IEEE SA
STANDARDS ASSOCIATION

INDUSTRY CONNECTIONS
WHITE PAPER

FILE FORMAT RECOMMENDATIONS FOR 3D BODY MODEL PROCESSING

Authored by

Maxim Fedyukov, Texel

With contributions from

Michael Stahl, Intel
Andrew Grant, Autodesk
Alfredo Ballester, Instituto de Biomecánica de Valencia
Carol McDonald, Gneiss Concept
Andrey Golub, ELSE Corp.
Andrey Poskonin, Texel
Ilya Petrov, Texel
Randy Rannow, Silverdraft
Yannick Francken, 3D Body Cloud

IEEE

3 Park Avenue | New York, NY 10016-5997 | USA

IEEE Industry Connections (IEEE-IC) File Format Recommendations for 3D Body Model Processing

Authors:

Maxim Fedyukov, Texel

Core Contributors:

Michael Stahl, Intel
Andrew Grant, Autodesk
Alfredo Ballester, Instituto de Biomecánica de Valencia
Carol McDonald, Gneiss Concept
Andrey Golub, ELSE Corp.
Andrey Poskonin, Texel
Ilya Petrov, Texel
Randy Rannow, Silverdraft
Yannick Francken, 3D Body Cloud

Trademarks and Disclaimers

IEEE believes the information in this publication is accurate as of its publication date; such information is subject to change without notice. IEEE is not responsible for any inadvertent errors.

The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

PDF: ISBN 978-1-5044-6179-5 STDVA23902

IEEE prohibits discrimination, harassment, and bullying. For more information, visit http://www.ieee.org/web/aboutus/whatis/policies/p9-26.html.

To order IEEE Press Publications, call 1-800-678-IEEE.
Find IEEE standards and standards-related product listings at: http://standards.ieee.org

Notice and Disclaimer of Liability
Concerning the Use of IEEE SA Industry Connections Documents

Contents

IEEE Industry Connections (IEEE-IC) File Format Recommendations for 3D Body Model Processing

Abstract

This is the just one in a series of white papers published by the IEEE 3D Body Processing Initiative (3DBP). The main areas of interest of IEEE 3DBP are apparel, athletics, health, and wellness industry applications.

The first white paper[1] introduced the background, goals, and status for the IEEE 3D Body Processing standard, which is under development. Another white paper[2] reviews existing standards relevant to the 3DBP initiative and provides a minimal set of landmarks and measurements needed for 3D body processing. These and other white papers are available at https://standards.ieee.org/develop/indconn/3d/bodyprocessing.html.

Yearly updates of the IEEE 3DBP are available from 3D Body Tech Conference Proceedings. https://www.3dbody.tech/cap/.

This white paper is dedicated to file formats, covering the surveys conducted by 3DBP group, determining industry needs, and providing file format recommendations to facilitate the 3D data processing. The requirements for 3D body models are thoroughly considered, five file formats are compared, two of them that meet the requirements (X3D and glTF) are explored, and the recommendations on representation options are given as well.

1. Introduction

The goal of the effort is to standardize file formats used for 3D body model processing.

This white paper covers only 3D body representation. It does not cover the representation of other entities, for example, garment or footwear representations.

Ideally, it would be perfect to have a match between the set of features needed for 3D body models and the set of features that are present in an existing format, such that both these sets coincide to the greatest extent possible.

If there is no good match between these two sets for a particular file format, two problems arise: if the needed features are not supported in a chosen file format, the 3D body-related applications get limited; and if a chosen file format has features that are not needed for 3D body processing, it becomes a source of complexity and incompatibility. The incompatibility may arise from advanced features in use by one company but not by others; or if a file format provides

[1] 3D Body Processing (3DBP) Initiative–An Introduction.
[2] Landmarks and Measurement Standards Comparison in 3D Body-model Processing.

different methods to achieve the same result. Without getting into complex details of scene or skeleton representations, consider the simple case of a 3D model color representation. Assume a file format allows for defining color for a 3D model with texture, vertex colors, or face colors. This may result in files that have both texture and vertex colors assigned. Different software may treat this case differently.

