Google Summer of Code FreeCAD

# Project Proposal

**Issue No: #12843**

**Project title:** There is no 3D PDF export option

## Personal Information

| | |
|---|---|
| **Name** | Chirag Singh |
| **Email Address** | chirag.singh1711@gmail.com |
| **Forum Username** | @chiragsingh1711 |
| **GitHub Link** | https://github.com/chiragsingh1711/ |
| **Brief Background Info** | I have a proficient experience in **3D modelling, animation and scripting** with softwares like **Blender and FreeCAD**. I am also a **WebGL** and **XR Developer**. |
| **Link to resume** | https://drive.google.com/file/d/1wtsZeVJIfsoXqjELWPNkoxMZ-ndwHcct/view?usp=sharing |
| **Have I ever contributed in open source before?** | Yes, I was selected for Meta XROS'23 at Fanisko and GSoC'24 at BRL-CAD (ifcOpenShell) |

## Brief Project Summary (< 500 words)

The 3D PDF Export for FreeCAD project aims to develop a native capability for exporting interactive 3D models directly to PDF format, enhancing FreeCAD's documentation and collaboration capabilities. Currently, FreeCAD users must rely on complex multi-step workflows involving external tools like MeshLab, LaTeX, or even commercial CAD software to create 3D PDFs, often resulting in loss of model fidelity, colors, and structure.

This project proposes implementing a seamless 3D PDF export feature integrated directly into FreeCAD's TechDraw workbench. Users will be able to create standard technical drawings with traditional orthographic views and dimensions, plus an interactive 3D view that appears as a regular isometric view in print but becomes fully interactive when viewed digitally.

Key components of the implementation include:

1. **Document Structure Access**: Utilizing FreeCAD's Python API to traverse the document object hierarchy or need to rely on Qt for accessing certain structural information.
2. **3D Format Conversion**: Implementing exporters for both U3D format and PRC format to support NURBS surfaces for higher fidelity representation.
3. **PDF Generation**: Creating a PDF container that properly embeds the 3D content with interactive viewing capabilities.
4. **TechDraw Integration**: Extending the TechDraw workbench to include a new "3D View" type alongside traditional orthographic views.

The implementation will follow a phased approach, first developing a basic 3D PDF export capability using mesh conversion, then integrating with TechDraw, and finally exploring direct NURBS-to-PRC conversion for higher fidelity exports. This approach ensures deliverable functionality at each stage while progressing toward a comprehensive solution.

The completed feature will significantly enhance FreeCAD's usability for professional documentation, enabling users to share interactive 3D models with clients and collaborators who don't have CAD software, addressing a long-requested feature within the FreeCAD community.

Existing manual workflows that community uses:

link 1 - https://forum.freecad.org/viewtopic.php?t=48915
link 2 - https://forum.freecad.org/viewtopic.php?style=3&p=428909
link 3 - https://forum.freecad.org/viewtopic.php?style=5&p=431875

# Detailed Project Description (> 500 words)

## Phase 1: U3D Export Implementation

**Objective**:

Implement a direct U3D format exporter in FreeCAD to enable 3D PDF export without requiring external tools.

**Approach**:
We'll leverage the U3D exporter implementation from KiCad, which has proven that direct U3D generation is feasible without external dependencies. The KiCad developer has created a clean implementation that converts OpenCASCADE geometry directly to U3D format, offering their code for potential reuse in FreeCAD enabling interactive viewing of 3D models within PDF readers. This allows users to rotate, pan, and zoom 3D models embedded in documentation without requiring specialized CAD software.

The implementation will involve three primary components:

1. Geometric data extraction from FreeCAD's internal representation (Reference - https://singerlinks.com/2023/11/how-to-iterate-the-freecad-document-tree-using-python/)
2. Conversion of hierarchical model structure to U3D format
3. Binary encoding according to ECMA-363 specifications with proper compression

**Reference Code**:

https://gitlab.com/mroszko/kicad/-/commit/20b1e652eae2fcd5033c321a2060701c80b5501e?view=inline

https://gitlab.com/mroszko/kicad/-/tree/20b1e652eae2fcd5033c321a2060701c80b5501e/pcbnew/exporters/u3d

## Phase 2: PRC Export Implementation

**Objective**:

Implement PRC format export to provide better representation of CAD models.

