MTL files exported from 3DS Max and Maya.

(5.1.0) WildMagic5/Tools/ObjMtlImporter/ObjLoader.cpp
(5.1.0) WildMagic5/Tools/ObjMtlImporter/ObjLoader.h
(5.1.0) WildMagic5/Tools/ObjMtlImporter/ObjLoader.inl
(5.1.0) WildMagic5/Tools/ObjMtlImporter/ObjLoaderCodes.cpp
(5.1.0) WildMagic5/Tools/ObjMtlImporter/MtlLoader.cpp
(5.1.0) WildMagic5/Tools/ObjMtlImporter/MtlLoader.h
(5.1.0) WildMagic5/Tools/ObjMtlImporter/MtlLoader.inl
(5.1.0) WildMagic5/Tools/ObjMtlImporter/MtlLoaderCodes.cpp
WildMagic5/Tools/ObjMtlImporter/ReadMe.txt

2 Updates to Version 5.1

April 18, 2010. The assignment operator needed to copy the mQuantity member.

(5.0.1) LibImagics/Images/Wm5Lattice.cpp

April 18, 2010. Fixed compiler errors for Microsoft Visual Studio 2010 RC1, compiling for a 64-bit target on 32-bit Microsoft Windows XP, and with the latest STLport. The problem does not occur when compiling for a 32-bit target. The error message is "`Conversion from integral type to pointer type requires reinterpret_cast, C-style cast or function-style cast.`"

(5.0.1) LibMathematics/ComputationalGeometry/Wm5Delaunay2.cpp
(5.0.1) LibMathematics/ComputationalGeometry/Wm5Delaunay3.cpp
(5.0.1) LibMathematics/ComputationalGeometry/Wm5IncrementalDelaunay2.cpp

(5.0.1) LibMathematics/Meshes/Wm5ETManifoldMesh.cpp
(5.0.1) LibMathematics/Meshes/Wm5ETNonmanifoldMesh.cpp
(5.0.1) LibMathematics/Meshes/Wm5VEManifoldMesh.cpp

April 30, 2010. The connected component labelers were written in the mid 1990s based on a seminar I took on image analysis. The algorithm is effectively a union-find without the heuristics for speeds up (rank, path compression) and was relatively slow in 2D and very slow in 3D. I replaced these with simple nonrecursive depth-first traversals. The 2D labeling is approximately 3 times faster. The 3D labeling is approximately 100 times faster.

(5.0.1) LibImagics/BinaryOperations/Wm5Binary2D.h
(5.0.1) LibImagics/BinaryOperations/Wm5Binary2D.cpp
(5.0.1) LibImagics/BinaryOperations/Wm5Binary3D.h
(5.0.1) LibImagics/BinaryOperations/Wm5Binary3D.cpp
(5.0.1) SampleImagics/BinaryOperations/BinaryOperations.cpp

April 30, 2010. Rewrote Binary2D and Binary3D to be only a collection of static functions. The code now avoids all image copies (minimize memory, maximize speed). The BinaryOperations sample application now has component labeling for 8-connect and 4-connect (in 2D) and for 26-connect, 18-connect, and 6-connect (in 3D).

(5.0.2) LibImagics/BinaryOperations/Wm5Binary2D.h
(5.0.2) LibImagics/BinaryOperations/Wm5Binary2D.cpp
(5.0.2) LibImagics/BinaryOperations/Wm5Binary3D.h
(5.0.2) LibImagics/BinaryOperations/Wm5Binary3D.cpp
(5.0.1) SampleImagics/BinaryOperations/BinaryOperations.h
(5.0.2) SampleImagics/BinaryOperations/BinaryOperations.cpp

April 30, 2010. The console and window titles are now std::string so that OnPrecreate() can change them. This is safe to do, because the path system access the title strings before they can be changed. The ImViewer sample application now shows the image filename in the title bar.

(5.0.1) LibApplications/Wm5ConsoleApplication.h
(5.0.1) LibApplications/Wm5ConsoleApplication.cpp
(5.0.1) LibApplications/Wm5WindowApplication.h
(5.0.1) LibApplications/Wm5WindowApplication.inl
(5.0.1) SampleImagics/ImViewer/ImViewer.cpp

May 1, 2010. Sometimes it is convenient to have a childless dummy Node in a scene graph. The default WorldBound has a center of $(0, 0, 0)$ and a radius of 0. This affects the WorldBound for predecessors, which is a problem if $(0, 0, 0)$ is not even close to the scene (objects can be culled when in fact they are visible). I modified code in the Bound class so that a Bound with a zero radius is considered to be invalid and is ignored in Bound::GrowToContain and Bound::TestIntersection calls. The Culler::IsVisible call was modified to ignore bounds with zero radius.

(5.0.1) LibGraphics/DataTypes/Wm5Bound.cpp
(5.0.1) LibGraphics/SceneGraph/Wm5Culler.cpp

May 6, 2010. Added member mAllowResize with default value of true. If this is set to false in OnPrecreate(), a window is created that cannot be resized. The ImViewer sample application window now cannot be resized.

(5.0.2) LibApplications/Wm5WindowApplication.h
(5.0.1) LibApplications/Wm5WindowApplication.cpp
(5.0.2) LibApplications/WinApplication/Wm5WinApplication.cpp
(5.0.2) SampleImagics/ImViewer/ImViewer.cpp

May 11, 2010. The application window cannot be resized. The window title displays the filename.

(5.0.1) Tools/WmtfViewer/WmtfViewer.cpp

May 11, 2010. The IsConnected function had a begin() call that needed to be an end() call.

(5.0.2) LibMathematics/Meshes/Wm5ETNonmanifoldMesh.cpp

May 12, 2010. Even after testing, I failed to include the latest project that contains references to TransformController and BlendTransformController. The Xcode project does have these references. I have re-zipped the WildMagic5p1.zip file, but you can download the project file by itself or you can just drag Wm5TransformController.* and Wm5BlendTransformController.* into your local project.

LibGraphics/LibGraphics_VC90.vcproj

May 20, 2010. An error was introduced when converting the member names from WM4 to WM5. The member F needed to be mF.

(5.0.1) LibMathematics/Interpolation/Wm5IntpBilinear2.h
(5.0.1) LibMathematics/Interpolation/Wm5IntpBilinear2.cpp

June 21, 2010. Added two new physics samples. The CollisionsMovingSpheres sample computes the first contact time and first contact point for two spheres moving with constant linear velocity (continuous-in-time algorithm, not a discrete time stepper with bisection). The CollisionsMovingSphereTriangle sample computes the first contact time and first contact point for a sphere and a triangle, both moving with constant linear velocities (continuous-in-time algorithm, not a discrete time stepper with bisection).

(5.2.0) SamplePhysics/CollisionsMovingSpheres/CollisionsMovingSpheres.h
(5.2.0) SamplePhysics/CollisionsMovingSpheres/CollisionsMovingSpheres.cpp
(5.2.0) SamplePhysics/CollisionsMovingSpheres/Colliders.h
(5.2.0) SamplePhysics/CollisionsMovingSpheres/Colliders.cpp
(5.2.0) SamplePhysics/CollisionsMovingSpheres/SphereColliders.h
(5.2.0) SamplePhysics/CollisionsMovingSpheres/SphereColliders.cpp
SamplePhysics/CollisionsMovingSpheres/CollisionMovingSpheres_VC90.vcproj
(5.2.0) SamplePhysics/CollisionsMovingSphereTriangle/CollisionsMovingSphereTriangle.h
(5.2.0) SamplePhysics/CollisionsMovingSphereTriangle/CollisionsMovingSphereTriangle.cpp
(5.2.0) SamplePhysics/CollisionsMovingSphereTriangle/RTSphereTriangle.h
(5.2.0) SamplePhysics/CollisionsMovingSphereTriangle/RTSphereTriangle.cpp
SamplePhysics/CollisionsMovingSpheres/CollisionsMovingSphereTriangle_VC90.vcproj

June 24, 2010. Newton's method for root finding was used for computing the curve parameter at which a specified arc length occurs. This is not robust when the arc length integral has second derivative that is sometimes negative. Now the code uses a hybrid of Newton's method and bisection, which is robust.

(5.0.1) LibMathematics/CurvesSurfacesVolumes/Wm5SingleCurve2.cpp
(5.0.1) LibMathematics/CurvesSurfacesVolumes/Wm5MultipleCurve2.cpp
(5.0.1) LibMathematics/CurvesSurfacesVolumes/Wm5SingleCurve3.cpp
(5.0.1) LibMathematics/CurvesSurfacesVolumes/Wm5MultipleCurve3.cpp
Documentation/MovingAlongCurveSpecifiedSpeed.pdf

June 28, 2010. Fixed some warnings for Microsoft Visual Studio 2010 about unused code (using Level 4 warnings). The warnings are incorrect (compiler bug).

(5.0.1) LibGraphics/CurvesSurfaces/Wm5BoxSurface.cpp

June 29, 2010. A bug was reported for a 64-bit Linux machine with Fedora 12. A static typecast that loses precision apparently generates an error rather than a warning. The static typecast was replaced with reinterpret_cast.

(5.0.1) LibCore/Memory/Wm5Memory.cpp

July 1, 2010. Change preprocessor #error to allow MSVS 2005 to run.

(5.0.1) LibCore/Wm5LibCoreLIB.h

July 1, 2010. The g++ shipped with Fedora 12 on a 64-bit machine still generated errors after the 29Jun2010 modification. The Wm5Memory code involved explicitly testing for 32-bit or 64-bit addresses and typecasting accordingly. This code was removed and now a pointer-address format is used for the address printing. Similar typecasting problems occurred when loading a *.wmof file from disk. The unique ID associated with an object pointer was a 32-bit integer and was changed to a 64-bit integer.

(5.0.2) LibCore/Memory/Wm5Memory.cpp
(5.0.2) LibCore/ObjectSystems/Wm5InStream.h
(5.0.1) LibCore/ObjectSystems/Wm5InStream.inl
(5.0.1) LibCore/ObjectSystems/Wm5InStream.cpp

July 6, 2010. The LLVM 2.0 compiler complained that the construction of static const PickRecord Picker::msInvalid; in Wm5Picker.cpp required a user-defined default constructor in the PickRecord class.

(5.0.1) LibGraphics/SceneGraph/Wm5PickRecord.h
(5.0.1) LibGraphics/SceneGraph/Wm5PickRecord.inl

July 7, 2010. Created project files for Microsoft Visual Studio 2005 and 2010. Several modifications were made to other environmental files (not listed here). New files that implement least-squares fitting for polynomials with user selected polynomial terms.

(5.2.0) LibMathematics/Approximation/Wm5ApprPolynomialFit2.h
(5.2.0) LibMathematics/Approximation/Wm5ApprPolynomialFit2.cpp
(5.2.0) LibMathematics/Approximation/Wm5ApprPolynomialFit3.h
(5.2.0) LibMathematics/Approximation/Wm5ApprPolynomialFit3.cpp
(5.2.0) LibMathematics/Approximation/Wm5ApprPolynomialFit4.h
(5.2.0) LibMathematics/Approximation/Wm5ApprPolynomialFit4.cpp
(5.0.1) LibMathematics/Wm5Mathematics.h
LibMathematics/LibMathematics_VC90.vcproj
LibMathematics/LibMathematics_VC100.vcxproj
LibMathematics/LibMathematics_VC100.vcxproj.filters

3 Updates to Version 5.2

July 17, 2010. Revisited the algorithm for computing an approximation to the inverse square root function.

Documentation/FastInverseSqrt.pdf

July 20, 2010. Added two new sections to the fast inverse square root document. These are about computing Newton iterates until the floating-point value does not change (for accurate inverse square root). One section shows the magic number for obtaining the minimum average number of iterations. Another section shows a couple of mathematically equivalent formulations whose numerical implementations are not as good as the original formulation.

Documentation/FastInverseSqrt.pdf

July 20, 2010. Fixed a typographical error in the `r20` entry for the first displayed equation in Section 2.12. It was `-sy sz1` but needed to be `-sy cz1`.

Documentation/EulerAngles.pdf

July 31, 2010. Modified the signature for Renderer::ReadColor. The old function always created a new Texture2D object. The new function allows you to pass in an already existing Texture2D object. If the input is null, the function will create the texture for you. If the input is not null and the texture does not match that of the render target, the input is recreated to match the format.

(5.0.1) LibGraphics/Renderers/Wm5Renderer.h
(5.0.1) LibGraphics/Renderers/Wm5Renderer.cpp
(5.0.1) LibGraphics/Renderers/Dx9Renderer/Wm5Dx9RenderTarget.h
(5.0.1) LibGraphics/Renderers/Dx9Renderer/Wm5Dx9RenderTarget.cpp
(5.0.1) LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLRenderTarget.h
(5.0.1) LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLRenderTarget.cpp

July 31, 2010. Added support for the OpenGL extension GL_ARB_texture_rg and fixed the mappings from luminance-alpha to the correct red-green ones.

(5.0.1) LibGraphics/Renderers/OpenGLRenderer/Wm5GlExtensions.h
(5.0.2) LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLMapping.cpp

July 31, 2010. New mathematics sample that illustrates how to compute the roots of a function using an exhaustive-evaluation method on a GPU.

(5.2.0) SampleMathematics/GpuRootFinder/GpuRootFinder.h
(5.2.0) SampleMathematics/GpuRootFinder/GpuRootFinder.cpp
(5.2.0) SampleMathematics/GpuRootFinder/RootFinderEffect.h
(5.2.0) SampleMathematics/GpuRootFinder/RootFinderEffect.cpp
SampleMathematics/GpuRootFinder/GpuRootFinder_VC80.vcproj
SampleMathematics/GpuRootFinder/GpuRootFinder_VC90.vcproj
SampleMathematics/GpuRootFinder/GpuRootFinder_VC100.vcxproj
SampleMathematics/GpuRootFinder/GpuRootFinder_VC100.vcxproj.filters
SampleMathematics/GpuRootFinder/GpuRootFinder.xcodeproj/project.pbxproj
SampleMathematics/GpuRootFinder/Shaders/RootFinder.fx
SampleMathematics/GpuRootFinder/Shaders/Compile.bat

August 5, 2010. Switched from MS Windows system mutex to CRITICAL_SECTION in the Mutex class. The system mutex is too slow and is not needed for the single-process applications of Wild Magic. In case you relied on the system mutex, you can comment out a define in Wm5Mutex.cpp to obtain the old behavior.

(5.0.1) LibCore/Threading/Wm5Mutex.cpp

August 11, 2010. I had an incorrect equation for the derivative of the quaternion function $q(t)^{f(t)}$, the quaternion function $q(t)$ raised to the power $f(t)$ and $f(t)$ is a real-valued function. I replaced the equation with a small discussion and an equation for the derivative of $q(t)$ generally.

Documentation/Quaternions.pdf

August 18, 2010. The Find() function had an early-out no-intersection test that was incorrect. The premise was that the center of the circle of intersection is of the form $K = C_0 + t * (C_1 - C_0)$, where $0 \leq t \leq 1$. However, the constraints on t are incorrect.

(5.0.1) LibMathematics/Intersection/Wm5IntrSphere3Sphere3.h
(5.0.1) LibMathematics/Intersection/Wm5IntrSphere3Sphere3.cpp

August 20, 2010. I restored the construction for the derivative of the quaternion function $q(t)^{f(t)}$ but also modified the construction of the derivative of the squad function accordingly.