Two possible solutions may be as follows:

a) To develop a new file format, or

b) To find existing file formats that support the needed features and to standardize the use of the features to avoid incompatibility.

There are pros and cons for both solutions.

For example, considering the evolution of both existing file formats and the 3D body processing industries' growing needs, suitable file formats found today may become a bad match for 3D body model processing in the future.

Nevertheless, considering the efforts needed to develop and maintain a new file format, solution b) would be the more reasonable choice. This approach is also in line with the position already stated by IEEE 3DBP group in a previous white paper:

> - The 3DBP initiative guidelines for recommending a file format are as follows:
>   - Do not invent yet another format. Find existing format(s) that provide the functionality listed below.
>   - Support embedding of metadata as part of the file.
>   - Support mandatory metadata (defined field names and defined enumeration data when applicable; must not be NULL) and optional metadata (defined field names and defined enumeration data when applicable; may be NULL or may not exist at all).
>   - Support vendor-specific metadata (define structure but nothing else).
>   - Support file authentication.
>   - Support encryption of parts or all of the file.
> - Using standard formats to deliver 3D body assets will improve interoperability and decreases friction across different software and use cases. Including metadata that pertains to quality attributes and creation methods can aid in giving a better understanding of what the file contains, how the body model was produced, and how it can be used.

Therefore, members of 3D Body Processing Initiative have agreed that finding suitable file formats for 3D body processing needs to be addressed.

One of the key problems of 3D model file formats is as follows.

There are extremely simple old file formats that have gained widespread adoption. Thanks to extreme simplicity, they are quick to implement in any language. The old file formats that are no longer under development have stable and static versions. This implies that the file format version tracking is not required. The file formats that are under development would require version tracking. This may impact file compatibility between versions and as such may impact the interoperability between versions.

In contrast, newer advanced formats have a rich set of features and extensibility, but they are not in wide use.

In summary, one can choose from the following:

1) Simple old file format, like OBJ
2) New advanced file formats, like X3D or glTF
3) Something in-between 1) and 2): not that advanced and extensible, but widespread, like VRML or FBX
4) The recommendation to use different file formats for different use cases
5) The recommendation to use one file format now (widespread, but not very advanced), and the recommendation to use another file format in the future (when new widespread formats appear)

In case 1) a number of features need to be rejected (e.g., there is no vertex color support in an OBJ specification). If necessary and sufficient attributes are not contained in a single feature format when multiple features are required, then the result could lead to format packages as shown in the example below. Consequently, it will be necessary to include additional files (in other standard formats whenever possible) that were not supported in the original single feature format. For example,

```
model012345.zip (optionally encrypted)
    model012345.obj
    model012345.mtl (optional)
    model012345.jpg (optional)
    model012345.xml (for metadata, optional)
    model012345.bvh (for animation or body pose changing, optional)
    model012345.dcm (for corresponding DICOM data from medical imaging devices, optional)
    model012345.sig (for signature of the listed files, optional)
```

Here it should be noted that some use cases will require extensive use of metadata. In these cases, `model012345.xml` will in fact become a partial implementation of some XML 3D model file format, like X3D or Collada.

In case 2), to avoid a stillborn standard that is good on paper but requires excessive resources to be implemented, the following must be provided:

a) Open-source implementations of import/export in a number of programming languages.

b) Good step-by-step documentation for import/export integration with common use case examples (code for mesh import/export, code for animation import/export, code for metadata import/export and so on).

To have the ability to work with these files, not only in the 3D body processing tool chain, but also in external software, the following would be needed:

c) Implementation of import/export for popular software that supports plugins (Blender, 3ds Max, Maya, Unity, and others).

d) Implementation of file conversion tools for exporting to other software that does not support plugins (a rarity in today's environment).

