Subject: XForm3D library for U++
Posted by Oblivion on Sun, 20 Apr 2025 11:19:31 GMT
View Forum Message <> Reply to Message

Hi,

I hereby announce the immediate availability of Xform3D package.

What is Xform3d?

Xform3D is a software-based, lightweight 3D transformation library for U++, focused on providing essential matrix math and geometric tools for 3D graphics applications.
It includes a templated 4×4 matrix class and basic support for 3D point manipulation, with a design optimized for row-major, left-handed coordinate systems.

## Features

- Templated Matrix Class (`Matrix4_`): A generic 4×4 matrix supporting translation, scaling, rotation (axis-aligned), perspective, and frustum projection.

- Row-Major Layout with Left-Handed Coordinates:** Matches typical 3D rendering setups for clarity and compatibility.

- Basic 3D Geometry (`Point3_`, `Point4_`): Simple 3D point/vector classes with support for `Null` value handling and homogeneous coordinates.

## Examples

|:----------|:----------|
| `Teapot3D` | Example code rendering the famous Utah teapot model, using Xform3D and Painter. |

As a side note: I created this library to be the base of Upp::Sculptor, a Painter-based 3D rendering library I was planning to release. (It is in queue.)
Therefore it is best utilized by Painter. If you look at the example code, you'll see that the Painter integration can be amazingly simple (for simple stuff, of course).

I also plan to vectorize (simd/avx1-2) the matrix math.

Screenshot from Teapot3D:

Best regards,
Oblivion

---

Subject: Re: XForm3D library for U++
Posted by Didier on Sun, 20 Apr 2025 17:32:55 GMT
View Forum Message <> Reply to Message

I tried you're xform3D package the firts time you published it on the forum.

As for vectorisation, you can use clang options that help you rely on auto vectorisation: compile

---

Subject: Re: XForm3D library for U++
Posted by Oblivion on Sun, 27 Apr 2025 11:54:52 GMT
View Forum Message <> Reply to Message

Hello Didier,

Quote:As for vectorisation, you can use clang options that help you rely on auto vectorisation: compile messages are very useful to find out why a loop isn't vectorizef

Thank you, I'll certainly try your suggestion. :)

---

As for the library itself:

I've added a Box3_ template. It is an axis-aligned bounding box (AABB), though it can be used to render boxes too.
It has many useful and familiar methods (such as intersect/clamp/contains/inflate/deflate), all in familiar U++ fashion but for 3D. And has feature parity with the other components (Point3_/Point4_), such as serialization.
Also I added an extensive unit test for Box3_.

I am satisfied with the library's current condition, so aside from a couple of new functions or methods and bug fixes I consider this complete.
It is meant to be the foundation of higher level libraries or apps.

Best regards,