Documentation/Quaternions.pdf

August 22, 2010. Added initialization for OpenGL 2.1 through 4.0.

(5.0.1) LibGraphics/Renderers/OpenGLRenderer/Wm5GlUtility.h
(5.0.2) LibGraphics/Renderers/OpenGLRenderer/Wm5GlExtensions.h
(5.0.1) LibGraphics/Renderers/OpenGLRenderer/Wm5GlExtensions.cpp

August 30, 2010. The HPlane::Normalize function had a bug in the computation of length.

(5.0.1) LibMathematics/Algebra/Wm5HPlane.cpp

August 31, 2010. The construction of the pixel shader in the constructor need its number of inputs to be set to 1.

(5.0.1) LibGraphics/LocalEffects/Wm5DefaultEffect.cpp

September 7, 2010. A new sample mathematics example that shows a general system for solving nonlinear parabolic partial differential equations in 1D, 2D, and 3D. The sample involves a PDE that models combustion (solutions become infinite in finite time, represents blow up of the material).

(5.3.0) SampleMathematics/NonlinearBlowup/NonlocalBlowup.h
(5.3.0) SampleMathematics/NonlinearBlowup/NonlocalBlowup.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/DisplacementEffect.h
(5.3.0) SampleMathematics/NonlinearBlowup/DisplacementEffect.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/OpenGLHelper.h
(5.3.0) SampleMathematics/NonlinearBlowup/OpenGLHelper.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/NonlocalSolver2.h
(5.3.0) SampleMathematics/NonlinearBlowup/NonlocalSolver2.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/FpuSupport.h
(5.3.0) SampleMathematics/NonlinearBlowup/CpuPdeSolver1.h
(5.3.0) SampleMathematics/NonlinearBlowup/CpuPdeSolver1.inl
(5.3.0) SampleMathematics/NonlinearBlowup/CpuPdeSolver2.h
(5.3.0) SampleMathematics/NonlinearBlowup/CpuPdeSolver2.inl
(5.3.0) SampleMathematics/NonlinearBlowup/CpuPdeSolver3.h
(5.3.0) SampleMathematics/NonlinearBlowup/CpuPdeSolver3.inl
(5.3.0) SampleMathematics/NonlinearBlowup/GpuPdeSolver1.h
(5.3.0) SampleMathematics/NonlinearBlowup/GpuPdeSolver1.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/GpuPdeSolver2.h
(5.3.0) SampleMathematics/NonlinearBlowup/GpuPdeSolver2.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/GpuPdeSolver3.h
(5.3.0) SampleMathematics/NonlinearBlowup/GpuPdeSolver3.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/GpuPyramid2.h
(5.3.0) SampleMathematics/NonlinearBlowup/GpuPyramid2.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/GpuAvrPyramid2.h
(5.3.0) SampleMathematics/NonlinearBlowup/GpuAvrPyramid2.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/GpuMaxPyramid2.h
(5.3.0) SampleMathematics/NonlinearBlowup/GpuMaxPyramid2.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/GpuEvaluate2.h
(5.3.0) SampleMathematics/NonlinearBlowup/GpuEvaluate2.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/Image.h
(5.3.0) SampleMathematics/NonlinearBlowup/Image.inl
(5.3.0) SampleMathematics/NonlinearBlowup/Image.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/Image1.h
(5.3.0) SampleMathematics/NonlinearBlowup/Image1.inl
(5.3.0) SampleMathematics/NonlinearBlowup/Image2.h
(5.3.0) SampleMathematics/NonlinearBlowup/Image2.inl
(5.3.0) SampleMathematics/NonlinearBlowup/Image3.h
(5.3.0) SampleMathematics/NonlinearBlowup/Image3.inl
(5.3.0) SampleMathematics/NonlinearBlowup/PixelBGRA8.h
(5.3.0) SampleMathematics/NonlinearBlowup/SaveBMP32.h
(5.3.0) SampleMathematics/NonlinearBlowup/SaveBMP32.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/RunConsole.cpp

(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/CpuLocalSolver1.h
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/CpuLocalSolver1.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/CpuLocalSolver2.h
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/CpuLocalSolver2.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/CpuLocalSolver3.h
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/CpuLocalSolver3.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/CpuNonlocalSolver1.h
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/CpuNonlocalSolver1.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/CpuNonlocalSolver2.h
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/CpuNonlocalSolver2.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/CpuNonlocalSolver3.h
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/CpuNonlocalSolver3.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/GpuLocalSolver1.h
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/GpuLocalSolver1.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/GpuLocalSolver2.h
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/GpuLocalSolver2.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/GpuLocalSolver3.h
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/GpuLocalSolver3.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/GpuNonlocalSolver1.h
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/GpuNonlocalSolver1.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/GpuNonlocalSolver2.h
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/GpuNonlocalSolver2.cpp
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/GpuNonlocalSolver3.h
(5.3.0) SampleMathematics/NonlinearBlowup/ConsoleSource/GpuNonlocalSolver3.cpp
SampleMathematics/NonlinearBlowup/NonlinearBlowup_VC80.vcproj
SampleMathematics/NonlinearBlowup/NonlinearBlowup_VC90.vcproj
SampleMathematics/NonlinearBlowup/NonlinearBlowup_VC100.vcxproj
SampleMathematics/NonlinearBlowup/NonlinearBlowup_VC100.vcxproj.filters
SampleMathematics/NonlinearBlowup/NonlinearBlowup.xcodeproj/project.pbxproj
SampleMathematics/NonlinearBlowup/Shaders/Displacement.fx
SampleMahhematics/NonlinearBlowup/Data/Domain.wmtf
SampleMahhematics/NonlinearBlowup/Data/Grid.wmtf

September 8, 2010. The condition retVal ! = 0 in Find needed to be retVal == 0. This is a porting bug from a previous version of the engine.

(5.0.1) LibMathematics/Intersection/Wm5IntrBox3Sphere3.cpp

September 8, 2010. A hack was needed to expose InitializeCriticalSectionAndSpinCount when using Microsoft Visual Studio 2005.

(5.0.1) LibCore/Threading/Wm5Mutex.cpp

September 8, 2010. Added a using statement to satisfy g++.

(5.0.2) LibMathematics/Intersection/Wm5IntrSphere3Sphere3.h

September 8, 2010. The change to the Renderer::ReadColor signature required a change to this file.

(5.0.1) SampleGraphics/CubeMaps/CubeMapEffect.cpp

September 8, 2010. The fprintf format %I64d is Microsoft specific. Fixed this to use %d (for the Macintosh and Linux).

(5.2.1) SampleMathematics/GpuRootFinder/GpuRootFinder.cpp

4 Updates to Version 5.3

September 9, 2010. The mDelta values had 1/(dimension-1) but should be 1/dimension in order that the texel centers match the pixel centers. In this application, the slight differences in the denominators do not change the results. The GL_NEAREST filtering leads to the correct texture image samples.

(5.3.1) SampleMathematics/NonlocalBlowup/GpuPdeSolver1.cpp
(5.3.1) SampleMathematics/NonlocalBlowup/GpuPdeSolver2.cpp
(5.3.1) SampleMathematics/NonlocalBlowup/GpuPdeSolver3.cpp

September 19, 2010. Changed WM5_USE_MEMORY_ASSERT_ON_PREMAIN_POSTMAIN_OPERATIONS to WM5_USE_MEMORY_ASSERT_ON_PREINIT_POSTTERM_OPERATIONS and modified the assertion strings. This makes it clear that the issue is inability to track allocations and deallocations outside a Memory::Initialize/MemoryTerminate block. Added an assertion to Memory::Initialize when it is called twice in a row without a Memory::Terminate between. Edited the assertion string in Memory::Terminate to indicate it was called twice in a row without a Memory::Initialize between. Added an msMutex.Leave() call when msMap does not exist to free the msMutex lock on the critical section.

(5.0.2) LibCore/Wm5CoreLIB.h
(5.0.1) LibCore/Memory/Wm5Memory.h
(5.0.1) LibCore/Memory/Wm5Memory.inl
(5.0.3) LibCore/Memory/Wm5Memory.cpp

September 19, 2010. Changed the signatures of Vector2 and Vector3 GetBarycentrics. Now the user must supply the epsilon used to test whether the determinant of the underlying linear system is (nearly) zero. Previously, Math::ZERO_TOLERANCE was used, but this is not a reasonable number for all applications. The default is zero. Also, the function now returns a bool that is true when the absolute value of the determinant is larger than epsilon. In this case, the barycentric coordinates have been computed. If the absolute value of the determinant is smaller or equal to epsilon, the barycentric values are set to zero (an invalid state) and the function returns false. These modifications are a change of semantics and behavior of GetBarycentrics, so if you are using these functions, you should recheck your own application logic.

(5.0.1) LibMathematics/Algebra/Wm5Vector2.h
(5.0.1) LibMathematics/Algebra/Wm5Vector2.inl
(5.0.1) LibMathematics/Algebra/Wm5Vector3.h
(5.0.1) LibMathematics/Algebra/Wm5Vector3.inl

September 19, 2010. Added two new preprocessor values, WM5_ASSERT_ON_BARYCENTRIC2_DEGENERATE and
WM5_ASSERT_ON_BARYCENTRIC3_DEGENERATE. When enabled, the Vector2 and Vector3 GetBarycentrics functions will assert when the incoming triangle or tetrahedron is degenerate. The default is that the values are not defined (no assertions are triggered).

(5.0.2) LibMathematics/Wm5MathematicsLIB.h

September 19, 2010. GetQueryType was returning an int rather than the enum Query::Type. The assertion in the constructor tested for a positive number of vertices. To support the code path in loading triangulations from files, it needs to test instead for a nonnegative number of vertices. The Load(FileIO&) function had a cut-and-paste error when porting from WM4, using Write instead of Read. Initialized the mPath array elements to zero, just to avoid having uninitialized memory.

(5.0.1) LibMathematics/ComputationalGeometry/Wm5Delaunay.h
(5.0.1) LibMathematics/ComputationalGeometry/Wm5Delaunay.cpp
(5.0.2) LibMathematics/ComputationalGeometry/Wm5Delaunay2.cpp
(5.0.3) LibMathematics/ComputationalGeometry/Wm5Delaunay3.cpp

October 1, 2010. Re-added support for dynamic libraries on all platforms. The number of affected files is approximately 800, so they are not listed here. The file-version date is 2010/10/01 on these files. The only sample application for Microsoft Windows and Macintosh that has dynamic library configurations is BillboardNodes. These illustrate how to set up such configurations in your own applications.

5 Updates to Version 5.4

October 2, 2010. Wild Magic 5.4 and previous uses DirectX SDK (March 2009) and included dxerr9.h and linked in dxerr9.lib. These are deprecated, o in order to use DirectX SDK (June 2010), some source files had to be modified and all the application projects had to be modified.

(5.0.1) LibGraphics/Renderers/Dx9Renderer/Wm5Dx9IndexBuffer.cpp
(5.0.1) LibGraphics/Renderers/Dx9Renderer/Wm5Dx9PixelShader.cpp
(5.0.1) LibGraphics/Renderers/Dx9Renderer/Wm5Dx9Renderer.cpp
(5.0.1) LibGraphics/Renderers/Dx9Renderer/Wm5Dx9RendererLIB.h
(5.0.2) LibGraphics/Renderers/Dx9Renderer/Wm5Dx9RenderTarget.cpp
(5.0.1) LibGraphics/Renderers/Dx9Renderer/Wm5Dx9Shader.cpp
(5.0.1) LibGraphics/Renderers/Dx9Renderer/Wm5Dx9Texture1D.cpp
(5.0.1) LibGraphics/Renderers/Dx9Renderer/Wm5Dx9Texture2D.cpp
(5.0.1) LibGraphics/Renderers/Dx9Renderer/Wm5Dx9Texture3D.cpp
(5.0.1) LibGraphics/Renderers/Dx9Renderer/Wm5Dx9TextureCube.cpp
(5.0.1) LibGraphics/Renderers/Dx9Renderer/Wm5Dx9VertexBuffer.cpp
(5.0.1) LibGraphics/Renderers/Dx9Renderer/Wm5Dx9VertexFormat.cpp
(5.0.1) LibGraphics/Renderers/Dx9Renderer/Wm5Dx9VertexShader.cpp
(5.2.1) Tools/GenerateProjects/TemplateVC80.cpp
(5.0.1) Tools/GenerateProjects/TemplateVC90.cpp
(5.2.1) Tools/GenerateProjects/TemplateVC100.cpp
LibGraphics/LibGraphics_VC80.vcproj
LibGraphics/LibGraphics_VC90.vcproj
LibGraphics/LibGraphics_VC100.vcxproj
Tools/BmpToWmtf/BmpToWmtf_VC80.vcproj
Tools/BmpToWmtf/BmpToWmtf_VC90.vcproj
Tools/BmpToWmtf/BmpToWmtf_VC100.vcxproj
Tools/WmfxCompiler/WmfxCompiler_VC80.vcproj
Tools/WmfxCompiler/WmfxCompiler_VC90.vcproj

Tools/WmfxCompiler/WmfxCompiler_VC100.vcxproj
Tools/WmtfViewer/WmtfViewer_VC80.vcproj
Tools/WmtfViewer/WmtfViewer_VC90.vcproj
Tools/WmtfViewer/WmtfViewer_VC100.vcxproj
Sample*/*/*_VC80.vcproj
Sample*/*/*_VC90.vcproj
Sample*/*/*_VC100.vcxproj

October 15, 2010. Added code to test for separation, containment, or intersection of ellipses and ellipsoids. This implements the algorithms in IntersectionOfEllipses.pdf and IntersectionOfEllipsoids.pdf, and uses the more detailed reduction to circle-ellipse and sphere-ellipsoid of IntersectionSweptEllipsesEllipsoids.pdf.

(5.0.2) LibMathematics/Intersection/Wm5IntrEllipse2Ellipse2.h
(5.0.2) LibMathematics/Intersection/Wm5IntrEllipse2Ellipse2.cpp
(5.5.0) LibMathematics/Intersection/Wm5IntrEllipsoid3Ellipsoid3.h
(5.5.0) LibMathematics/Intersection/Wm5IntrEllipsoid3Ellipsoid3.cpp
(5.0.2) LibMathematics/Wm5Mathematics.h
(5.5.0) SampleMathematics/IntersectionEllipsesEllipsoids/IntersectionEllipsesEllipsoids.h
(5.5.0) SampleMathematics/IntersectionEllipsesEllipsoids/IntersectionEllipsesEllipsoids.cpp
LibMathematics/LibMathematics_VC80.vcproj
LibMathematics/LibMathematics_VC90.vcproj
LibMathematics/LibMathematics_VC100.vcxproj
LibMathematics/LibMathematics_VC100.vcxproj.filters
SampleMathematics/IntersectionEllipsesEllipsoids/IntersectionEllipsesEllipsoids_VC80.vcproj
SampleMathematics/IntersectionEllipsesEllipsoids/IntersectionEllipsesEllipsoids_VC90.vcproj
SampleMathematics/IntersectionEllipsesEllipsoids/IntersectionEllipsesEllipsoids_VC100.vcxproj
SampleMathematics/IntersectionEllipsesEllipsoids/IntersectionEllipsesEllipsoids_VC100.vcxproj.filters
SampleMathematics/makefile.wm5
Documentation/IntersectionSweptEllipsesEllipsoids.pdf
WildMagic5_VC80.sln
WildMagic5_VC90.sln
WildMagic5_VC100.sln

October 15, 2010. Matrix2¡Real¿::EigenDecomposition and Matrix3¡Real¿::QLAlgorithm needed to trap and handle the case when the matrix is (nearly) diagonal.