Both file formats recommended in this white paper (glTF and X3D) meet these requirements quite well.

This paper is structured as follows:

- A survey was conducted to determine the file format features needed for 3D body models. The results of this survey are given in Section 2.
- An extensive set of existing 3D model file formats was investigated. Seven file formats were short-listed. Comparisons made according to the surveyed needs is given in Section 3.
- Regarding feature-rich file formats, compatibility limitations may need to be set for the future IEEE P3141, Draft Standard for 3D Body Processing. Therefore, a second survey was conducted. The results of the survey on model representation are given in Section 4.
- Recommendations on representation options to be included in future IEEE P3141, Draft Standard for 3D Body Processing are discussed in Section 5.
- Section 6 outlines the conclusions and future work.

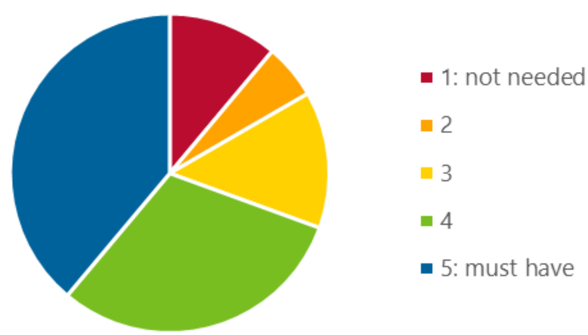
2. Format features needed for human body 3D models

To survey the suitability of existing file formats for human body 3D models, an extensive questionnaire was developed. The survey contained 25 questions on what features must be supported by a file format. Thirty-seven people representing the following organizations answered:

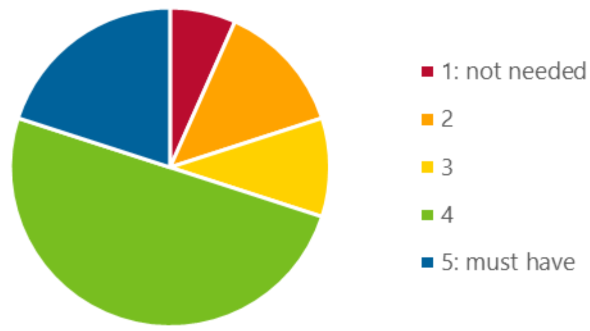
- 3Daboutme
- 3dMD LLC
- AIST
- Avametric / UC Berkeley
- Bauerfeind AG
- Body2Garment Solutions
- Browzwear
- Elasizer
- ELSE Corp
- Gneiss Concept
- Instituto de Biomecánica de Valencia
- Intel
- Metail Ltd.
- Novaptus Systems, Inc.
- NSRDEC, Anthropometry Team
- Picanova (3D.me)
- Polytechnic University of Tirana, Faculty of Mechanical Engineering, Textile & FashionDepartment
- QuantaCorp
- Size Stream
- Target
- Texel
- True Fit
- TU Delft
- University of Michigan
- Web3D Consortium
- ZelusFX

Results of the survey on format features needed for human body 3D models

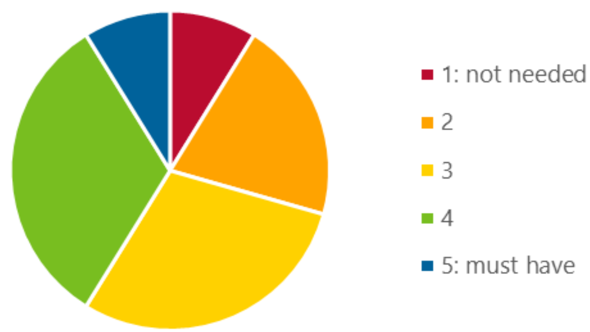
- Q1: File format supports color data, represented as texture