**Approach**:

While U3D is easier to implement due to its free specification, PRC format offers significant advantages for CAD data representation. We'll implement a PRC exporter using mesh conversion initially, with plans to extend to direct NURBS conversion later.

The Product Representation Compact (PRC) format, defined by ISO 14739-1:2014, was specifically designed for high-fidelity representation of 3D CAD data. Unlike U3D, which focuses on visualization, PRC preserves the exact mathematical representation of 3D models including precise NURBS surfaces, B-rep solids, and PMI (Product Manufacturing Information). This preservation of engineering data makes PRC the preferred format for technical documentation where dimensional accuracy is crucial.

**Why PRC is better than U3D?**

| Feature | PRC | U3D | Importance for FreeCAD |
|---|---|---|---|
| NURBS Support | Yes | No | Critical for preserving exact CAD geometry |
| File Size | Smaller | Larger | Important for complex models |
| CAD Data Fidelity | High | Low | Essential for engineering use cases |
| Compression | Better | Basic | Helps with large assemblies |
| B-rep Support | Yes | No | Maintains precise edges and surfaces |
| Color/Material | Per face | Limited | Preserves design intent |
| Standard | ISO 14739-1:2014 | ECMA-363 | Industry adoption |

My initial implementation will focus on mesh-based PRC export since this provides a more straightforward path to integration while still offering advantages over U3D. In future iterations, we plan to implement direct B-rep conversion to fully leverage PRC's capabilities for preserving exact CAD geometry.

References –

https://www.youtube.com/watch?v=Q_ufaCN2hb4

## Phase 3: 3D PDF Export

**Objective**:

Create a PDF with embedded PRC content for higher-fidelity 3D representation.

**Approach**:

We'll extend the PDF generation approach from Phase 2 to use PRC instead of U3D, leveraging libHaru's PRC support. This implementation requires careful attention to the technical differences between U3D and PRC embedding in PDF documents.

The PDF architecture for PRC content utilizes specialized PDF objects and streams that differ from U3D embedding.

The implementation will prioritize compatibility with Adobe Reader (which has the most comprehensive PRC support) while ensuring reasonable functionality in other PDF readers with 3D capabilities.

**Code:**

```
#include "oPrcFile.h"

#include "hpdf.h"

#pragma comment(lib, "libprc.lib")
#pragma comment(lib, "libhpdfd.lib")

void testPrc(void)
{
    prc::oPRCFile aPrcFile("test.prc");

    prc::PRCmaterial materialGreen(
        prc::RGBAColour(0.0,0.18,0.0),
        prc::RGBAColour(0.0,0.878431,0.0),
        prc::RGBAColour(0.0,0.32,0.0),
        prc::RGBAColour(0.0,0.072,0.0),
        1.0,0.1);

    // Sphere
    aPrcFile.begingroup("Sphere");
    aPrcFile.addSphere(1.0, materialGreen);
    aPrcFile.endgroup();

    aPrcFile.addCylinder(0.8, 6.0, materialGreen);

    aPrcFile.finish();

    // embed prc file to pdf.
    HPDF_Doc aPdfDoc = HPDF_New (NULL, NULL);
    if (!aPdfDoc)
    {
        printf("error: cannot create PdfDoc object");
        return;
    }

    HPDF_Rect aRect = {0, 0, 800, 800};
    HPDF_Page aPage = HPDF_AddPage(aPdfDoc);
    HPDF_Page_SetWidth(aPage, aRect.right);
    HPDF_Page_SetHeight(aPage,aRect.top);

    HPDF_U3D aU3D = HPDF_LoadU3DFromFile(aPdfDoc, "test.prc");
    HPDF_Annotation aAnnot = HPDF_Page_Create3DAnnot(aPage, aRect, HPDF_TRUE,
HPDF_FALSE, aU3D, NULL);

    HPDF_SaveToFile(aPdfDoc, "test.pdf");
    HPDF_Free(aPdfDoc);
}

int main(int argc, char* argv[])
{
    testPrc();

    return 0;
}
```