(5.0.1) LibMathematics/Algebra/Wm5Matrix2.inl
(5.0.1) LibMathematics/Algebra/Wm5Matrix3.inl

October 16, 2010. MinSphere3::UpdateSupport4 had traps for numerical round-off errors and handled the sphere updating accordingly. The UpdateSupport2 and UpdateSupport3 functions must have similar traps. Test code missed the problems due to an incorrect assertion regarding minimum radius. The MinCircle2::UpdateSupport* functions were modified to use the same structure as that of MinSphere3.

(5.0.2) LibMathematics/Containment/Wm5ContMinSphere3.cpp
(5.0.2) LibMathematics/Containment/Wm5ContMinCircle2.cpp

October 23, 2010. The template parameters were incorrect for the {Set,Get}{Row,Column} functions.

(5.0.1) LibCore/DataTypes/Wm5Table.h
(5.0.1) LibCore/DataTypes/Wm5Table.inl

October 24, 2010. Implementation of the adaptive skeleton climbing algorithm.

(5.5.0) SampleImagics/AdaptiveSkeletonClimbing2/AdaptiveSkeletonClimbing2.h
(5.5.0) SampleImagics/AdaptiveSkeletonClimbing2/AdaptiveSkeletonClimbing2.cpp
(5.5.0) SampleImagics/AdaptiveSkeletonClimbing2/AreaMergeTree.h
(5.5.0) SampleImagics/AdaptiveSkeletonClimbing2/AreaMergeTree.cpp
(5.5.0) SampleImagics/AdaptiveSkeletonClimbing2/Climb2D.h
(5.5.0) SampleImagics/AdaptiveSkeletonClimbing2/Climb2D.cpp
(5.5.0) SampleImagics/AdaptiveSkeletonClimbing2/LinearMergeTree.h
(5.5.0) SampleImagics/AdaptiveSkeletonClimbing2/LinearMergeTree.cpp
(5.5.0) SampleImagics/AdaptiveSkeletonClimbing2/QuadTree.h
(5.5.0) SampleImagics/AdaptiveSkeletonClimbing2/QuadTree.cpp
(5.5.0) SampleImagics/AdaptiveSkeletonClimbing2/Rectangle2.h
(5.5.0) SampleImagics/AdaptiveSkeletonClimbing2/Rectangle2.cpp
SampleImagics/AdaptiveSkeletonClimbing2/AdaptiveSkeletonClimbing2_VC80.vcproj
SampleImagics/AdaptiveSkeletonClimbing2/AdaptiveSkeletonClimbing2_VC90.vcproj
SampleImagics/AdaptiveSkeletonClimbing2/AdaptiveSkeletonClimbing2_VC100.vcxproj
SampleImagics/AdaptiveSkeletonClimbing2/AdaptiveSkeletonClimbing2_VC100.vcxproj.filters
(5.5.0) SampleImagics/AdaptiveSkeletonClimbing3/AdaptiveSkeletonClimbing3.h
(5.5.0) SampleImagics/AdaptiveSkeletonClimbing3/AdaptiveSkeletonClimbing3.cpp
(5.5.0) SampleImagics/AdaptiveSkeletonClimbing3/Climb3D.h
(5.5.0) SampleImagics/AdaptiveSkeletonClimbing3/Climb3D.cpp
(5.5.0) SampleImagics/AdaptiveSkeletonClimbing3/LinearMergeTree.h
(5.5.0) SampleImagics/AdaptiveSkeletonClimbing3/LinearMergeTree.cpp
(5.5.0) SampleImagics/AdaptiveSkeletonClimbing3/MergeBox.h
(5.5.0) SampleImagics/AdaptiveSkeletonClimbing3/OctBox.h
(5.5.0) SampleImagics/AdaptiveSkeletonClimbing3/VETable.h
(5.5.0) SampleImagics/AdaptiveSkeletonClimbing3/VETable.cpp
SampleImagics/AdaptiveSkeletonClimbing3/AdaptiveSkeletonClimbing3_VC80.vcproj
SampleImagics/AdaptiveSkeletonClimbing3/AdaptiveSkeletonClimbing3_VC90.vcproj
SampleImagics/AdaptiveSkeletonClimbing3/AdaptiveSkeletonClimbing3_VC100.vcxproj
SampleImagics/AdaptiveSkeletonClimbing3/AdaptiveSkeletonClimbing3_VC100.vcxproj.filters
WildMagic5_VC80.sln
WildMagic5_VC90.sln
WildMagic5_VC100.sln

October 24, 2010. A pair of new documents. The first shows how to define a Bicubic Bézier tensor-product surface patch (4×4 control points) that exactly interpolate a given 4×4 set of samples. The second shows how minimizing bending energy and degree elevation are related when trying to construct the control points of a Bzier curve based on partial information that determines only a strict subset of the control points. The remaining are computed based on minimizing the curve's bending energy.

Documentation/BicubicExactInterpolation.pdf
Documentation/BezierCurveBendingElevation.pdf

November 2, 2010. New code and sample to test for intersection of stationary triangle and cylinder.

(5.0.3) LibMathematics/Wm5Mathematics.h
(5.5.0) LibMathematics/Intersection/Wm5IntrTriangle3Cylinder3.h
(5.5.0) LibMathematics/Intersection/Wm5IntrTriangle3Cylinder3.cpp

(5.5.0) SampleMathematics/IntersectTriangleCylinder/IntersectTriangleCylinder.h
(5.5.0) SampleMathematics/IntersectTriangleCylinder/IntersectTriangleCylinder.cpp
SampleMathematics/IntersectTriangleCylinder/IntersectTriangleCylinder_VC80.vcproj
SampleMathematics/IntersectTriangleCylinder/IntersectTriangleCylinder_VC90.vcproj
SampleMathematics/IntersectTriangleCylinder/IntersectTriangleCylinder_VC100.vcxproj
SampleMathematics/IntersectTriangleCylinder/IntersectTriangleCylinder_VC100.vcxproj.filters
SampleMathematics/IntersectTriangleCylinder/IntersectTriangleCylinder.xcodeproj/project.pbxproj
SampleMathematics/makefile.wm5
WildMagic5_VC80.sln
WildMagic5_VC90.sln
WildMagic5_VC100.sln

November 20, 2010. Fixed an error in the equations for the area bounded by an elliptical arc and a line segment. The theta angles in the document refer to polar coordinates, but I had re-used theta for a parameterization of the ellipse. The parameterizations now use angles phi and psi to avoid confusion.

Documentation/AreaIntersectingEllipses.pdf

December 5, 2010. Added -lpthread to LIBS line to satisfy Ubuntu Linux.

SampleGraphics/makeapp.wm5
SampleImagics/makeapp.wm5
SampleMathematics/makeapp.wm5
SamplePhysics/makeapp.wm5

December 11, 2010. The SetKnot and GetKnot functions had index-range checks whose upper bounds were off by one. GetKnot now returns knot values for uniform splines. The comments in the header file were expanded to contain more information about the knots.

(5.0.2) LibMathematics/CurvesSurfacesVolumes/Wm5BSplineBasis.h
(5.0.2) LibMathematics/CurvesSurfacesVolumes/Wm5BSplineBasis.cpp

December 11, 2010. The Resize functions called the back-end set-viewport functions, but failed to set the mWidth and mHeight data members.

(5.0.2) LibGraphics/Renderers/Dx9Renderer/Wm5DXRenderer.cpp
(5.0.1) LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLRenderer.cpp

December 24, 2010. The Load/Save functions has a cut-and-paste error. mSVertices needed to be loaded/saved instead of mVertices being processed a second time.

(5.0.4) LibMathematics/ComputationalGeometry/Wm5Delaunay3.cpp

January 18, 2011. The code to compute the sphere passing through four points was ill conditioned for moderate sized data, even when using double-precision arithmetic. The linear system is of the form $A^{\mathrm{T}}A\boldsymbol{u} = \boldsymbol{r}$. The old code computed $A^{\mathrm{T}}A$ and inverted to solve for $\boldsymbol{u}$. The new code solves the system in two steps. Matrix A is inverted and $A\boldsymbol{u} = A^{-\mathrm{T}}\boldsymbol{r} = \boldsymbol{s}$ is computed (the right-hand side is computed); then $\boldsymbol{u} = A^{-1}\boldsymbol{s}$ is computed.

(5.0.3) LibMathematics/Containment/Wm5ContMinSphere3.cpp

January 23, 2011. The keyboard event handling was incorrectly structured. The key-down and key-up handlers were both handling key-repeat events, which caused problems with Wild Magic's camera-moving code. Now only the key-down handler processes the key-repeat events. Also, the key-modifier-changed events were being handled by key-up and key-down, but without regard to the last-known state of the modifier keys. A new event handler was added to track the state and to call Wild Magic's OnSpecialKeyUp or OnSpecialKeyDown as needed. This fixes the BlendAnimation bug on the Macintosh, whereby the SHIFT + LARROW keys make the character transition from walk to run but releasing the SHIFT key did not make the character transition back to a walk.

(5.0.2) LibApplications/AglApplication/Wm5AglApplication.cpp

February 7, 2011. Fixed a porting error in Classify when converting WM4 to WM5.

(5.0.2) LibMathematics/Intersection/Wm5IntrLine2Line2.cpp

February 8, 2011. Apparently, most flavors of LINUX conditionally define __linux__, but we have used __LINUX__. Added code to define __LINUX__ when __linux__ is defined.

(5.0.4) LibCore/Wm5CoreLIB.h

February 9, 2011. Fixed a typographical error on page 5, the last displayed equation. It had `lambda1 = c2/3 - rho` but needed to be `lambda1 = c2/3 - 2*rho`.

Documentation/EigenSymmetric3x3.pdf

February 14, 2011. Added assertions when the visual or instance input pointers are null. If either is null, the function returns without attempting to draw (graceful handling in Release builds). The typical reason for a null instance is that the Visual object does not have a VisualEffectInstance attached to it.

(5.0.2) LibGraphics/Renderers/Wm5Renderer.cpp

March 9, 2011. Fixed a porting error in MinimizeN<Real>::ComputeDomain when converting WM4 to WM5. Added many comments to the header files to describe the algorithms at a high level.

(5.0.2) LibMathematics/NumericalAnalysis/Wm5Minimize1.h
(5.0.2) LibMathematics/NumericalAnalysis/Wm5MinimizeN.h
(5.0.2) LibMathematics/NumericalAnalysis/Wm5MinimizeN.cpp

March 13, 2011. The SquaredLength and Dot functions summed one too many elements.

(5.0.1) LibMathematics/Algebra/Wm5GVector.inl

March 27, 2011. When the last iteration of a search produces the current minimum of the search, the current minimum value and location were not updated. The bookkeeping is now fixed and the code updates the current minimum value and location after each function evaluation.

(5.0.2) LibMathematics/NumericalAnalysis/Wm5Minimize1.cpp

March 27, 2011. Changed from TYPE operator[](int) const to const TYPE& operator[](int) const to be consistent with how STL handles operator[]. Changed from TYPE operator()(int,int) const to const TYPE& operator()(int,int) const to be consistent with how STL handles operator().

(5.0.2) LibCore/DataTypes/Wm5Table.h
(5.0.2) LibCore/DataTypes/Wm5Table.inl
(5.0.2) LibCore/DataTypes/Wm5Tuple.h
(5.0.1) LibCore/DataTypes/Wm5Tuple.inl
(5.0.1) LibMathematics/Algebra/Wm5GMatrix.h
(5.0.1) LibMathematics/Algebra/Wm5GMatrix.inl
(5.0.1) LibMathematics/Algebra/Wm5GVector.h
(5.0.2) LibMathematics/Algebra/Wm5GVector.inl
(5.0.1) LibMathematics/Algebra/Wm5HMatrix.h
(5.0.1) LibMathematics/Algebra/Wm5HMatrix.inl
(5.0.2) LibMathematics/Algebra/Wm5HPlane.h
(5.0.1) LibMathematics/Algebra/Wm5HPlane.inl
(5.0.2) LibMathematics/Algebra/Wm5HPoint.h
(5.0.1) LibMathematics/Algebra/Wm5HPoint.inl
(5.0.2) LibMathematics/Algebra/Wm5HQuaternion.h
(5.0.1) LibMathematics/Algebra/Wm5HQuaternion.inl
(5.0.1) LibMathematics/Algebra/Wm5Polynomial1.h
(5.0.1) LibMathematics/Algebra/Wm5Polynomial1.inl
(5.0.2) LibMathematics/Algebra/Wm5Quaternion.h
(5.0.1) LibMathematics/Algebra/Wm5Quaternion.inl
(5.0.1) LibMathematics/Rational/Wm5IVector.h
(5.0.1) LibMathematics/Rational/Wm5IVector.inl
(5.0.1) LibMathematics/Rational/Wm5RVector.h
(5.0.1) LibMathematics/Rational/Wm5RVector.inl

March 27, 2011. Added explicit typecasts for APoint conversion to Vector3f to satisfy g++.

(5.5.1) SampleMathematics/IntersectTriangleCylinder/IntersectTriangleCylinder.cpp

March 27, 2011. Added missing files to the projects. Rearranged the order of initialization in the compiler to satisfy g++.

SampleImagics/AdaptiveSkeletonClimbing2/AdaptiveSkeletonClimbing2.xcodeproj/project.pbxproj
SampleImagics/AdaptiveSkeletonClimbing3/AdaptiveSkeletonClimbing3.xcodeproj/project.pbxproj
(5.5.1) SampleImagics/AdaptiveSkeletonClimbing3/Climb3D.cpp

March 27, 2011. Added Wm5IntrTriangle3Cylinder3.{h,cpp} and Wm5IntrEllipsoid3Ellipsoid3.{h,cpp} to the project.

LibMathematics/LibMathematics.xcodeproj/project.pbxproj

March 27, 2011. Removed the explicit lines to search /usr/include and /usr/lib. The defaults will be used automatically, and not specifying /usr/lib ensures that 64-bit systems will compile the distribution correctly.

SampleGraphics/makeapp.wm5
SampleImagics/makeapp.wm5
SampleMathematics/makeapp.wm5
SampleMathematics/makeapp.wm5
SamplePhysics/makeapp.wm5

6 Updates to Version 5.5

April 16, 2011. The GetClassification function needed logic to handle the special case when the ellipsoid centers are equal.

(5.5.1) LibMathematics/Intersection/Wm5IntrEllipsoid3Ellipsoid3.cpp

April 16, 2011. The member mClosestPoint0 stores the closest triangle point and the member mClosestPoint1 stores the closest rectangle point. The last code block of GetSquared() had them reversed.

(5.0.2) LibMathematics/Distance/Wm5DistTriangle3Rectangle3.cpp

May 1, 2011. The LoadWMTF functions need to delete the texture returned from the base-class LoadWMTF function if the texture type does not match.

(5.0.1) LibGraphics/Resources/Wm5Texture.cpp
(5.0.1) LibGraphics/Resources/Wm5Texture1D.cpp
(5.0.1) LibGraphics/Resources/Wm5Texture2D.cpp
(5.0.1) LibGraphics/Resources/Wm5Texture3D.cpp
(5.0.1) LibGraphics/Resources/Wm5TextureCube.cpp

May 3, 2011. Disable the min and max macros that are sucked in by the inclusion of windows.h. These conflict with std::numeric_limits<type>::max().

