Open CASCADE Technology

Version 7.6.0

Release Notes

November 5, 2021

Overview

Open CASCADE Technology 7.6.0 provides more than 410 improvements and corrections over the previous release 7.5.0.

Highlights

General

- Dropped support of VS2008 (finally).
- Compilation on ARM64 (Apple M1).
- New interface Standard::StackTrace() for dumping stack trace.
- Removed a set of deprecated classes (TCollection, legacy Boolean operation API and others).

Modeling

- Numerous bug fixes and improved robustness of modeling algorithms.
- Added progress indication and user break support for Boolean operations, GeomPlate, DistShapeShape, Shape Offset.
- Added multi-thread mode of BRepCheck_Analyzer.
- Prohibition of scaled transformation within shape location.

Visualization

- Improved compatibility with WebGL and OpenGL ES.
- New interactive object AIS_LightSource for a light source.
- Support for shadow casting using shadow maps (without ray-tracing).
- Improved performance of rectangular selection.
- New selection scheme interface to AIS_InteractiveContext.
- New accurate order-independent transparency (OIT) option - depth peeling.
- OpenGL and OpenGL ES are no more mutually exclusive graphic drivers (TKOpenGl and TKOpenGles can be now built simultaneously).
- Ray-Tracing engine is now available within OpenGL ES 3.2.
- Regression testing of OpenGL ES graphic driver.
- More flexible configuration (Xlib, FreeType can be now excluded).

Mesh

- Store deflection calculated upon triangulation along with parameters passed to a mesher to Poly_Triangulation.

Data Exchange

- Kinematics entities can be read now from a STEP file.
- glTF import/export improvements and fixes for passing the validator.
- Support of KHR_draco_mesh_compression extension within glTF import.
- Introduced OSD_FileSystem for working with file streams.

Application Framework

- Improvements of XCAF document persistence (normal storage, length unit information, option to store in older format).
- Safe reading of independent OCAF documents in different threads.
- Partial loading of OCAF document and appending parts to document.
- Speed up methods of getting label by entry and vice versa.

IVtk

- Extraction of per-vertex surface normals for smooth shading.
- VTK9 compatibility fixes.

Draw Test Harness

- Tk is now an optional dependency (USE_TK in CMake).
- Support building DRAWEXE with statically linked plugins and as WebAssembly.
- Support multi-touch viewer gestures on Windows platform.

Samples

- Fixes for iOS sample building.

Documentation

- Added changelog for XCafBin and XCafXml storage formats.
- Added highlighting for code snippets throughout the documentation.

New Features

OCAF partial document loading

All kinds of binary document formats since the new version 12 are saved with the support of partial reading (sub-set of labels and sub-set of attributes). For that, the shapes data structures are stored with the related NamedShape attributes in the file, not in the particular section at the start of the document.

Size allocated for each label, each attribute, stored shapes data are stored in the file. This allows skipping big parts of the document in partial reading mode. The additional opening parameters could be defined in **PCDM_ReaderFilter**.

As a result, the new binary files become smaller, but default reading and writing of the whole document may take some more time (depending on the environment), up to 15 percent slower in the worst cases.

Backward compatibility (loading of old documents in the newer version) is still fully supported, as well as writing of the older versions of the document.

Added progress indication in modeling algorithms

The new improved mechanism of progress indication with user break functionality (introduced in version 7.5.0) has been added to the following modeling algorithms:

- Boolean operations and their family (general fuse, section, splitter, cells builder, volume maker, self-intersection checker, defeaturing)
- GeomPlate
- BRepExtrema_DistShapeShape
- BRepOffsetAPI_MakeOffsetShape

To enable this mechanism it is needed to pass the instance of Message_ProgressRange via the optional parameter in the constructor, Build or Perform method.

Added multi-thread mode of BRepCheck_Analyzer

Now it is possible to run the algorithm of checking shapes for validity in parallel mode, taking all the power of modern multi-core processors. To enable it just turn on the optional parameter theIsParallel of the algorithm constructor.

Prohibition of scaled transformation within shape location

The fact is that use of transformations with non-unit scale factor in shape locations (e.g. method TopoDS_Shape::Moved()) is not well supported in OCCT: in most cases (especially if underlying geometry is elementary) this will lead to invalid shape, and most algorithms will fail due to being not able to handle effect of scaling on parameterization. It is quite regular that users get confused by this possibility.

To prevent confusion, now the library throws the exception Standard_DomainError("Moving with scaling transformation is forbidden") when scaled or mirror transformation is used for shape.

However, in some cases, (for example, when the geometry is completely presented by B-splines), there is no error in the interpretation of shapes by algorithms. To enable scaled transformation in such cases the optional parameter theRaiseExc can be turned off. This possibility is left, and a developer can use it at his own risk.

If you have old shapes that may have scale or mirror transformations, you may use the new tools to overcome this problem.
BRepTools::CheckLocations. This method for a given shape returns the list of subshapes having problematic transformations.
BRepTools_PurgeLocations. This algorithm removes location datums, which satisfy problematic conditions, from all locations of the given shape and its sub-shapes.

