Radiance Refactoring Effort

Greg Ward
Taoning Wang

Refactoring Plan

- **Stage 1** - Encapsulate core rendering functionality in C++ classes
  - Support multi-threading for Windows platform
- **Stage 2** - Redesign scene language for spectra
  - Improve material programmability and support hyperspectral color
- **Stage 3** - Modernize rendering code with C++ classes
  - Continuous replacement of straight C code with C++

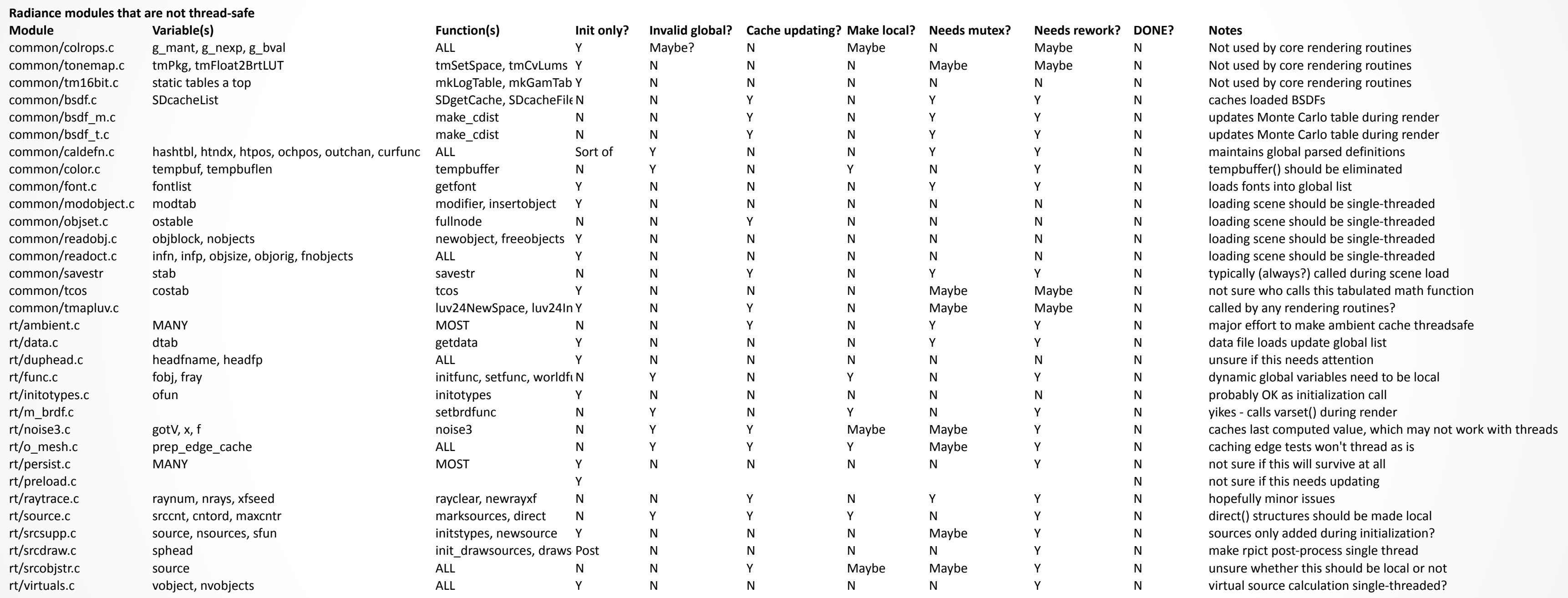
Stage 1 Status

- C++ base class for ray-tracing thread manager
  - Threads not currently enabled, still working out queuing mechanics
- Declared C++ classes for **rtrace**, **rpict**, and **rcontrib** equivalents
  - These classes will be called from replacement tools, but also made accessible as direct calls from integrator platforms
- Implemented and tested **rtrace** replacement (called "rxtrace" for now)
- Implementations of **rpict** and **rcontrib** functionality are more challenging
  - Will wait until threads are working

RtraceSimulManager Methods

- LoadOctree(const char *octn)
- SetThreadCount(int nt = 0)
- SetCookedCall(RayReportCall *cb, void *cd = NULL)
- SetTraceCall(RayReportCall *cb, void *cd = NULL)
- Ready()
- EnqueueBundle(const FVECT orig_direc[], n)
- FlushQueue()
- Cleanup(bool everything = false)