**Output PDF:**

https://drive.google.com/file/d/1pjQ1rA6djgocESFS0htIiO6fLjyqKZyr/view?usp=drive_link

## Phase 4: TechDraw Integration

**Objective**: Integrate 3D PDF export with FreeCAD's TechDraw workbench to enable customizable technical documentation.

**Approach**:
Based on community feedback, particularly user @serchu's suggestion about editable formats similar to TechDraw's SVG templates, we'll extend TechDraw to include 3D views that can be embedded in technical drawings and exported as interactive PDFs.

TechDraw is FreeCAD's dedicated technical drawing workbench, providing tools for creating traditional engineering drawings with multiple views, dimensions, annotations, and section cuts. The workbench already supports sophisticated template-based documentation through SVG templates, making it an ideal foundation for 3D PDF integration.

The architectural integration will involve several components:

1. **TechDraw3DView Class**: A new view type derived from TechDraw::DrawView that represents an interactive 3D view within a drawing
    - Properties for view direction, perspective settings, rendering style
    - Configuration options for lighting, background, and initial view state
    - Support for user-defined clipping planes and cross-sections
2. **View Placement System**: Integration with TechDraw's existing view placement mechanisms
    - Drag-and-drop positioning within templates
    - Alignment tools for precise layout
    - Size adjustment with aspect ratio preservation
3. **Visual Style Control**: Options for controlling the appearance of 3D views
    - Rendering mode (shaded, wireframe, hidden line, etc.)

    - Edge visibility and emphasis settings
    - Background configuration (transparent, gradient, solid)
    - Optional bounding box display

4. **Export Pipeline**: Integration with TechDraw's existing export functionality
    - PDF generation with embedded 3D views
    - Mixed-content documents with traditional orthographic views alongside interactive 3D

The implementation will carefully integrate with TechDraw's existing architecture, leveraging its Template system, page layout mechanisms, and dimension tools. This approach creates a unified documentation workflow where traditional 2D drawings and interactive 3D models coexist in a single document, providing the best of both worlds for technical communication.

Particular attention will be paid to the transition between screen display (where 3D views may be represented as static images) and PDF export (where those views become interactive). This ensures a consistent user experience and predictable results in the final documentation.

## Potential Challenges Identified

Based on the search results and the feedback from @keithloan52, here are the potential challenges identified for implementing 3D PDF export in FreeCAD:

1. FreeCAD Structure Preservation:
    - Limitations in the FreeCAD API for preserving document structure hierarchy during export
    - Potential need to rely on Qt for accessing certain structural information
    - Challenge: Ensuring the exported 3D PDF accurately represents the FreeCAD model's structure
    - Reference - https://singerlinks.com/2023/11/how-to-iterate-the-freecad-document-tree-using-python/
2. Global Positioning Issues:
    - Difficulty in obtaining global position information for objects, especially linked objects
    - The getGlobalPlacement() method doesn't fully support Links
    - Challenge: Accurately preserving object positions in the exported 3D PDF, particularly for complex assemblies
3. PRC Format Implementation:
    - The PRC format specification (ISO 14739-1:2014) is not freely available, which may complicate implementation

- Challenge: Accurately implementing PRC export without direct access to the full specification