OSD_FileSystem

OCCT 7.6.0 introduces a new interface **OSD_FileSystem** for working with file streams. Existing file operations creating std::ifstream and std::ofstream objects have been updated to use **OSD_FileSystem::DefaultFileSystem()** - a global object creating file streams from the given path via **OSD_FileSystem::OpenIStream()** and **OSD_FileSystem::OpenOStream()** methods.

Conventional interface for passing user-defined streams to algorithm exposes methods taking std::istream as an alternative to file path. While this covers a major part of scenarios, it has important limitations like passing more than one stream to handle external references supported by many file formats (STEP, glTF, JT, OBJ). It also requires considerable efforts for providing stream-aware API across most OCCT classes.

OSD_FileSystem is intended to solve these and several other problems with help of:

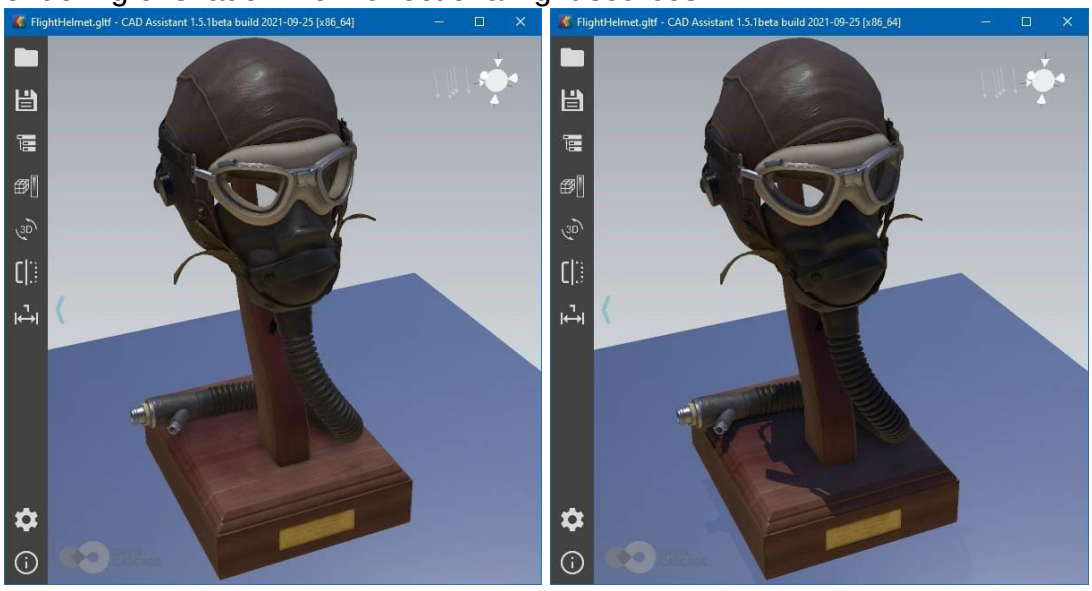
- **OSD_FileSystem::DefaultFileSystem()** global object allows avoiding modifications of existing algorithms by keeping file management operations local through universal file path interface.
- **OSD_FileSystemSelector** allowing to register application-provided file protocols - like remote URLs (http://), local network resources (smb://), resources embedded into application (content://), emulated in-memory file systems, distributed file systems (hdfs://) and others.
- **OSD_CachedFileSystem** implementing a local cache of a last opened file stream to avoid expensive reopening of files referred multiple times.

Visualization features

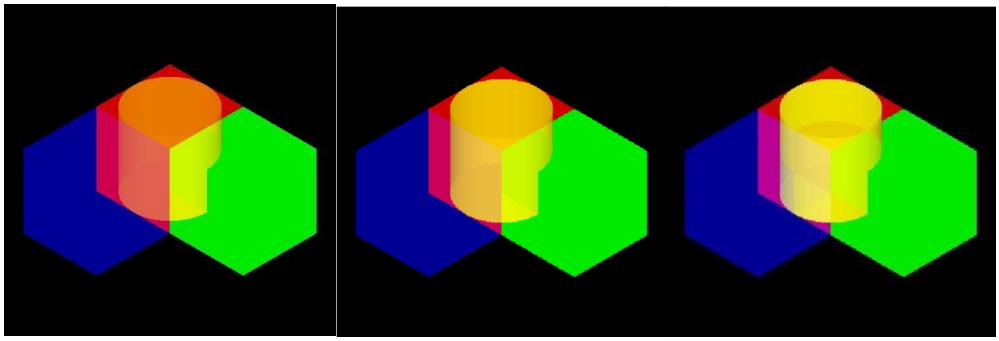
OCCT 3D Viewer continues to receive compatibility improvements and new features, including:

- Ray-Tracing (Path-Tracing) is now available on mobile Android devices supporting OpenGL ES 3.2.

- The conventional rendering engine (e.g., not using Ray-Tracing) now supports rendering of shadow from directional light sources.



- One more Order-Independent Transparency (OIT) option has been added - depth peeling. From left to right - unordered, weighted OIT, and depth peeling OIT.



- Added interactive object **AIS_LightSource** representing a light source supporting interactions like on/off action and synchronized source location transformations.