(5.0.2) LibGraphics/Renderers/Dx9Renderer/Wm5Dx9RendererLIB.h
(5.0.1) LibGraphics/Renderers/WglRenderer/Wm5WglExtensions.h

May 3, 2011. Added an include of <limits> to access the std::numeric_limits class.

(5.0.5) LibCore/Wm5LibCoreLIB.h

May 7, 2011. GetStreamingSize needed to use input.length() rather than strlen(input.c_str()) in case the string has null characters embedded in it.

(5.0.1) LibCore/ObjectSystems/Wm5Stream.cpp

May 19, 2011. Fixed some errors. In the pseudocode on page 12, the code

```
Real invMax = 1/M[maxRow][maxCol];
```

needs to be

```
Real invMax = 1/M[0][maxCol];
```

because the rows were swapped to ensure the maximum value occurs in row 0. Similarly, the pseudocode on page 13 has

```
Real invMax = 1/M[maxRow][maxCol];
```

needs to be

```
Real invMax = 1/M[1][maxCol];
```

On page 17, the pseudocode case `rank1 == 2` grabs rows from M0

```
row0 = M0.GetRow(0);
row1 = M0.GetRow(1);
```

but they need to come from M1

```
row0 = M1.GetRow(0);
row1 = M1.GetRow(1);
```

Documentation/EigenSymmetric3x3.pdf

May 22, 2011. Revised the thin-plates spline PDF to include an informal mathematical discussion of the construction. The mapping of data to unit square/cube was hard-coded. Now you can select via a constructor input whether this happens. Member accessors were added to get the coefficients computed for the spline and to get the smoothing parameter. A function was added to compute the minimum functional value. The kernel for the 3D spline was coded as $r^2 \log(r^2)$, but it needed to be the function $-|r|$. Added a sample application to show how to use the code. This used to be embedded in the PDF.

Documentation/ThinPlateSplines.pdf
(5.0.2) LibMathematics/Interpolation/Wm5IntpThinPlateSpline2.h
(5.0.2) LibMathematics/Interpolation/Wm5IntpThinPlateSpline2.cpp
(5.0.2) LibMathematics/Interpolation/Wm5IntpThinPlateSpline3.h
(5.0.2) LibMathematics/Interpolation/Wm5IntpThinPlateSpline3.cpp
(5.6.0) SampleMathematics/ThinPlateSplines/ThinPlateSplines.h
(5.6.0) SampleMathematics/ThinPlateSplines/ThinPlateSplines.cpp

May 23, 2011. The pseudocode in Section 2.6 for the case `r20 = -1` was missing a minus sign on the `atan2` term.

Documentation/EulerAngles.pdf

June 18, 2011. New document that combines DistancePointToEllipse2.pdf and DistancePointToEllipsoid.pdf. The discussion is greatly expanded to show how to compute the distance and closest point robustly. It also shows why the polynomial-root-finding approach mentioned in Graphics Gems IV can have numerical problems. The generalization is discussed to distance between a point and a hyperellipsoid in any dimension. Replaced the point-ellipse and point-ellipsoid distance code of Wild Magic with code for the new algorithm. Added a sample application to illustrate.

Documentation/DistancePointEllipseEllipsoid.pdf
(5.0.2) LibMathematics/Distance/Wm5DistPoint2Ellipse2.h
(5.0.2) LibMathematics/Distance/Wm5DistPoint2Ellipse2.cpp
(5.0.2) LibMathematics/Distance/Wm5DistPoint3Ellipsoid3.h
(5.0.2) LibMathematics/Distance/Wm5DistPoint3Ellipsoid3.cpp
(5.6.0) SampleMathematics/DistancePointEllipseEllipsoid/DistancePointEllipseEllipsoid.h
(5.6.0) SampleMathematics/DistancePointEllipseEllipsoid/DistancePointEllipseEllipsoid.cpp
SampleMathematics/DistancePointEllipseEllipsoid/DistancePointEllipseEllipsoid_VC80.vcproj
SampleMathematics/DistancePointEllipseEllipsoid/DistancePointEllipseEllipsoid_VC90.vcproj
SampleMathematics/DistancePointEllipseEllipsoid/DistancePointEllipseEllipsoid_VC100.vcxproj
SampleMathematics/DistancePointEllipseEllipsoid/DistancePointEllipseEllipsoid_VC100.vcxproj.filters
SampleMathematics/DistancePointEllipseEllipsoid/DistancePointEllipseEllipsoid.xcodeproj/project.pbxproj
WildMagic5_VC80.sln
WildMagic5_VC90.sln
WildMagic5_VC100.sln

June 19, 2011. If the incoming line is the empty string, the program needed to trap that and continue (presumably we are at end-of-file). Without this trap in MSVS 2010, the program crashes when line[0] is accessed. However, the program does not crash in MSVS 2008 when accessing line[0]. We had built all our sample *.wmfx files using WmfxCompiler built with MSVS 2008, which is why we did not see this problem previously.

(5.0.2) Tools/WmfxCompiler/FxCompiler.cpp

June 19, 2011. GetPickRay was originally created to build a ray for a perspective camera. It needed to be updated to support also an orthographic camera.

(5.0.3) LibGraphics/Renderers/Wm5Renderer.cpp

June 19, 2011. After making some changes previously in ExactSphere4 to deal with floating-point issues, there are still some issues. The M.Inverse() call uses a default tolerance of 0 when testing the determinant of M to see whether it is invertible. This causes failure in some data sets, so now the call is M.Inverse(mEpsilon). A TODO is now in the code:

```
// TODO:  With mEpsilon == 0.0, there are data sets for which this
// algorithm fails.  A small positive mEpsilon appears to help, but
// this is a classic problem of computational geometry--determining
// the correct sign of a determinant when using floating-point
// arithmetic.  One of the goals of the Malleable Mathematics
// Library is to eliminate such problems (using arbitrary precision
// arithmetic or a filtered predicate).
```

(5.0.4) LibMathematics/Containment/Wm5ContMinSphere3.cpp

June 19, 2011. Initialized sqrDistance to avoid compiler warning C4701 in MSVS 2008. The compiler is unable to parse the logic to determine that sqrDistance is assigned a value in all cases. MSVS 2010 appears to have eliminated warning C4701; it does not show up in the index for Microsoft Help Viewere 1.1 - Catalog VS_100_EN-US.

(5.0.3) LibMathematics/Distance/Wm5DistPoint3Ellipsoid3.cpp

June 21, 2011. Added an alpha channel to ColorRGB. For now, the class name is not changed to avoid having to change all the 2D sample application code. The name change will occur closer to shipping of WM5.6. The mScreenBuffer is now an array of 32-bits-per-pixel data, which allows better performance for pixel copies to the back buffer. The ClearScreen() function was converting float-color to byte-color each pixel when it only needed to do it once. The DX9 and OpenGL renderers Draw(unsigned char*) functions were modified to copy RGBA data (old code copied RGB data). The DX9 copy used D3DXLoadSurfaceFromMemory which is really slow. Now the incoming screen buffer is copied to an IDirect3DSurface9, and then the back buffer is updated from this surface. The frame rate is now 2 to 3 times larger for DX9 2D applications. The OpenGL pixel copy needed only a format change from GL_BGR to GL_BGRA; the frame rate increased about 20 percent.

(5.0.2) LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLRenderer.cpp
(5.0.3) LibGraphics/Renderers/Dx9Renderer/Wm5Dx9Renderer.cpp
(5.0.1) LibApplications/Wm5WindowApplication2.h
(5.0.1) LibApplications/Wm5WindowApplication2.cpp

June 27, 2011. Changed the signature void Draw (const unsigned char* screenBuffer) to void Draw (const unsigned char* screenBuffer, bool reflectY = false). The renderers now directly support reflection in y. The WinApplication2 class no longer has the inefficient mechanism for reflecting the screen buffer to a temporary buffer followed by passing the temporary buffer to the Draw call.

(5.0.3) LibGraphics/Renderers/Wm5Renderer.h
(5.0.3) LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLRenderer.cpp
(5.0.4) LibGraphics/Renderers/Dx9Renderer/Wm5Dx9Renderer.cpp
(5.0.2) LibApplications/Wm5WindowApplication2.h
(5.0.2) LibApplications/Wm5WindowApplication2.cpp

June 29, 2011. Modified the template to contain include and library paths in the DX9 configurations. The paths use the DXSDK_DIR environment variable. This avoids the problem on x64 machines whereby Program Files (x86) is the path to the DX9 files, but the global paths in Visual Studio 2010 use instead Program Files.

(5.2.2) Tools/GenerateProjects/TemplateVC100.cpp

July 9, 2011. A patch was added some time ago to compute world bounds when dummy nodes (Node with no children) existed in the graph. The patch did not handle the case when the first child of a node is a dummy node. Also, when a node has its children removed and then Update() is called at this node, the world bound is not reset to invalid (zero radius). Fixed UpdateWorldBound() so that the world bound is first made invalid, and now GrowToContain(...) properly handles valid or invalid bounds, whether the incoming child bound or the current world bound.

(5.0.1) LibGraphics/SceneGraph/Wm5Node.cpp
(5.0.2) LibGraphics/DataTypes/Wm5Bound.cpp

July 9, 2011. The deferred creation of textures was interfering with the enabling and disabling of textures during the first Renderer::Draw call. The binding of new textures to a target during creation needed to retrieve previously bound textures to the target so that after creation, the previously bound textures could be rebound. The OpenGLTexture* classes now have a new member, mPreviousTexture, that is used to store previously bound textures before the Enable call and rebind them at the end of the Disable call. This is for persistence in the one case where the rebind does not occur within the same member function call.

(5.0.1) LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLTexture1D.h
(5.0.1) LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLTexture1D.cpp
(5.0.1) LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLTexture2D.h
(5.0.1) LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLTexture2D.cpp
(5.0.1) LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLTexture3D.h
(5.0.1) LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLTexture4D.cpp
(5.0.1) LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLTextureCube.h
(5.0.1) LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLTextureCube.cpp
(5.0.2) LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLRenderTarget.cpp
(5.0.1) LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLMapping.h
(5.0.3) LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLMapping.cpp

July 9, 2011. Fixed a bug due to the removal of WindowApplication::Index.

(5.0.3) SampleImagics/ImViewer/ImViewer.cpp

July 9, 2011. Added an implementation of Brent's Method for root finding in one dimension.

(5.6.0) LibMathematics/NumericalAnalysis/Wm5BrentsMethod.h
(5.6.0) LibMathematics/NumericalAnalysis/Wm5BrentsMethod.cpp
(5.0.4) LibMathematics/Wm5Mathematics.h

July 9, 2011. The GMatrix class had a subtle problem with the mData member. During linear system solving or matrix inversion, row swaps were performed by swapping pointers of mEntry. If a row swap occurs, then any access of GMatrix elements via mData, such as the implicit operator conversions to Real* or const Real* is not safe, because mData is never re-ordered even though mEntry is, leading to inconsistent access (via mData or via mEntry). The memcpy calls in the comparison predicates are then also not safe. The only reason for the pointer swaps was to avoid copying row data when GMatrix has a large number of columns. However, for the short term this small gain in performance is less important than the access-correctness of the matrix elements. The class was rewritten to eliminate mData, allocating mEntry as a 2D array with contiguous elements stored in row-major order in mEntry[0]. Some of the class member accessors were renamed to be consistent with other classes in the engine; for example, GetRows() is now GetNumRows(). The other classes that were modified were using only these accessors. GMatrix had a GetInverse member, but this was a duplicate of the code in the LinearSystem class. The GMatrix version was removed. The LinearSystem code calls GMatrix::SwapRows(...), which hides how the swap is performed, so there should be no unexpected side effects in application code.

(5.0.2) LibMathematics/Algebra/Wm5GMatrix.h
(5.0.2) LibMathematics/Algebra/Wm5GMatrix.inl

(5.0.2) LibMathematics/NumericalAnalysis/Wm5EigenDecomposition.cpp
(5.0.2) LibMathematics/NumericalAnalysis/Wm5LinearSystem.cpp
(5.0.2) LibMathematics/NumericalAnalysis/Wm5OdeImplicitEuler.cpp
(5.0.2) LibMathematics/NumericalAnalysis/Wm5PolynomialRoots.cpp
(5.0.2) LibMathematics/NumericalAnalysis/Wm5SingularValueDecomposition.cpp
(5.0.4) LibMathematics/Wm5MathematicsLIB.h

July 23, 2011. The llvm-gcc-4.2 compiler on the Macintosh warned about && within || in a compound Boolean expression. In fact, that expression was incorrectly formed and required parentheses.

(5.0.2) LibMathematics/Intersection/Wm5IntrBox3Sphere3.cpp

July 23, 2011. The template static members must be explicitly instantiated before the template class is explicitly instantiated. This avoids compiler errors generated by llvm-gcc-4.2 on the Macintosh.

(5.0.2) LibMathematics/Interpolation/Wm5IntpBicubic2.cpp
(5.0.3) LibMathematics/Interpolation/Wm5IntpBilinear2.cpp
(5.0.2) LibMathematics/Interpolation/Wm5IntpTrilinear3.cpp
(5.0.2) LibMathematics/Interpolation/Wm5IntpTricubic3.cpp
(5.0.2) LibMathematics/Miscellaneous/Wm5GridGraph2.cpp
(5.0.2) LibMathematics/NumericalAnalysis/Wm5NoniterativeEigen3x3.cpp
(5.0.2) LibMathematics/NumericalAnalysis/Wm5PolynomialRootsR.cpp

July 23, 2011. Added a using statement for base-class members to avoid compiler errors generated by llvm-gcc-4.2 on the Macintosh.

(5.0.2) LibMathematics/CurvesSurfacesVolumes/Wm5CubicPolynomialCurve2.h
(5.0.2) LibMathematics/CurvesSurfacesVolumes/Wm5CubicPolynomialCurve3.h
(5.0.2) LibMathematics/CurvesSurfacesVolumes/Wm5NaturalSpline2.h
(5.0.2) LibMathematics/CurvesSurfacesVolumes/Wm5NaturalSpline3.h
(5.0.2) LibMathematics/CurvesSurfacesVolumes/Wm5SingleCurve2.h
(5.0.2) LibMathematics/CurvesSurfacesVolumes/Wm5SingleCurve3.h
(5.0.2) LibMathematics/CurvesSurfacesVolumes/Wm5TCBSpline2.h
(5.0.2) LibMathematics/CurvesSurfacesVolumes/Wm5TCBSpline3.h
(5.0.2) LibMathematics/Interpolation/Wm5IntpAkimaNonuniform1.h
(5.0.2) LibMathematics/Interpolation/Wm5IntpAkimaUniform1.h
(5.0.2) LibMathematics/Interpolation/Wm5IntpBSplineUniform1.h
(5.0.2) LibMathematics/Interpolation/Wm5IntpBSplineUniform2.h
(5.0.2) LibMathematics/Interpolation/Wm5IntpBSplineUniform3.h
(5.0.2) LibMathematics/Interpolation/Wm5IntpBSplineUniform4.h
(5.0.2) LibMathematics/Interpolation/Wm5IntpBSplineUniformN.h
(5.0.2) LibMathematics/NumericalAnalysis/Wm5OdeEuler.h
(5.0.2) LibMathematics/NumericalAnalysis/Wm5OdeImplicitEuler.h
(5.0.2) LibMathematics/NumericalAnalysis/Wm5OdeMidpoint.h
(5.0.2) LibMathematics/NumericalAnalysis/Wm5OdeRungeKutta4.h
(5.0.1) LibMathematics/Query/Wm5Query2Filtered.h
(5.0.1) LibMathematics/Query/Wm5Query3Filtered.h

July 23, 2011. The memory leak detection system was enabled in Debug configurations. For those users not using the Wild Magic application layer, this required calling Memory::Initialize and Memory::Terminate in

main or WinMain. If the initialization is not called, assertions are triggered when allocating memory (these are benign). To make it easier for new users, the memory leak detection system is disabled in all configurations. If you want this system, you must expose the #define WM5_USE_MEMORY in Wm5CoreLIB.h and rebuild all the libraries.