## Development Schedule

**Total Hours: 350 hours**

| Week | Topic | Description |
|---|---|---|
| **Week 1-2 (30 hours)** | **Project Setup and Research** | ▪ Review project requirements and engage with FreeCAD community<br>▪ Set up development environment with necessary dependencies<br>▪ Research U3D and PRC formats in depth<br>▪ Study KiCad's U3D implementation for potential code reuse<br>▪ Review FreeCAD's API and TechDraw architecture |
| **Week 3-4 (40 hours)** | **U3D Export Implementation** | ▪ Develop U3D exporter class<br>▪ Implement mesh data extraction from FreeCAD objects<br>▪ Create binary U3D file writer according to ECMA-363 specification<br>▪ Test with various FreeCAD models including complex assemblies |
| **Week 5-6 (40 hours)** | **Basic 3D PDF Export** | ▪ Integrate libHaru PDF library with appropriate bindings<br>▪ Implement PDF generation with embedded U3D content<br>▪ Create configurable view parameters and camera settings<br>▪ Develop simple export interface for FreeCAD users<br>▪ Test PDF compatibility across different viewers |
| **Week 7-8 (50 hours)** | **PRC Export Implementation** | ▪ Research PRC format requirements and ISO standards<br>▪ Implement mesh-based PRC exporter<br>▪ Add support for colors, materials, and transparency<br>▪ Implement model hierarchy preservation<br>▪ Test PRC export with various model complexity levels |
| **Week 9-10 (30 hours)** | **PRC-based 3D PDF Export** | ▪ Extend PDF generation to support PRC format<br>▪ Implement enhanced CAD-specific view settings<br>▪ Add support for model tree and structure in PDF<br>▪ Optimize lighting and rendering styles for technical documents<br>▪ Compare with U3D results and document advantages/limitations |
| **Week 11-12 (30 hours)** | **TechDraw Integration - Part 1** | ▪ Create new TechDraw3DView class<br>▪ Implement view placement in TechDraw sheets<br>▪ Develop property panel for 3D view configuration<br>▪ Create view direction and perspective controls<br>▪ Test integration with existing TechDraw functionality |

| | | |
|---|---|---|
| **Week 13-14 (40 hours)** | **TechDraw Integration - Part 2** | ▪ Implement PDF export from TechDraw with embedded 3D views<br>▪ Add support for multiple 3D views in a single document<br>▪ Develop mixed-content generation (2D orthographic + 3D views)<br>▪ Create preview functionality for 3D views<br>▪ Ensure proper integration with dimension tools and annotations |
| **Week 15-16 (50 hours)** | **Template System and Finalization** | ▪ Implement XML-based template definition system<br>▪ Create standard templates for engineering drawings, presentations<br>▪ Develop template editor interface<br>▪ Create comprehensive documentation and tutorials<br>▪ Final testing and performance optimization |
| **Buffer Time (20 hours)**<br>Allocate 20 hours for unforeseen issues, additional testing, and final adjustments. | | |

## Time Availability

During the contribution period, daily I'll be able to put in **5-6 hours/day** (40 hours a week) working remotely from Dehradun (UK), India.

12:00 pm to 6:00 pm **IST – 6 hours**

10:00 pm to 2:00 am **IST – 4 hours**

## Why FreeCAD?

The 3D PDF export project specifically addresses a critical gap in FreeCAD's functionality that users have been requesting for years. By implementing this feature, I can make a meaningful contribution that enhances collaboration between FreeCAD users and stakeholders who may not have access to specialized CAD software.

## Why I am the best person for this Project?

As my profile closely matches the tech stack used in FreeCAD—particularly my expertise in CAD softwares, Python, C++ and 3D rendering. My skills align seamlessly with the project's needs, making it an ideal opportunity for me to leverage my existing knowledge while expanding my expertise in a challenging and impactful open-source environment.

## Related work?

Yes, I have worked on a similar project when I was contributing to Bonsai (Previously BRL-CAD) at ifcOpenShell where I was responsible to add a completely new rendering system called Radiance to the Blender add-0n.

I also have tons of experience with using Blender and CAD softwares for modelling, animating and scripting.

Here's my Blender Portfolio -
https://drive.google.com/drive/folders/1WP7O56nRw86tea6WZRFSCRQDhPggMkG3?usp=sharing