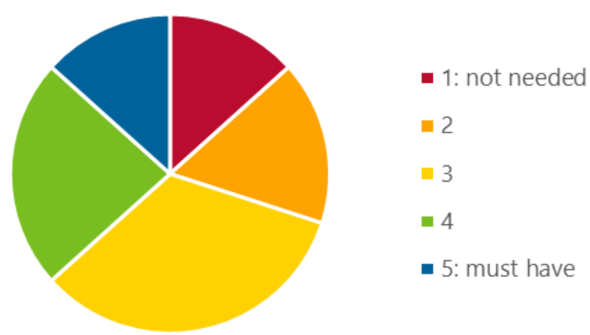
- Q2: File format supports color data, represented as vertex colors



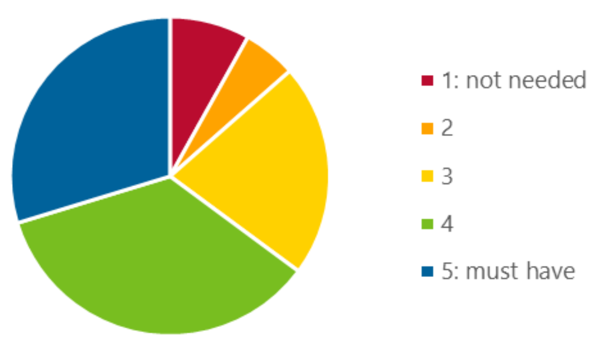
- Q3: File format supports color data, represented as face (polygon) colors



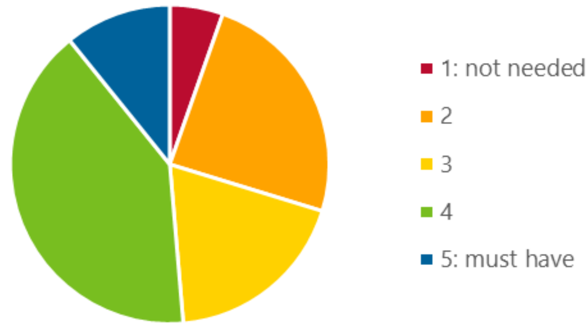
- Q4: File format supports multiple maps (not only texture map, but also, for example, scanner noise map, hole filling map, etc.)



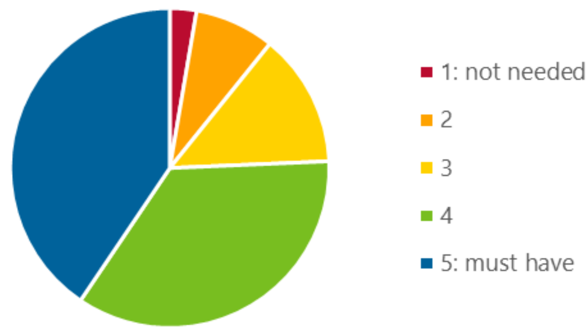
- Q5: File format supports storing everything in one file (including textures)



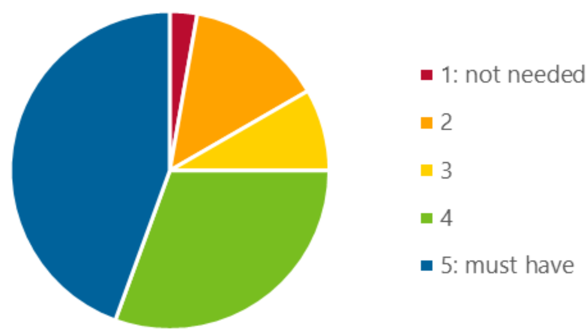
- Q6: Small file size (file format native support of efficient data compression, instead of simple storing of body model file(s) in ZIP-archive)