(5.0.6) LibCore/Wm5CoreLIB.h

July 24, 2011. MSVS 2010 did not compile all members, but llvm-gcc-4.2 did and complained about several operator comparisons accessing undefined variables.

(5.0.3) LibMathematics/Algebra/Wm5GMatrix.inl

July 27, 2011. Removed unused variable n from Test0.

(5.5.1) SampleImagics/AdaptiveSkeletonClimbing2/AdaptiveSkeletonClimbing2.cpp

July 27, 2011. Added WM5_UNUSED macro to avoid warning about unused variable in release builds.

(5.5.1) SampleImagics/AdaptiveSkeletonClimbing3/VETable.cpp
(5.5.1) SampleMathematics/IntersectionEllipsesEllipsoids/IntersectionEllipsesEllipsoids.cpp
(5.2.1) SamplePhysics/CollisionsMovingSpheres/CollisionsMovingSpheres.cpp

July 27, 2011. The order of the inputs to new2 were reversed from what they should have been.

(5.0.4) LibMathematics/Algebra/Wm5GMatrix.inl

July 28, 2011. Modified the delta-time from 1/6 to 0.01 to slow down the simulation to a realistic rate.

(5.0.1) SamplePhysics/DoublePendulum/DoublePendulum.cpp

July 30, 2011. Added generation of project.pbxproj in Xcode 3.2 format.

(5.0.1) Tools/GenerateProjects/GenerateProjects.cpp
(5.6.0) Tools/GenerateProjects/TemplateXC32.h
(5.6.0) Tools/GenerateProjects/TemplateXC32.cpp

July 30, 2011. Modified the preprocessor statements to enable display-list text only with WIN32 but not using GLUT.

(5.0.3) LibGraphics/Wm5GraphicsLIB.h

July 30, 2011. Added a hack to work around getenv returning NULL on Mac OS X 10.7 [Lion] for any specified environment variable.

(5.0.2) LibApplications/Wm5Application.cpp

7 Updates to Version 5.6

October 22, 2011. A publication that shows a novel approach to approximating SLERP as a polynomial of two variables.

> David Eberly (2011): A Fast and Accurate Algorithm for Computing SLERP
> Journal of Graphics, GPU, and Game Tools, 15:3, 161-176
>
> Abstract. The SLERP of quaternions is a common operation in keyframe animation. The operation can be a significant bottleneck in an animation-heavy application. The standard implementation of SLERP for an FPU typically involves trigonometric function evaluations, divisions, and branching. Commonly available SIMD implementations will modify the standard implementation and obtain a moderate speed-up. This paper provides a novel approach to computing SLERP, using only multiplications and additions. The algorithm is based on ideas from Chebyshev polynomials, power series solutions for differential equations, and error balancing using the Chebyshev Equioscillation Theorem and the associated Remez Algorithm. Implementations are provided for the FPU and SIMD. When SLERPing a pair of quaternions in parallel using four time samples, performance measurements show a 10-fold acceleration per SLERP over the standard SLERP implementation on an FPU. Source code is available online.

October 22, 2011. Several problems when computing first contact between moving triangles.

1. FindContactSet was using index i when it should have been using cfg*.mIndex[i], as specified in the documentation on the method of separating axes.
2. GetEdgeEdgeIntersection asserts that $0 \leq s \leq 1$ but has no epsilon tolerance. Clamping is now used, but assertions are generated if s is not sufficiently close to $[0, 1]$.
3. GetEdgeFaceIntersection computes the intersection of two intervals of projection. Numerical round-off errors can cause the calculator to fail when the intervals theoretically intersect at an endpoint. When the calculator fails, rather than assert the code now uses a slow but robust 3D distance calculator to find the intersection point (which must exist).

> (5.0.2) LibMathematics/Intersection/Wm5IntrTriangle3Triangle3.cpp

October 22, 2011. Item 4 that contains equations (13) and (14) needed *minor* and *major* to to be swapped.

> Documentation/InformationAboutEllipses.pdf

October 22, 2011. New document that describes how to compute (numerically) the curve of intersection (or single point when tangent) of two infinite cylinders. The algorithm may be easily modified to handle finite cylinders.

> Documentation/IntersectionInfiniteCylinders.pdf

October 22, 2011. Fixed a minor typographical error in section 2 about eliminating extraneous solutions.

Documentation/IntersectionOfEllipses.pdf

October 22, 2011. Reorganized the project folders to reflect the directory structure. Renamed the Biped Files folder to Bones. Added Animations and moved the Idle, Walk, and Run folders to it. Added Skins and added the corresponding data files to that project folder. Created 4 new documents that describe the formats of the *.raw files.

SampleGraphics/BlendedAnimations/BlendedAnimations_VC80.vcproj
SampleGraphics/BlendedAnimations/BlendedAnimations_VC90.vcproj
SampleGraphics/BlendedAnimations/BlendedAnimations_VC100.vcxproj
SampleGraphics/BlendedAnimations/BlendedAnimations_VC100.vcxproj.filters
SampleGraphics/BlendedAnimations/Data/FormatMeshRaw.txt
SampleGraphics/BlendedAnimations/Data/FormatNodeRaw.txt
SampleGraphics/BlendedAnimations/Data/FormatSkinCtrlRaw.txt
SampleGraphics/BlendedAnimations/Data/FormatXfrmCtrlRaw.txt

October 22, 2011. Added SetProjectionMatrix that has inputs of four vertices of a convex quadrilateral points in camera coordinates, a near-plane extrusion value, and a far-plane extrusion value. The plane of the quadrilateral is the view plane and the quadrilateral is the viewport. The quadrilateral is extruded by the specified values to form a cuboidal view volume. This illustrates the concepts in the perspective mappings document posted on August 17, 2011.

(5.0.3) LibGraphics/SceneGraph/Wm5Camera.h
(5.0.1) LibGraphics/SceneGraph/Wm5Camera.inl
(5.0.1) LibGraphics/SceneGraph/Wm5Camera.cpp

October 22, 2011. A new document that describes the *.im format used in the image processing library.

Documentation/ImFileFormat.txt

October 26, 2011. The Find() query normalized the difference of sphere centers before using it to compute the intersection circle center. The normalization has to occur after computing the circle center. This error was introduced in the conversion from WM4 to WM5.

(5.0.3) LibMathematics/Intersection/Wm5IntrSphere3Sphere3.cpp

8 Updates to Version 5.7

November 12, 2011. The project files did not have the correct source file references.

SampleImagics/AdaptiveSkeletonClimbing2/AdaptiveSkeletonClimbing2_VC80.vcproj
SampleImagics/AdaptiveSkeletonClimbing2/AdaptiveSkeletonClimbing2_VC90.vcproj
SampleImagics/AdaptiveSkeletonClimbing3/AdaptiveSkeletonClimbing2_VC80.vcproj
SampleImagics/AdaptiveSkeletonClimbing3/AdaptiveSkeletonClimbing2_VC90.vcproj

February 19, 2012. Extra logic was needed for when the closest point on the ray $\boldsymbol{M} + t(1, 0)$ is a vertex shared by newly added edges (problem occurred when the polygon had two or more holes).

(5.0.2) LibMathematics/ComputationalGeometry/Wm5TriangulateEC.cpp

March 9, 2012. The GMatrix negation operator was using the incorrect array pointer. Tuple, Table, GVector, and GMatrix were using memcmp for the comparison predicates. Unfortunately, this circumvents the semantics of floating-point comparisons. Switched to an algorithm that adheres to those semantics.

(5.0.3) LibCore/DataTypes/Wm5Tuple.h
(5.0.2) LibCore/DataTypes/Wm5Tuple.inl
(5.0.3) LibCore/DataTypes/Wm5Table.h
(5.0.3) LibCore/DataTypes/Wm5Table.inl
(5.0.5) LibMathematics/Wm5MathematicsLIB.h
(5.0.3) LibMathematics/Algebra/Wm5GVector.inl
(5.0.5) LibMathematics/Algebra/Wm5GMatrix.inl

March 9, 2012. The allocation of mImage should use the number of texels rather than the number of bytes.

(5.0.2) Tools/WmtfViewer/WmtfViewer.cpp

March 9, 2012. The currentSS array needed the texture unit as index, not the texture unit plus base (the base is positive for vertex textures).

(5.0.2) LibGraphics/Renderers/Dx9Renderer/Wm5Dx9Shader.cpp

March 9, 2012. Fixed the title on the document.

DistanceEllipse2Ellipse2.pdf

March 17, 2012. Updated the defines to detect that the Microsoft compiler version is MSVS 2010 (or later). Added conditional compilation so that MSVS 2010 (or later) includes <stdint.h> rather than defining the symbols that the standard integer package now supports. This avoids name clashes with third-party packages that use the standard integer package.

(5.0.7) LibCore/Wm5CoreLIB.h

May 9, 2012. The Find(...) function for moving segment/triangle intersection test was missing an assignment to the axis direction in the nonparallel case.

(5.0.2) LibMathematics/Intersection/Wm5IntrSegment3Triangle3.cpp

June 12, 2012. Added explicit path information for DX9 using environment variable DXSDK_DIR.

LibApplications/LibApplications_VC100.vcxproj
LibGraphics/LibGraphics_VC100.vcxproj

June 24, 2012. Fixed the linker errors for GLUT DLL configurations. Disabled the warnings (via a pragma) about unused GLUT functions.

(5.6.1) LibGraphics/Renderers/GlutRenderer/Wm5GlutRendererInput.h
LibGraphics/LibGraphics_VC100.vcxproj

June 24, 2012. The Find call has blocks with if (calc.Find(...)) { } that compute the point of intersection. Each block should return true, indicating that an intersection was found; the old code did not have these return values. The FindTriangleSphereCoplanarIntersection function has a block with the correct return.

(5.0.2) LibMathematics/Intersection/Wm5IntrTriangle3Sphere3.cpp

June 24, 2012. When specifying BT_CLAMPED, the NaturalSpline{2,3} classes automatically computed the endpoint derivatives to be P[1]-P[0] (start endpoint) and P[N]-P[N-1] (final endpoint). Generally, a user should be allowed to specify the derivatives for clamped splines. New constructors were added to the natural splines classes to support this.

(5.0.3) LibMathematics/CurvesSurfacesVolumes/Wm5NaturalSpline2.h
(5.0.1) LibMathematics/CurvesSurfacesVolumes/Wm5NaturalSpline2.cpp
(5.0.3) LibMathematics/CurvesSurfacesVolumes/Wm5NaturalSpline3.h
(5.0.1) LibMathematics/CurvesSurfacesVolumes/Wm5NaturalSpline3.cpp

July 1, 2012. Added DX9/WGL renderer headers to the precompiled header system. Renamed LoadString to LoadStdString to avoid name clash with the global LoadString in WinUser.h.

(5.0.2) LibGraphics/Wm5Graphics.h
(5.0.2) LibGraphics/Shaders/Wm5VisualEffect.h
(5.0.1) LibGraphics/Shaders/Wm5VisualEffect.cpp

July 1, 2012. Reordered the headers to avoid a warning about redefinition of CALLBACK.

(5.6.2) LibGraphics/Renderers/GlutRenderer/Wm5GlutRendererInput.h

July 1, 2012. Replace PtrToUint by typecast of unsigned int to avoid 64-bit compiler errors.

(5.0.3) LibApplications/WinApplication/Wm5WinApplication.cpp

July 8, 2012. Project redesign and reorganization. Removed support for VC80 and VC90. Support only for VC100. Will add support for the next version of MSVS once Windows8 ships. Added x64 build configurations. Removed NoPCH build configurations. Switched to using macros to generate output directories for graphics-less projects

```
_Output\$(PlatformToolset)\$(Platform)\$(Configuration)
```

or graphics-based projects

```
_Output\$(PlatformToolset)\$(Platform)\Dx9$(Configuration)
_Output\$(PlatformToolset)\$(Platform)\Wgl$(Configuration)
```

Added macros for include/library DX9 paths, $(DXSDK_DIR), and selecting the x86 or x64 library paths accordingly. Eliminated the default vc100.pdb symbol file, using instead $(TargetName).pdb. Samples and tools now use warning level 4 instead of warning level 3. Samples and tools have *.sln files with library dependencies so that they are self-contained for compiling, linking, and running. Added references to the libraries in addition to specifying build order to avoid a MSVS bug.

July 8, 2012. The March 9, 2012 change to Wm5Table.* was incorrect. On the Macintosh, the templates are tested for correctness even when not used. Replaced mat by table.

(5.0.4) LibCore/DataTypes/Wm5Table.inl

July 8, 2012. Converted .cpp files to .inl files and removed the import/export macro to avoid a DLL linker problem in applications. The FloatN classes are exported in the DLL, leading to instantiation of Tuple<N,float> in the library. Applications that use Tuple<N,float> directly have instantiation via the Tuple template, but this causes multiply defined destructors for Tuple<N,float>.

(5.0.2) LibMathematics/Base/Wm5Float1.h
(5.0.1) LibMathematics/Base/Wm5Float1.inl
(5.0.2) LibMathematics/Base/Wm5Float2.h
(5.0.1) LibMathematics/Base/Wm5Float2.inl
(5.0.2) LibMathematics/Base/Wm5Float3.h
(5.0.1) LibMathematics/Base/Wm5Float3.inl
(5.0.2) LibMathematics/Base/Wm5Float3.h
(5.0.1) LibMathematics/Base/Wm5Float3.inl

July 8, 2012. Fixed a bug in the February 19, 2012 bug fix for triangulation.

(5.0.3) LibMathematics/ComputationalGeometry/Wm5TriangulateEC.cpp

July 8, 2012. The logic for selecting the GetRoots function needed to include also whether any of the c0, c1, or c2 are positive or zero.

(5.0.3) LibMathematics/Intersection/Wm5IntrEllipse2Ellipse2.cpp
(5.5.2) LibMathematics/Intersection/Wm5IntrEllipsoid3Ellipsoid3.cpp

July 8, 2012. DiskOverlapsPolygon had a call to Dot that should have been DotPerp.

(5.5.1) LibMathematics/Intersection/Wm5IntrTriangle3Cylinder3.cpp

July 8, 2012. In Clip(), replaced the inequality comparisons to mEpsilon with strict inequalities and added a count of the number of times the point is on the plane (within the specified mEpsilon). This avoids incorrect sign counting when the default mEpsilon=0 is used.

(5.0.1) SampleMathematics/IntersectConvexPolyhedra/ConvexClipper.cpp

July 8, 2012. Fixed compiler error due to removal of base class Index(...) function.

(5.0.3) Tools/WmtfViewer/WmtfViewer.cpp

July 8, 2012. Added/fixed typecast to avoid compiler warning.

