Hi,

The basic problem I had with the std was the assumption of the "worse" case classes that needed individual constructors on moves or expansion. The std allocator would get me the alignment solution but the rest of the class methods would drive performance into the mud.

Yet I wanted some of the runtime checks and protections that std::vector provided. So, although I could use a standard array, I would be consistently writing ASSERTS and Checks... that the code became too "hard to follow".

Hence I ended up a my own class that assumed deep copy, provided alignment and provided my basic requirements of a set of "vector" methods. (each method being added as I needed them.) The only real difference between Vector and my class was alignment awareness.

Ultimate++ assumptions about moveable was one of the "exciting" things about the framework. I really don't want to go back to using std and I would like to leverage Ultimate++ as much as possible.

AlexN