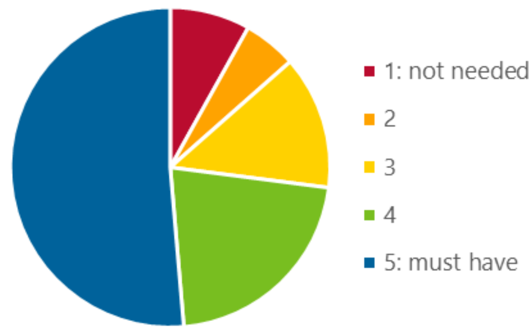
- Q7: File format native support for global model metadata, both predefined and custom, like model units, coordinate system, certainty values for model overall accuracy, etc. (instead of storing metadata in sidecar file, i.e., in .XMP)



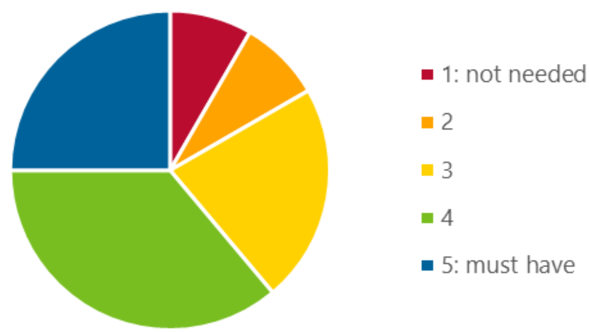
- Q8: File format native support for local body model metadata (like body landmarks, certainty values for landmarks placement and measurements values, and other metadata defined by 3DBP Metadata Sub-team)



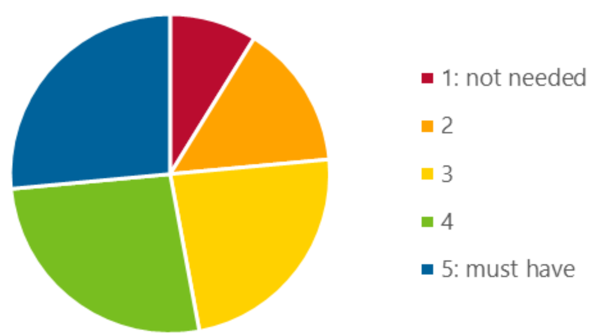
- Q9: Extensibility (easy and native way to add extra data: custom vendor data, or some new data accepted in the future versions of the standard)



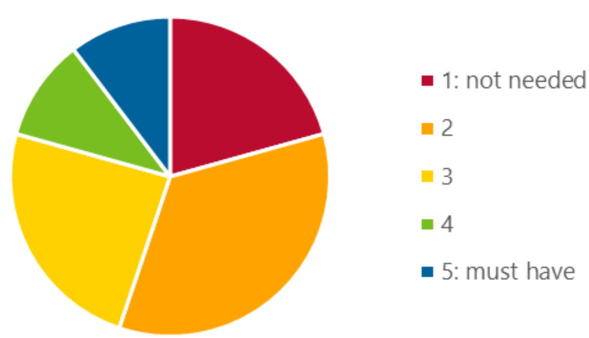
- Q10: Support for storing measurements as a simple table (Measurement name, Value)



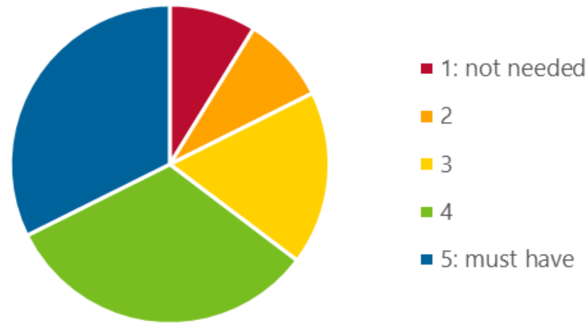
- Q11: Support for storing measurements not just as a simple table, but with curves on the body model surface, along which these measurements were made