(5.0.1) SampleGraphics/BlendedTerrain/BlendedTerrainEffect.cpp
(5.0.1) SampleMathematics/DrawImplicitSurface/DrawImplicitSurface.cpp

July 8, 2012. Removed unused parameters to avoid compiler warning.

(5.1.1) SampleGraphics/Castle/Castle.cpp
(5.0.1) SampleGraphics/SortFaces/SortedCube.cpp
(5.0.1) SampleGraphics/SurfaceMeshes/SimplePatch.cpp
(5.5.1) SampleImagics/AdaptiveSkeletonClimbing3/Climb3D.h
(5.5.2) SampleImagics/AdaptiveSkeletonClimbing3/Climb3D.cpp
(5.5.2) SampleImagics/AdaptiveSkeletonClimbing3/AdaptiveSkeletonClimbing3.cpp
(5.3.1) SampleMathematics/NonlocalBlowup/NonlocalSolver2.cpp
(5.3.1) SampleMathematics/NonlocalBlowup/ConsoleSource/GpuLocalSolver1.cpp
(5.3.1) SampleMathematics/NonlocalBlowup/ConsoleSource/GpuLocalSolver2.cpp
(5.3.1) SampleMathematics/NonlocalBlowup/ConsoleSource/GpuLocalSolver3.cpp
(5.3.1) SampleMathematics/NonlocalBlowup/ConsoleSource/GpuNonlocalSolver1.cpp
(5.3.1) SampleMathematics/NonlocalBlowup/ConsoleSource/GpuNonlocalSolver2.cpp
(5.3.1) SampleMathematics/NonlocalBlowup/ConsoleSource/GpuNonlocalSolver3.cpp
(5.3.1) SampleMathematics/NonlocalBlowup/ConsoleSource/CpuLocalSolver1.cpp
(5.3.1) SampleMathematics/NonlocalBlowup/ConsoleSource/CpuLocalSolver2.cpp
(5.3.1) SampleMathematics/NonlocalBlowup/ConsoleSource/CpuLocalSolver3.cpp
(5.3.1) SampleMathematics/NonlocalBlowup/ConsoleSource/CpuNonlocalSolver1.cpp
(5.3.1) SampleMathematics/NonlocalBlowup/ConsoleSource/CpuNonlocalSolver2.cpp
(5.3.1) SampleMathematics/NonlocalBlowup/ConsoleSource/CpuNonlocalSolver3.cpp
(5.3.1) SampleMathematics/NonlocalBlowup/NonlocalBlowup.cpp
(5.0.1) SampleMathematics/ClodPolyline/VertexCollapse.h
(5.0.1) SampleMathematics/ClodPolyline/VertexCollapse.cpp
(5.0.1) SamplePhysics/BouncingBall/DeformableBall.cpp
(5.2.1) SamplePhysics/CollisionsMovingSpheres/SphereColliders.cpp
(5.0.1) SamplePhysics/Fluids3D/Smoke3D.cpp
(5.1.1) Tools/BmpColorToGray/BmpColorToGray.cpp
(5.1.1) Tools/BmpToWmtf/BmpToWmtf.cpp

July 8, 2012. Initialize variable to avoid compiler warning about potential access to uninitialized variable. We know the variable is initialized but the compiler cannot determine that.

(5.1.1) SampleGraphics/Castle/CreateMeshes.cpp
(5.5.2) SampleImagics/AdaptiveSkeletonClimbing3/Climb3D.cpp

July 8, 2012. Added typecast to avoid compiler warning.

(5.5.1) SampleImagics/AdaptiveSkeletonClimbing2/Climb2D.cpp
(5.5.2) SampleImagics/AdaptiveSkeletonClimbing3/Climb3D.cpp
(5.5.1) SampleImagics/AdaptiveSkeletonClimbing3/LinearMergeTree.cpp
(5.5.2) SampleImagics/AdaptiveSkeletonClimbing3/VETable.cpp

July 8, 2012. Rewrote the class to eliminate the const reference members, instead passing the necessary input data through the functions.

(5.0.1) SampleMathematics/ClipMesh/PartitionMesh.h
(5.0.1) SampleMathematics/ClipMesh/PartitionMesh.cpp

July 8, 2012. Removed const modifier from application class member to avoid compiler warning about not being able to generate an assignment operator for that class.

(5.0.1) SampleMathematics/ConvexHull2D/ConvexHull2D.h
(5.0.1) SamplePhysics/CollisionsBoundTree/CollisionsBoundTree.h

July 8, 2012. Removed parameters from CreateSphere .

(5.0.1) SampleMathematics/ConvexHull2D/ConvexHull3D.h
(5.0.1) SampleMathematics/ConvexHull2D/ConvexHull3D.cpp

July 8, 2012. Removed unused parameters from Main. Initialized variables to avoid compiler warnings about potential access to uninitialized variable. We know the variables are initialized but the compiler cannot determine that. The function DistancePointHyperEllipsoid had a reflect[j] in its restore loop, but that should be reflect[i].

(5.5.2) SampleMathematics/IntersectionEllipseEllipsoid/IntersectionEllipseEllipsoid.cpp
(5.6.1) SampleMathematics/ThinPlateSplines/ThinPlateSplines.cpp
(5.6.2) SampleMathematics/DistancePointEllipseEllipsoid/DistancePointEllipseEllipsoid.cpp

July 8, 2012. Modified GenerateProjects to create new projects supporting Win32 and x64. Projects are now generated separately for Dx9 or Wgl. Removed support for VC80 and VC90.

(5.0.2) Tools/GenerateProjects/GenerateProjects.cpp
(5.2.1) Tools/GenerateProjects/TemplateVC100.h
(5.2.3) Tools/GenerateProjects/TemplateVC100.cpp
Tools/GenerateProjects/TemplateVC80.h
Tools/GenerateProjects/TemplateVC80.cpp
Tools/GenerateProjects/TemplateVC90.h
Tools/GenerateProjects/TemplateVC90.cpp

9 Updates to Version 5.8

July 15, 2012. Added a new document that shows how to compute the perspective projection of an ellipse onto a plane.

Documentation/PerspectiveProjectionEllipse.pdf

July 29, 2012. Wm4::GMatrix<Real> had implicit conversions to Real* and const Real* pointers, but they were replaced by Wm5::GMatrix<Real>::GetElements(). The function Wm5::Matrix3<Real>::SingularValueDecomposition used the implicit conversions but was not updated correctly, so instantiation of this function has compiler errors that are now fixed. There was also a cut-and-paste error that had a memcpy with 9*sizeof(double) that should have been 9*sizeof(Real).

(5.0.2) LibMathematics/Algebra/Wm5Matrix3.inl

July 29, 2012. The file needs to include Wm5GlExtensions.h in order to compile as a stand-alone file, because GLenum is used in the declaration of GTGetErrorString.

(5.0.2) LibGraphics/Renderers/OpenGLRenderer/Wm5GlUtility.h

July 29, 2012. The file needs to include Wm5Memory.h in order to compile as a stand-alone file when an explicit instantiation of the class occurs.

(5.0.1) LibMathematics/Meshes/Wm5UniqueVerticesTriangles.h

August 5, 2012. Updated Xcode projects to version 4.4 for Mac OS X 10.8 Mountain Lion. Added dynamic library build configurations to all samples. Fixed the shell script to work for dynamic build configurations.

MacBuildWm5.sh

August 5, 2012. Updated for Xcode 4.4 recommended settings. Replaced TemplateXC32.* by TemplateXC44.*. Now we generate configurations for both static and dynamic libraries.

(5.0.3) Tools/GenerateProjects/GenerateProjects.cpp
(5.9.0) Tools/GenerateProjects/TemplateXC44.h
(5.9.0) Tools/GenerateProjects/TemplateXC44.cpp
Tools/GenerateProjects/TemplateXC32.h
Tools/GenerateProjects/TemplateXC32.cpp
Tools/GenerateProjects/GenerateProjects_VC100.vcxproj
Tools/GenerateProjects/GenerateProjects_VC100.vcxproj.filter

August 5, 2012. Apple LLVM compiler 4.0 found a logic error in a loop (comma operator was used instead of AND). The code happened to work for this particular sample.

(5.0.1) SamplePhysics/BouncingTetrahedra/BouncingTetrahedra.cpp

August 5, 2012. Added parentheses to silence an Apple LLVM compiler 4.0 warning.

(5.0.1) SamplePhysics/PolyhedronDistance/PolyhedronDistance.cpp

August 5, 2012. Construct a named LCPSolver object to avoid warning with XCode 4.4 and Apple LLVM compiler 4.0.

(5.0.2) LibPhysics/LCPSolver/Wm5LCPPolyDist.cpp

10 Updates to Version 5.9

September 26, 2012. I had accidentally exposed WM5_USE_MEMORY for all configurations. This is now turned off for all configurations. You can expose it, if you prefer, for Debug configurations. Or you can add

it to MSVS project configurations as desired. The reason for turning it off is that some users were unaware of the requirements for calling Memory::Initialize and Memory::Terminate in main, which led to assertions triggered when dynamic memory is used pre-main or post-main. Turning it off led to fewer (incorrect) bug reports.

(5.0.8) LibCore/Wm5CoreLIB.h

October 31, 2012. When building the control points matrix Q, the indices were swapped (mControls[i1][i0] is correct, not mControls[i0][i1]). The problem showed up for non-square grids of points. Also, the indices on mSamples[][] were swapped. These were fixed, including in GetPosition. The comments about constraints between the degrees and number of controls points was incorrect (degree* + 1 < numControls* is instead required) and the assertions in the constructor were modified accordingly.

(5.0.2) LibMathematics/CurvesSurfacesVolumes/Wm5BSplineSurfaceFit.h
(5.0.2) LibMathematics/CurvesSurfacesVolumes/Wm5BSplineSurfaceFit.cpp

November 3, 2012. Small thresholds are used for testing the discriminant of the quadratic equation related to the computations: $Q(t) = a_2t^2 + 2a_1t + a_0$. The discriminant is $D = a_1a_1 - a_0a_2$. $Q(t)$ has no real-valued roots when $D < 0$, one real-valued root when $D = 0$, or two real-valued roots when $D > 0$. The code logic involves user-defined thresholds:

```
if (D < negThreshold) { no roots (no intersections) }
else if (D > posThreshold) { two roots (two intersections) }
else { one root (one intersection) }
```

The default values for the thresholds are zero, but you may set them to be nonzero (negThreshold ≤ 0 and posThreshold ≥ 0). Previously, the negative threshold was hard-coded as zero. The positive threshold was hard-coded to Math::ZERO_TOLERANCE, which is not suitable for some data sets (i.e. when ellipsoid extents are quite large). The default is now zero, so if your application relied on the old behavior, you must modify this value.

(5.0.2) LibMathematics/Intersection/Wm5IntrLine3Ellipsoid3.h
(5.0.2) LibMathematics/Intersection/Wm5IntrLine3Ellipsoid3.cpp

November 3, 2012. The intersection testing uses the center-extent form for line segments. If you start with endpoints (Vector2<Real>) and create Segment2<Real> objects, the conversion to center-extent form can contain small numerical round-off errors. Testing for the intersection of two egments that share an endpoint might lead to a failure due to the round-off errors. To allow for this, you may specify a small positive threshold that slightly enlarges the intervals for the segments. The default value is zero. The computation for determining whether the segments are parallel might contain small floating-point round-off errors. The algorithm reduces to testing whether a dot product is zero or nonzero. You may specify a small positive threshold to adjust the test (dot == 0) to (fabs(dot) > 0). The default threshold is zero.

((5.0.2) LibMathematics/Intersection/Wm5IntrSegment2Segment2.h
((5.0.2) LibMathematics/Intersection/Wm5IntrSegment2Segment2.cpp

November 3, 2012. The root finder Cubic(c0,c1,c2) in the case of positive discriminant had several errors.

(5.0.3) LibMathematics/NumericalAnalysis/Wm5PolynomialRootsR.cpp

November 3, 2012. Fixed a minor notational problem. Section 2 defined a point `r = (y0, y1, y2)`. Starting with equation (10), values `y0` and `y1` were introduced as coefficients of a linear combination involving `r`, but these coefficients are not the components of `r`. I revised Section 2 to use `r = (r0,r1,r2)` in order to avoid the confusion.

Documentation/PerspectiveMappings.pdf

November 3, 2012. Equation (15) for the cubic discriminant had a typographical error in a power.

Documentation/LowDegreePolynomialRoots.pdf

November 22, 2012. Some line-object squared-distance queries used temporary line objects. The temporary objects are accessed by reference in the query objects, so use of the query objects after their construction led to dereferencing already destroyed temporary objects.

(5.0.2) LibMathematics/Distance/Wm5DistRay3Rectangle3.cpp
(5.0.2) LibMathematics/Distance/Wm5DistRay3Triangle3.cpp
(5.0.2) LibMathematics/Distance/Wm5DistSegment3Rectangle3.cpp
(5.0.2) LibMathematics/Distance/Wm5DistSegment3Triangle3.cpp

November 22, 2012. Minor loop changes led to better performance (2- to 3-time speedup). Code comments describe the changes.

(5.0.3) LibMathematics/Containment/Wm5ContMinCircle2.cpp
(5.0.5) LibMathematics/Containment/Wm5ContMinSphere3.cpp

November 23, 2012. Added quotes to the mkdir and xcopy arguments involving the $(Configuration) macro. This avoids post-build errors when the configuration name has spaces.

LibApplications/LibDx9Applications_VC100.vcxproj
LibApplications/LibWglApplications_VC100.vcxproj
LibApplications/LibGlutApplications_VC100.vcxproj
LibCore/LibCore_VC100.vcxproj
LibGraphics/LibDx9Graphics_VC100.vcxproj
LibGraphics/LibWglGraphics_VC100.vcxproj
LibGraphics/LibGlutGraphics_VC100.vcxproj
LibImagics/LibImagics_VC100.vcxproj
LibMathematics/LibMathematics_VC100.vcxproj
LibPhysics/LibPhysics_VC100.vcxproj

November 23, 2012. Created Microsoft Visual Studio 2012 project files for the main Wild Magic libraries. The DX9 library is set up to use the June 2010 DirectX SDK. Be aware that MSVS 2012 ships with the DirectX 11.1 SDK, in the Windows Kits folder in the Program Files folder. It is possible to use the June 2010 DirectX SDK with Windows 8 as long as you have the MSVS 2010 runtime installed. I will generate MSVS 2012 project files for the samples shortly.

LibApplications/LibDx9Applications_VC110.vcxproj
LibApplications/LibDx9Applications_VC110.vcxproj.filters
LibApplications/LibWglApplications_VC110.vcxproj
LibApplications/LibWglApplications_VC110.vcxproj.filters
LibApplications/LibGlutApplications_VC110.vcxproj
LibApplications/LibGlutApplications_VC110.vcxproj.filters
LibCore/LibCore_VC110.vcxproj
LibCore/LibCore_VC110.vcxproj.filters
LibGraphics/LibDx9Graphics_VC110.vcxproj
LibGraphics/LibDx9Graphics_VC110.vcxproj.filters
LibGraphics/LibWglGraphics_VC110.vcxproj
LibGraphics/LibWglGraphics_VC110.vcxproj.filters
LibGraphics/LibGlutGraphics_VC110.vcxproj
LibGraphics/LibGlutGraphics_VC110.vcxproj.filters
LibImagics/LibImagics_VC110.vcxproj
LibImagics/LibImagics_VC110.vcxproj.filters
LibMathematics/LibMathematics_VC110.vcxproj
LibMathematics/LibMathematics_VC110.vcxproj.filters
LibPhysics/LibPhysics_VC110.vcxproj
LibPhysics/LibPhysics_VC110.vcxproj.filters

November 23, 2012. In order to build the GLUT graphics and application projects, the GLUT files must be copied as shown next.

```
C:\Program Files (x86)\Windows Kits\8.0\Include\um\gl\glut.h
C:\Program Files (x86)\Windows Kits\8.0\Lib\win8\um\x86\glut.def
C:\Program Files (x86)\Windows Kits\8.0\Lib\win8\um\x86\glut32.lib
```

November 23, 2012. Fixed a compile issue with std::greater. In MSVS 2010 this used to be accessed by inclusion of other STL headers, but not so in MSVS 2012. I added an include of <functional> to the core library header.