Multi-Threading: It’s Not All About Class



Radiance modules that are not thread-safe

Module	**Variable(s)**	**Function(s)**	**Init only?**	**Invalid global?**	**Cache updating?**	**Make local?**	**Needs mutex?**	**Needs rework?**	**DONE?**	**Notes**
common/colrops.c	g_mant, g_nexp, g_bval	ALL	Y	Maybe?	N	Maybe	N	Maybe	N	Not used by core rendering routines
common/tonemap.c	tmPkg, tmFloat2BrtLUT	tmSetSpace, tmCvLums	Y	N	N	N	Maybe	Maybe	N	Not used by core rendering routines
common/tm16bit.c	static tables a top	mkLogTable, mkGamTab	Y	N	N	N	N	N	N	Not used by core rendering routines
common/bsdf.c	SDcacheList	SDgetCache, SDcacheFile	N	N	Y	N	Y	Y	N	caches loaded BSDFs
common/bsdf_m.c		make_cdist	N	N	Y	N	Y	Y	N	updates Monte Carlo table during render
common/bsdf_t.c		make_cdist	N	N	Y	N	Y	Y	N	updates Monte Carlo table during render
common/caldefn.c	hashtbl, htndx, htpos, ochpos, outchan, curfunc	ALL	Sort of	Y	N	N	Y	Y	N	maintains global parsed definitions
common/color.c	tempbuf, tempbuflen	tempbuffer	N	Y	N	Y	N	Y	N	tempbuffer() should be eliminated
common/font.c	fontlist	getfont	Y	N	N	N	Y	Y	N	loads fonts into global list
common/modobject.c	modtab	modifier, insertobject	Y	N	N	N	N	N	N	loading scene should be single-threaded
common/objset.c	ostable	fullnode	N	N	Y	N	N	N	N	loading scene should be single-threaded
common/readobj.c	objblock, nobjects	newobject, freeobjects	Y	N	N	N	N	N	N	loading scene should be single-threaded
common/readoct.c	infn, infp, objsize, objorig, fnobjects	ALL	Y	N	N	N	N	N	N	loading scene should be single-threaded
common/savestr	stab	savestr	N	N	Y	N	Y	Y	N	typically (always?) called during scene load
common/tcos	costab	tcos	Y	N	N	N	Maybe	Maybe	N	not sure who calls this tabulated math function
common/tmapluv.c		luv24NewSpace, luv24In	Y	N	Y	N	Maybe	Maybe	N	called by any rendering routines?
rt/ambient.c	MANY	MOST	N	N	Y	N	Y	Y	N	major effort to make ambient cache threadsafe
rt/data.c	dtab	getdata	Y	N	N	N	Y	Y	N	data file loads update global list
rt/duphead.c	headfname, headfp	ALL	Y	N	N	N	N	N	N	unsure if this needs attention
rt/func.c	fobj, fray	initfunc, setfunc, worldfu	N	Y	N	Y	N	Y	N	dynamic global variables need to be local
rt/initotypes.c	ofun	initotypes	Y	N	N	N	N	N	N	probably OK as initialization call
rt/m_brdf.c		setbrdfunc	N	Y	N	Y	N	Y	N	yikes - calls varset() during render
rt/noise3.c	gotV, x, f	noise3	N	Y	Y	Maybe	Maybe	Y	N	caches last computed value, which may not work with threads
rt/o_mesh.c	prep_edge_cache	ALL	N	Y	Y	Y	Maybe	Y	N	caching edge tests won't thread as is
rt/persist.c	MANY	MOST	Y	N	N	N	N	Y	N	not sure if this will survive at all
rt/preload.c			Y						N	not sure if this needs updating
rt/raytrace.c	raynum, nrays, xfseed	rayclear, newrayxf	N	N	Y	N	Y	Y	N	hopefully minor issues
rt/source.c	srccnt, cntord, maxcntr	marksources, direct	N	Y	Y	Y	N	Y	N	direct() structures should be made local
rt/srcsupp.c	source, nsources, sfun	initstypes, newsource	Y	N	N	N	Maybe	Y	N	sources only added during initialization?
rt/srcdraw.c	sphead	init_drawsources, draws	Post	N	N	N	N	Y	N	make rpict post-process single thread
rt/srcobjstr.c	source	ALL	N	N	Y	Maybe	Maybe	Y	N	unsure whether this should be local or not
rt/virtuals.c	vobject, nvobjects	ALL	Y	N	N	N	N	Y	N	virtual source calculation single-threaded?

Spreadsheet listing functions that may require mutex or other thread protection

Next Steps

- Finish thread manager design
- Add mutexes and local variables to maintain consistency throughout code
  - This is the major headache of multi-threading, aside from debugging(!)
- Test multi-threading in **rxtrace**
- Implement *RpictSimulManager* class
  - Test functionality in rxpict replacement of **rpict**
- Implement *RcontribSimulManager* class
  - Test functionality in rxcontrib replacement of **rcontrib**
- Work with integrators to test new classes

Discussion