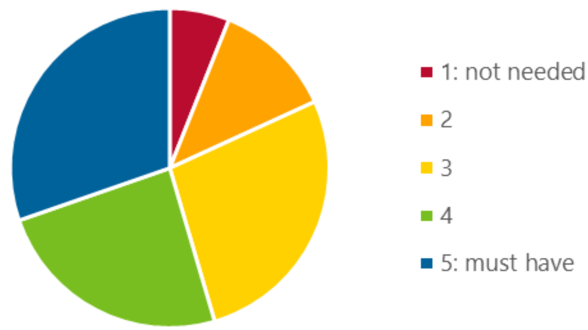
- Q12: File format native support for corresponding DICOM data from medical imaging devices (instead of storing metadata in sidecar file, i.e., in .DCM)



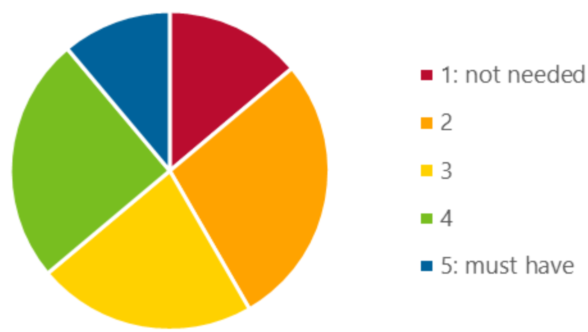
- Q13: Skeletal animation support for body animation and pose changing (to A-, T-, Y-pose or any other pose)



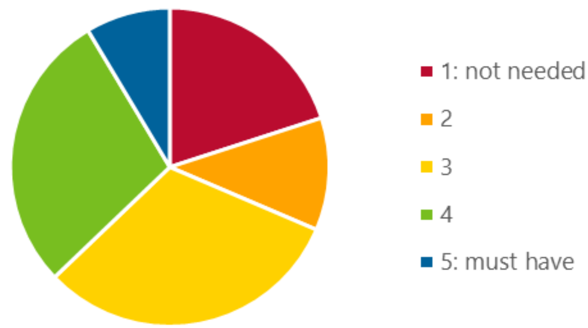
- Q14: Morph targets (a.k.a. blend shapes or shape keys) support: for face animation, body shape parameters (like “Left Wrist Circumference” or “Waist Height”) or other body model changes



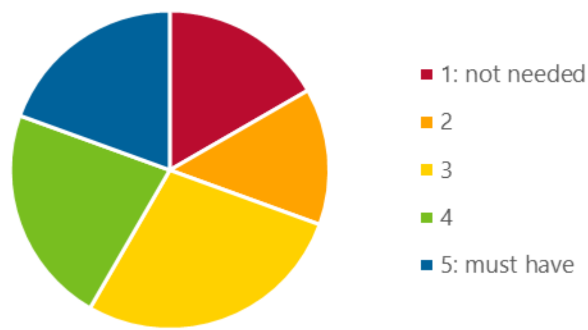
- Q15: File format is optimized for rendering (no need to convert data structures stored in the file before rendering)



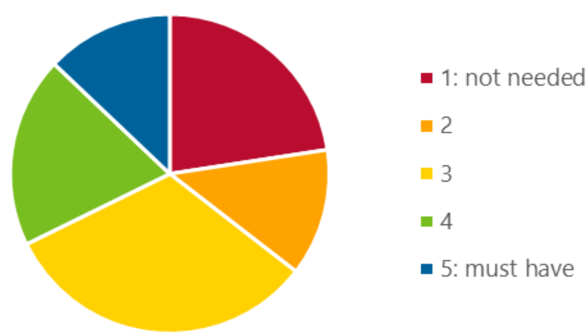
- Q16: Native whole file encryption (instead of packing it to encrypted ZIP-archive or similar solutions)