(5.0.10) LibCore/Wm5CoreLIB.h

December 15, 2012. The constructor had an initialization loop with an incorrect upper limit.

(5.0.1) LibGraphics/Resources/Wm5VertexBufferAccessor.cpp

January 3, 2013. The Disable function accessed the wrong profile.

(5.0.2) LibGraphics/Renderers/Dx9Renderer/Wm5Dx9VertexShader.cpp

January 3, 2013. The last loops in the SqrDistance functions had reflect[j] that should have been reflect[i].

(5.0.3) LibMathematics/Distance/Wm5DistPoint2Ellipse2.cpp
(5.0.4) LibMathematics/Distance/Wm5DistPoint3Ellipsoid3.cpp
(5.6.3) SampleMathematics/DistancePointEllipseEllipsoid/DistancePointEllipseEllipsoid.cpp

February 6, 2013. The GetPoint(int) function was intended to be used for returning the first point of contact between a moving segment and moving triangle (or the endpoints if the first contact is a segment). If called after the bool Find() call (stationary segment/triangle), GetPoint(int) returns uninitialized data. I modified bool Find() to store the point of intersection. The GetSegmentParameter() and GetTriBary*() functions were designed for you to compute the point explicitly for bool Find(). The bool Find() function was missing the assignment of mIntersectionType, so that was fixed.

(5.0.2) LibMathematics/Intersection/Wm5IntrSegment3Triangle3.h
(5.0.3) LibMathematics/Intersection/Wm5IntrSegment3Triangle3.cpp

March 1, 2013. The SetAlphaState function had a cut-and-paste error that sets the blend color blue channel to the green value.

(5.0.4) LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLRenderer.cpp

March 1, 2013. The FindContactSet function, line 1361, had a GetPointFromIndex function that used b0Index but should use b1Index.

(5.0.2) LibMathematics/Intersection/Wm5IntrUtility.cpp

March 1, 2013. Fixed a logic error in Triangles::UpdateModelTangentsUseTCoords. The last part of the loop was setting vertex attributes for index curr, but should have set for the associated vertex index, now in the code as v[curr].

(5.0.1) LibGraphics/SceneGraph/Wm5Triangle.cpp

March 17, 2013. Added new function ComputeDistancePointToHyperbola.

(5.10.0) LibMathematics/Distance/Wm5DistPoint2Hyperbola2.h
(5.10.0) LibMathematics/Distance/Wm5DistPoint2Hyperbola2.cpp
(5.0.5) LibMathematics/Wm5Mathematics.h

March 17, 2013. Added support for generating MSVS 2012 projects (v110).

(5.10.0) Tools/GenerateProjects/TemplateVC110.h
(5.10.0) Tools/GenerateProjects/TemplateVC110.cpp
(5.0.4) Tools/GenerateProjects/GenerateProjects.cpp
Tools/GenerateProjects/GenerateProjects_VC100.vcxproj
Tools/GenerateProjects/GenerateProjects_VC100.vcxproj.filters

May 12, 2013. Corrected the comments that describe the index output.

(5.0.2) LibMathematics/ComputationalGeometry/Wm5ConvexHull.h

July 13, 2013. Replaced glTexSubImage* calls in PdrTexture*::Unlock() to work around an apparent bug in AMD OpenGL drivers. The glTexSubImage* calls are generating GL_INVALID_OPERATION when the texture does not have a complete set of mipmaps. NVIDIA OpenGL drivers work correctly on these calls.

(5.0.2) LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLTexture1D.cpp
(5.0.2) LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLTexture2D.cpp
(5.0.2) LibGraphics/Renderers/OpenGLRenderer/Wm5OpenGLTexture3D.cpp

July 14, 2013. The application incorrectly compared the modifier against KEY_SHIFT rather than MODIFIER_SHIFT. This had worked on Windows 7 but not on Windows 8.

(5.0.1) SampleMathematics/Delaunay2DInsertRemove/Delaunay2DInsertRemove.cpp

July 14, 2013. The polyline for the path of the object is modified during the simulation, but the model-space bound needed to be recomputed and a scene update had to have occurred in order for the culler to notice the path needed to be drawn.

(5.0.1) SamplePhysics/BallHill/BallHill.cpp
(5.0.1) SamplePhysics/FoucaultPendulum/FoucaultPendulum.cpp

July 14, 2013. Because of my use of OpenGL pre-version-2 extensions, the NORMAL semantic in the Cg programs works fine with NVIDIA graphics cards but not with AMD graphics cards. The registers used for normals are different between the two. Until I rewrite the OpenGL renderer to use version-2 shader support, I have worked around the problem by using TEXCOORD semantics for normal vectors, modifying a couple of sample data sets, shaders, and sample code.

(5.0.1) LibGraphics/LocalEffects/Wm5LightDirPerPixEffect.cpp
(5.0.1) LibGraphics/LocalEffects/Wm5LightDirPerVerEffect.cpp
(5.0.1) LibGraphics/LocalEffects/Wm5LightPntPerPixEffect.cpp
(5.0.1) LibGraphics/LocalEffects/Wm5LightPntPerVerEffect.cpp
(5.0.1) LibGraphics/LocalEffects/Wm5LightSptPerPixEffect.cpp
(5.0.1) LibGraphics/LocalEffects/Wm5LightSptPerVerEffect.cpp
(5.1.1) SampleGraphics/Castle/LoadData.cpp
(5.0.1) SampleGraphics/GlossMaps/GlossMaps.cpp
(5.0.1) SampleGraphics/Lights/Lights.cpp
(5.0.1) SampleGraphics/MorphControllers/MorphControllers.cpp
(5.1.1) SampleGraphics/MorphFaces/MorphFaces.cpp
(5.0.1) SampleGraphics/PlanarReflections/PlanarReflections.cpp
(5.0.1) SampleGraphics/PlanarReflections/PlanarReflections.h
(5.0.1) SampleGraphics/PlanarShadows/PlanarShadows.cpp
(5.0.1) SampleGraphics/PlanarShadows/PlanarShadows.h
(5.0.1) SampleGraphics/ProjectedTextures/ProjectedTextures.cpp
(5.0.1) SampleGraphics/ReflectionsAndShadows/ReflectionsAndShadows.cpp
(5.0.1) SampleGraphics/ReflectionsAndShadows/ReflectionsAndShadows.h
(5.0.1) SampleGraphics/ScreenPolygons/ScreenPolygons.cpp
(5.0.1) SampleGraphics/ScreenPolygons/ScreenPolygons.h
(5.0.2) SampleGraphics/SkinnedBiped/SkinnedBiped.cpp
(5.0.1) SampleGraphics/SkinnedBiped/SkinnedBiped.h
(5.0.1) SampleGraphics/SphereMaps/SphereMaps.cpp
(5.0.1) SampleGraphics/SphereMaps/SphereMaps.h
(5.0.1) SampleImagics/ExtractLevelSurfacesCubes/ExtractLevelSurfacesCubes.cpp
(5.0.1) SampleImagics/ExtractLevelSurfacesTetra/ExtractLevelSurfacesTetra.cpp
(5.0.1) SampleMathematics/GeodesicHeightField/GeodesicHeightField.cpp
(5.0.1) SamplePhysics/FoucaultPendulum/FoucaultPendulum.cpp

(5.0.1) SamplePhysics/IntersectingBoxes/IntersectingBoxes.cpp
(5.0.1) SamplePhysics/SimplePendulumFriction/SimplePendulumFriction.cpp

July 15, 2013. Added projects and solutions for Microsoft Visual Studio 2012.

11 Updates to Version 5.10

July 28, 2013. Rewrote the line-circle distance and closest-point document. The previous version reduced the closest-point problem to solving a quartic polynomial. This polynomial can be degenerate, and it was not clear when you had multiple pairs of closest points. The new version provides a robust algorithm that makes clear what are the different geometric cases. The source code was modified to use the new algorithm.

Documentation/DistanceLine3Circle3.pdf
Documentation/DistancePoint3Circle3.pdf
(5.0.2) LibMathematics/Distance/Wm5DistLine3Circle3.h
(5.0.2) LibMathematics/Distance/Wm5DistLine3Circle3.cpp
(5.0.2) LibMathematics/Distance/Wm5DistPoint3Circle3.h
(5.0.2) LibMathematics/Distance/Wm5DistPoint3Circle3.cpp

July 28, 2013. The last parts of the permutation code had reflect[j] that should have been reflect[i]. The C++ source code used to have the same problem but was fixed some time ago.

Documentation/DistancePointEllipseEllipsoid.pdf

August 4, 2013. Added new documents on testing for intersection of rectangle-ellipse and of box-ellipsoid.

Documentation/IntersectionRectangleEllipse.pdf
Documentation/IntersectionBoxEllipsoid.pdf

August 11, 2013. Fixed several typographical errors. Simplified the pseudocode and switched to using lstlisting rather than verbatim.

Documentation/OrthonormalSets.pdf

August 13, 2013. Added a document to describe the nonuniform Akima interpolation algorithm implemented in our source code.

Documentation/AkimaInterpolation.pdf

November 12, 2013. The Ray2–Ray2, Ray2–Segment2, and Segment2–Segment2 incorrectly reported intersections when the components were collinear but not intersecting. Rewrote the Line2/Ray2/Segment2 test-find intersection code to expose some thresholds to the user and to handle properly the collinear cases.

(5.0.3) LibMathematics/Intersection/Wm5IntrRay2Ray2.cpp
(5.0.2) LibMathematics/Intersection/Wm5IntrLine2Line2.h

(5.0.3) LibMathematics/Intersection/Wm5IntrLine2Line2.cpp
(5.0.2) LibMathematics/Intersection/Wm5IntrLine2Ray2.h
(5.0.2) LibMathematics/Intersection/Wm5IntrLine2Ray2.cpp
(5.0.2) LibMathematics/Intersection/Wm5IntrLine2Segment2.h
(5.0.2) LibMathematics/Intersection/Wm5IntrLine2Segment2.cpp
(5.0.2) LibMathematics/Intersection/Wm5IntrRay2Ray2.h
(5.0.2) LibMathematics/Intersection/Wm5IntrRay2Ray2.cpp
(5.0.2) LibMathematics/Intersection/Wm5IntrRay2Segment2.h
(5.0.2) LibMathematics/Intersection/Wm5IntrRay2Segment2.cpp
(5.0.3) LibMathematics/Intersection/Wm5IntrSegment2Segment2.h
(5.0.3) LibMathematics/Intersection/Wm5IntrSegment2Segment2.cpp

November 12, 2013. The GMatrix<Real>::operator(int,int) functions have incorrect assertions. The column tests should have col $<$ mNumColumns, not col $\leq$ mNumColumns.

(5.0.6) LibMathematics/Algebra/Wm5GMatrix.inl

November 12, 2013. Modified the document to make explicit the assumption that the ellipsoid is in front of the eyepoint, in which case the projection onto the view plane must be an ellipse. Without this assumption, the projection could also be a hyperbola or parabola.

Documentation/PerspectiveProjectionEllipsoid.pdf

November 19, 2013. The file was missing the implementations for {Set,Get}DotThreshold.

(5.0.3) LibMathematics/Intersection/Wm5IntrRay2Ray2.cpp

December 14, 2013. Modified the copy scripts for Xcode version 5.

December 14, 2013. The upper-right element of the $d = 5$ matrix needed to be -1 rather than 1.

Documentation/BSplineInterpolation.pdf

December 14, 2013. Rewrote the document for fitting a point cloud with a cylinder. This version includes pseudocode for the implementation and screen captures for the fitting of several data sets.

Documentation/CylinderFitting.pdf

January 4, 2014. The Delaunay3 class for Delaunay tetrahedralization was not working correctly, even for exact rational arithmetic. The handling of the supertetrahedron and the visibility tests were not correct. For Delaunay2, inserting a point at a time into the supertriangle, starting with the first point, appears to work. This does not work for Delaunay3 and the supertetrahedron because of geometric and visibility issues due to the increase in dimensionality. I never liked the super-simplex approach, because it is not possible in advance to know how large the simplex must be in order to guarantee that the circum-ball tests are correct. Finally, I have abandoned the super-simplex approach, replacing it by an algorithm that updates the simplices in one of two ways: (1) The incoming point is inside the current set of simplices (inside the current convex hull of the points). (2) The point is outside the simplices (convex hull). Item (2) requires visibility tests for

half-spaces that are effectively the circum-ball tests when the super-simplex vertices are moved to the point at infinity. NOTE: You can be sure of the correctness of the output ONLY by using Query::QT_RATIONAL (exact rational arithmetic in order to classify correctly the signs of determinants in the visibility tests and in the circumcircle and circumsphere tests).

(5.0.2) LibMathematics/ComputationalGeometry/Wm5Delaunay2.h
(5.0.4) LibMathematics/ComputationalGeometry/Wm5Delaunay2.cpp
(5.0.2) LibMathematics/ComputationalGeometry/Wm5Delaunay3.h
(5.0.5) LibMathematics/ComputationalGeometry/Wm5Delaunay3.cpp

January 4, 2014. Added classes UnorderedTriangleKey, TetrahedronKey, and TSManifoldMesh for constructing a manifold tetrahedron mesh. Removed the operator size_t() functions (not used in Wild Magic). Added OrderedEdgeKey. These changes support modifications to Delaunay2 and Delaunay3.

(5.0.2) LibMathematics/Meshes/Wm5EdgeKey.h
(5.0.1) LibMathematics/Meshes/Wm5EdgeKey.inl
(5.0.2) LibMathematics/Meshes/Wm5TriangleKey.h
(5.0.1) LibMathematics/Meshes/Wm5TriangleKey.inl
(5.10.0) LibMathematics/Meshes/Wm5TetrahedronKey.h
(5.10.0) LibMathematics/Meshes/Wm5TetrahedronKey.cpp
(5.10.0) LibMathematics/Meshes/Wm5TSManifoldMesh.h
(5.10.0) LibMathematics/Meshes/Wm5TSManifoldMesh.cpp
(5.0.6) LibMathematics/Wm5Mathematics.h

January 4, 2014. Fixed a typographical error in the comments for ToPlane.

(5.0.5) LibMathematics/Query/Wm5Query3.h

January 4, 2014. Modified the samples to useQT_RATIONAL by default.

(5.0.5) SampleMathematics/Delaunay2D/Delaunay2D.cpp
(5.0.5) SampleMathematics/Delaunay3D/Delaunay3D.cpp

January 9, 2014. These files compiled with MSVS 2012/2013 but not with MSVS 2010 because of using C++ 11 for (auto...) loops. Replaced these with C++ 10 style loops over iterators so that the MSVS 2010 builds succeed.

(5.10.1) LibMathematics/Meshes/Wm5TSManifoldMesh.cpp
(5.0.5) LibMathematics/ComputationalGeometry/Wm5Delaunay2.cpp
(5.0.6) LibMathematics/ComputationalGeometry/Wm5Delaunay3.cpp

January 21, 2014. Updated all the Xcode projects to version 5.0.2

January 21, 2014. Replaced the C++ 11 std::array declarations by regular arrays. Removed use of auto. Then I did not have to use compiler flag std= $c++11$ on Fedora 20 with gcc-4.8.2. Added an include of Wm5Memory.h to Wm5TSManifoldMesh to avoid g++ compile error. Changed the input type for Wm5TSManifoldMesh::Print to be consistent with other mesh classes.

(5.0.3) LibMathematics/ComputationalGeometry/Wm5Delaunay2.h
(5.0.6) LibMathematics/ComputationalGeometry/Wm5Delaunay2.cpp
(5.0.7) LibMathematics/ComputationalGeometry/Wm5Delaunay3.cpp
(5.10.1) LibMathematics/Meshes/Wm5TetrahedronKey.h
(5.10.1) LibMathematics/Meshes/Wm5TetrahedronKey.cpp
(5.10.1) LibMathematics/Meshes/Wm5TSManifoldMesh.h
(5.10.1) LibMathematics/Meshes/Wm5TSManifoldMesh.cpp

January 21, 2014. Removed unused mN member.

(5.5.1) SampleImagics/AdaptiveSkeletonClimbing2/AreaMergeTree.h
(5.5.1) SampleImagics/AdaptiveSkeletonClimbing2/AreaMergeTree.cpp
(5.5.1) SampleImagics/AdaptiveSkeletonClimbing2/Climb2D.h
(5.5.2) SampleImagics/AdaptiveSkeletonClimbing2/Climb2D.cpp
(5.5.1) SampleImagics/AdaptiveSkeletonClimbing2/LinearMergeTree.h
(5.5.1) SampleImagics/AdaptiveSkeletonClimbing2/LinearMergeTree.cpp
(5.5.2) SampleImagics/AdaptiveSkeletonClimbing3/Climb3D.h
(5.5.3) SampleImagics/AdaptiveSkeletonClimbing3/Climb3D.cpp
(5.5.1) SampleImagics/AdaptiveSkeletonClimbing3/LinearMergeTree.h
(5.5.2) SampleImagics/AdaptiveSkeletonClimbing3/LinearMergeTree.cpp

12 Updates to Version 5.11

January 23, 2014. Yet another test case that exposed a problem. In the Update(i) function, when point i is inside the current hull, a containing tetrahedron is identified. That tetrahedron is subdivided into 4 tetrahedron with common vertex i. When point i is on a tetrahedron face (but not an edge), there must be only 3 subtetrahedra. When point i is on a tetrahedron edge, there must be only 2 subtetrahedra. The initialization code already removes duplicate vertices, so no fear of point i being an already existing vertex. Added a test for degenerate subtetrahedra, ignoring them when they occur.

(5.0.7) LibMathematics/ComputationalGeometry/Wm5Delauny2.cpp
(5.0.8) LibMathematics/ComputationalGeometry/Wm5Delauny3.cpp

January 23, 2014. Renamed some input variables for consist naming convention (not a code change).

(5.0.2) LibMathematics/Query/Wm5Query3.h

February 1, 2014. The matrix $A^{\mathrm{T}}A$ is size $(n+1)\times(n+1)$, where $n+1$ is the number of control points. The matrix is said to be banded with $d+1$ lower bands and $d+1$ upper bands. This is true when $n > d$, where d is the degree of the polynomial. The case $n = d$ (one more control point than the degree of the polynomial) is an edge case, and the matrix has d lower bands and d upper bands.

(5.0.2) LibMathematics/CurvesSurfacesVolumes/Wm5BSplineCurveFit.cpp
Documentation/BSplineCurveLeastSquaresFit.pdf

February 14, 2014. In the case of cylinder axis intersecting the plane in a unique point and axis direction not parallel to the plane, the Find function set the ellipse center using the projection center rather than mCylinder-¿Axis.Origin.

(5.0.2) LibMathematics/Intersection/Wm5IntrPlane3Cylinder3.cpp

February 28, 2014. Fixed a typographical error on a subscript. Change the name X to P in the final equations to match the description provided previously.

Documentation/BicubicExactInterpolation.pdf

March 10, 2014. The function CreateFreeSpline allocated delta, invDelta, and fDeriv to have mNumSamples values. These needed to be mNumSegments and the loop that initializes them need an upper bound of mNumSegments.

(5.0.2) LibMathematics/CurvesSurfacesVolumes/Wm5NaturalSpline1.cpp

March 10, 2014. On page 6, the pseudocode for factoring into xzy-rotation when z = +pi/2 had theta_x = atan2(-r20,r22). The right-hand side should be -atan2(-r20,r22). The Wild Magic source code, however, has the correct equation.

Documentation/EulerAngles.pdf

April 10, 2014. The top-level solutions files for MSVS 2012 and MSVS 2013 showed the correct build order for the libraries, but the parallel compiling in multiple threads ignored the order. The library build order was determined from the project reference mechanism in all the sample applications, but the libraries themselves do not have the references. The project dependencies also had to be properly set up (via the check box dialog).

WildMagic5Dx9_VC100.sln
WildMagic5Dx9_VC110.sln
WildMagic5Dx9_VC120.sln
WildMagic5Wgl_VC100.sln
WildMagic5Wgl_VC110.sln
WildMagic5Wgl_VC120.sln

May 14, 2014. Fixed several errors in the pseudocode. Added a comment that the pseudocode uses left-handed coordinates, because the original test code was written for polyline data in an image with origin at the upper-left corner.

Documentation/MinimalCycleBasis.pdf

July 2, 2014. Replaced the EigenDecomposition and SingularValueDecomposition implementations by high-quality ones. This code was back-ported from the Geometric Tools Engine that will be released in August 2014. The public interfaces of the two classes were not modified so as not to break anyone's application code. The first four files in the list are the back-ported classes. They can be used stand-alone if desired.

(5.12.0) LibMathematics/NumericalAnalysis/Wm5SymmetricEigensolverGTE.h
(5.12.0) LibMathematics/NumericalAnalysis/Wm5SymmetricEigensolverGTE.cpp
(5.12.0) LibMathematics/NumericalAnalysis/Wm5SingularValueDecompositionGTE.h

(5.12.0) LibMathematics/NumericalAnalysis/Wm5SingularValueDecompositionGTE.cpp
(5.0.2) LibMathematics/NumericalAnalysis/Wm5EigenDecomposition.h
(5.0.3) LibMathematics/NumericalAnalysis/Wm5EigenDecomposition.cpp
(5.0.2) LibMathematics/NumericalAnalysis/Wm5SingularValueDecomposition.h
(5.0.3) LibMathematics/NumericalAnalysis/Wm5SingularValueDecomposition.cpp

13 Updates to Version 5.12

July 9, 2014. The initial angle was computed using acos, but this can generate an angle in the wrong quadrant. The call was replaced by atan2 and the initial minimum and maximum angles were adjusted accordingly. This code is quite old and uses the opposite convention for factoring a symmetric matrix M. In other parts of the engine, $M = RDR^{\mathrm{T}}$, where R is a rotation, D is diagonal, and R^{T} is the transpose of R. The columns of R are the eigenvectors of M. In this old code, the factoring is $M = R^{\mathrm{T}}DR$, so the rows of R are the eigenvectors (the ellipse axis directions). Unfortunately, this caused the extents e0 and e1 in the constructor to be incorrectly computed, because the columns were used for the axis directions. Moreover, the MakeRotation call needs to be passed the negative of the minimizing angle (pass -vMin[4] instead of vMin[4]). Finally, the Energy function had two wrongs that made it right! The code was modified to be consistent with the (old) convention: The matrix construction is now passed -input[4] rather than input[4] and the diff*rotate becomes rotate*diff.

(5.0.2) LibMathematics/Approximation/Wm5ApprEllipseFit2.cpp

July 9, 2014. The symmetric eigensolver can return an eigenvector matrix that is a rotation or a reflection. Without sorting, the matrix is a rotation whenever the size is even; otherwise, it is a reflection. With sorting, however, the type of the matrix depends on the permutation of columns induced by the sorting. Added code to track the permutations, and added IsRotation() to report the type of matrix. Before new symmetric eigensolver, EigenDecomposition kept track of the permutations correctly. The new code broke that, hence the need for IsRotation().

(5.12.1) LibMathematics/NumericalAnalysis/Wm5SymmetricEigensolverGTE.h
(5.12.1) LibMathematics/NumericalAnalysis/Wm5SymmetricEigensolverGTE.cpp
(5.0.4) LibMathematics/NumericalAnalysis/Wm5EigenDecomposition.cpp

July 9, 2014. Fixed several cut-and-paste errors where variables involving z instead were using y.

(5.0.3) LibMathematics/Interpolation/Wm5IntpTrilinear3.cpp

14 Updates to Version 5.13

November 22, 2015. Fixed VS2015 compiler warnings about inner scope variable names same as in outer scope. Fixed a crash bug—the vertex format for the mesh specified 3 attributes but added 4 to the format; changed the input to 4.

(5.0.2) SamapleMathematics/GeodesicHeightField/GeodesicHeightField.cpp

November 23, 2015. Added testing for OpenGL 4.1 and later. Added a default return of version 4.0 if subversion is not yet trapped by the test.

(5.0.2) LibGraphics/Renderers/OpenGLRenderer/Wm5GlExtenions.cpp
(5.0.3) LibGraphics/Renderers/OpenGLRenderer/Wm5GlUtility.h

November 25, 2015. Fixed VS2015 compiler warnings about inner scope variable names same as in outer scope.

(5.0.2) SamplePhysics/BouncingBall/DeformableBall.cpp

November 27, 2015. Replaced the ZERO_TOLERANCE by 0 in the Normalize functions. This avoids catastrophic failures in some algorithms when the normal vector is set to $(0, 0, 0)$.

LibMathematics/Algebra/Wm5AVector.h
LibMathematics/Algebra/Wm5GVector.h
LibMathematics/Algebra/Wm5HPlane.h
LibMathematics/Algebra/Wm5HQuaternion.h
LibMathematics/Algebra/Wm5Quaternionr.h
LibMathematics/Algebra/Wm5Vector2.h
LibMathematics/Algebra/Wm5Vector3.h
LibMathematics/Algebra/Wm5Vector4.h

November 27, 2015. XKeycodeToKeysym has been deprecated due to a bug. Using the suggested replacement.

LibApplications/GlxApplication/Wm5GlxApplication.cpp

November 27, 2015. Added projects and solutions for Microsoft Visual Studio 2015. Modified the minor versions in the Linux makefiles. Converted MSVS 2013 projects to avoid continue-and-edit and to flag warnings as errors. Got the code to compile on Mac OS X 10.11.0 (El Capitan) and Xcode 7.1.1.

LibApplications/makefile.wm5
LibCore/makefile.wm5
LibGraphics/makefile.wm5
LibImagics/makefile.wm5
LibMathematics/makefile.wm5
LibPhysics/makefile.wm5
LibCore/makeprj.wm5
LibGraphics/makeprj.wm5
LibImagics/makeprj.wm5
LibMathematics/makeprj.wm5
LibPhysics/makeprj.wm5
LibGraphics/Renderers/GlxRenderer/makerend.wm5

November 27, 2015. Added project generation for MSVS 2013 and MSVS 2015.

Tools/GenerateProjects/GenerateProjects.cpp
Tools/GenerateProjects/GenerateProjects_VC120.vcxproj

Tools/GenerateProjects/GenerateProjects_VC120.vcxproj.filters
Tools/GenerateProjects/TemplateVC120.cpp
Tools/GenerateProjects/TemplateVC120.h
Tools/GenerateProjects/TemplateVC140.cpp
Tools/GenerateProjects/TemplateVC140.h

15 Updates to Version 5.14

May 27, 2017. Created projects and solutions for Microsoft Visual Studio 2017.

May 27, 2017. Increased the minor version number to 15.

LibApplications/makefile.wm5
LibCore/makefile.wm5
LibGraphics/makefile.wm5
LibImagics/makefile.wm5
LibMathematics/makefile.wm5
LibPhysics/makefile.wm5
LibCore/makeprj.wm5
LibGraphics/makeprj.wm5
LibImagics/makeprj.wm5
LibMathematics/makeprj.wm5
LibPhysics/makeprj.wm5
LibGraphics/Renderers/GlxRenderer/makerend.wm5

May 27, 2017. Fixed the project configurations. x64 were hooked up to x86 (probably wrong search-and-replace when porting VS2013 projects to VS2015).

SampleGraphics/Skinning/SkinningWgl_VC140.sln

May 27, 2017. Fixed two assertion(...) calls to take native string pointers rather than the std::string objects themselves.

(5.0.1) LibGraphics/Terrain/Wm5Terrain.cpp

May 27, 2017. Fixed bugs in the twist-swing decompositions.

(5.0.2) LibMathematics/Algebra/Wm5Quaternion.inl

16 Updates to Version 5.15

August 24, 2017. Increased the minor version number to 16.

LibApplications/makefile.wm5
LibCore/makefile.wm5

LibGraphics/makefile.wm5
LibImagics/makefile.wm5
LibMathematics/makefile.wm5
LibPhysics/makefile.wm5
LibCore/makeprj.wm5
LibGraphics/makeprj.wm5
LibImagics/makeprj.wm5
LibMathematics/makeprj.wm5
LibPhysics/makeprj.wm5
LibGraphics/Renderers/GlxRenderer/makerend.wm5

August 24, 2017. Several users requested templated versions of the float-specific algebra classes that were written for the graphics engine.

(5.0.7) LibMathematics/Wm5Mathematics.h
(5.16.0) LibMathematics/Algebra/Wm5TAPoint.h
(5.16.0) LibMathematics/Algebra/Wm5TAVector.h
(5.16.0) LibMathematics/Algebra/Wm5THMatrix.h
(5.16.0) LibMathematics/Algebra/Wm5THPlane.h
(5.16.0) LibMathematics/Algebra/Wm5THPoint.h
(5.16.0) LibMathematics/Algebra/Wm5THQuaternion.h

17 Updates to Version 5.16

October 12, 2017. Once again bit by the fact that Microsoft compilers do not complain when you access a base-class member in the templated derived class. g++ on Linux is not forgiving about this. Fixed the problem I introduced with the August 24, 2017 additions.

(5.16.1) LibMathematics/Algebra/Wm5TAPoint.h
(5.16.1) LibMathematics/Algebra/Wm5TAVector.h

The g++ compiler complained about violating strict antialiasing rules with the construct `*(int*)&value` where `value` is of type `float`. Fixed this by using a union of a `float` and an `int` type.

(5.0.1) LibMathematics/Base/Wm5BitHacks.inl

Increased the minor version number to 17.

LibApplications/makefile.wm5
LibCore/makefile.wm5
LibGraphics/makefile.wm5
LibImagics/makefile.wm5
LibMathematics/makefile.wm5
LibPhysics/makefile.wm5
LibCore/makeprj.wm5

LibGraphics/makeprj.wm5
LibImagics/makeprj.wm5
LibMathematics/makeprj.wm5
LibPhysics/makeprj.wm5
LibGraphics/Renderers/GlxRenderer/makerend.wm5