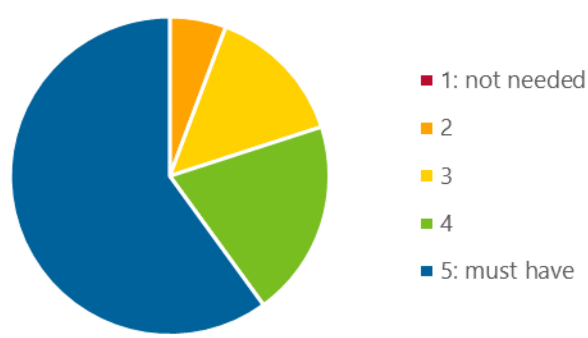
- Q17: Native encryption of parts of the file, i.e., vendor specific metadata or attributes (instead of storing them in external encrypted file)



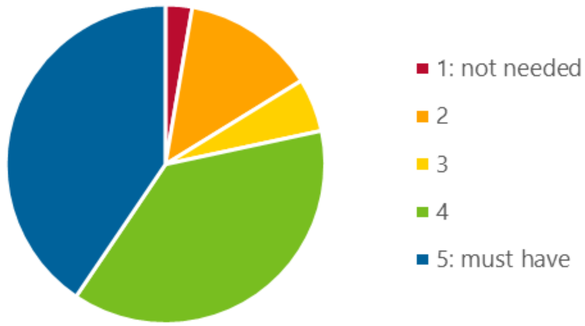
- Q18: Native document signing support (instead of storing digital signature in external sidecar file)



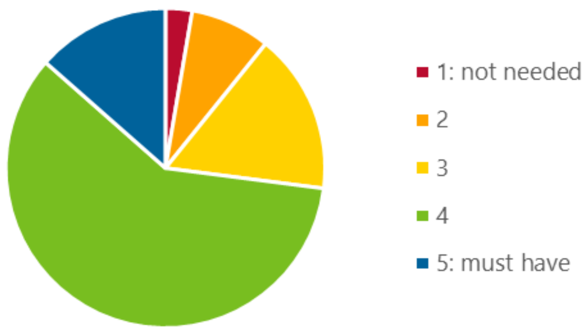
- Q19: Open specification of file format



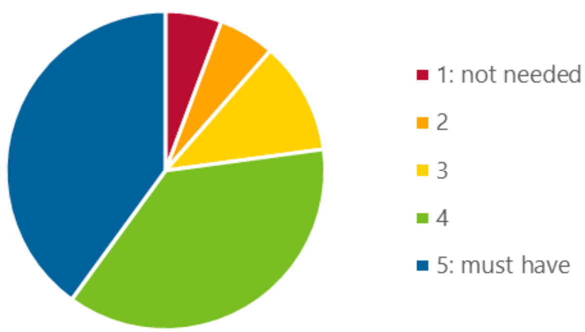
- Q20: There is ready-to-use open-source implementation of import/export at least in one popular programming language



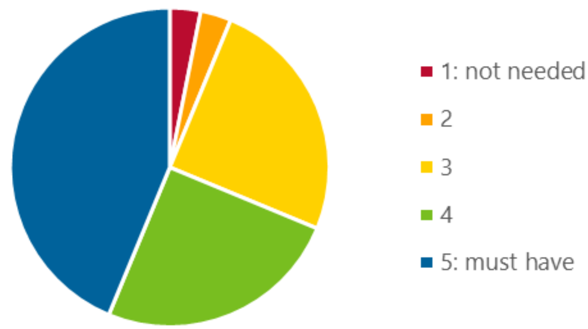
- Q21: There are ready-to-use open-source implementations of import/export in a number of popular programming languages (available for integration in applications in C++, C#, Visual Basic, Java, Python, ...)



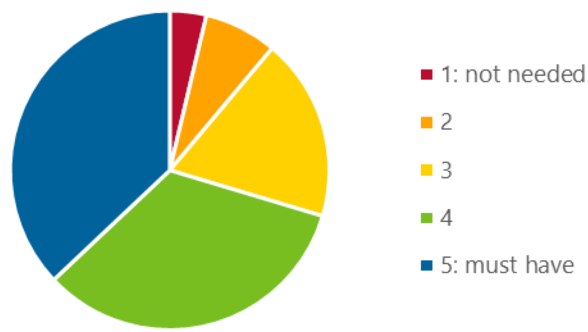
- Q22: There is good step-by-step documentation for import/export integration with common use case examples (code for mesh import/export, code for animation import/export, code for metadata import/export and so on)



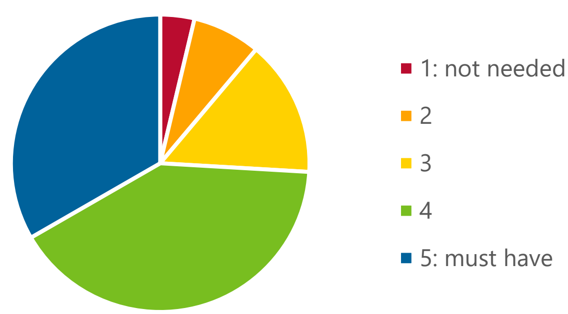
- Q23: File format is not dead, but is in active development (with ongoing adoption of new features to the file format specification, like advanced character animation, support for 3D models from CT, MRI, X-Ray scans, etc.)



- Q24: The ability to inspect the file with a free simple viewer (compiled for major desktop OSes: Windows, Linux and MacOS)



- Q25: The ability not only to view but also to edit and work with the file in popular software (either with a native support or with the help of the plugins)



3. File format comparison

There are many 3D model file formats available today, as listed herein, for example:

- https://en.wikipedia.org/wiki/List_of_file_formats#3D_graphics
- http://edutechwiki.unige.ch/en/3D_file_format
- https://help.sketchfab.com/hc/en-us/articles/202508396-3D-File-Formats
- https://en.wikipedia.org/wiki/Category:3D_graphics_file_formats

The following formats were explored in more detail: 3DS, 3DXML, 3MF, BLEND, COLLADA, DSON, DWG, DXF, E57, FBX, glTF, IGES, JT, LWO, MD2, MD3, OBJ, OFF, OpenCTM, OpenGEX, PLY, PRC, STEP, STL, U3D, VRML 1.0, VRML 2.0, X, X3D, XSI, and XVL.

Among these formats, five advanced were short-listed as suitable for 3D body model processing needs, according to the survey results described in Section 2: FBX, X3D (ISO/IEC 19775/19776/19777, successor of VRML), glTF (released in 2015 by Khronos Group, developer of COLLADA), BLEND (Blender native format), and DSON (Daz Studio native format). FBX, X3D, glTF and BLEND are used by the members of IEEE 3DBP in their pipelines currently. DSON was chosen for comparison because it is a format that contains powerful features for human models as well as being an open format.

A thorough analysis of all these formats was conducted by the members of 3D Body Processing Initiative and invited experts from Autodesk, Daz Productions, Khronos Group and the X3D Working Group.

A comparative table has the same 25 questions of the survey on format features needed for human body 3D models (described in Section 2). The full table is given below. For reference, the results of the survey that are provided in Section 2 as pie charts are given in the Table 1 again as numerical values in columns 2–6. The table shows whether or not the file format has the listed feature.

According to the comparisons, the formats most suitable for human body 3D models are X3D and glTF. These two formats are recommended as those having all features needed for 3D body models: see color coding in the table for quick evaluation (green is for “Yes”, red is for “No”, and yellow is for the options in-between). See descriptions and reference links in the cells for details.

Table 1: Comparison of the advanced file formats suitable for human body 3D models.

	FBX	X3D (ISO/IEC 19775/19776/19777)	glTF	BLEND (Blender native format)	DSON (Daz Studio native format)
Q1: File format supports color data, represented as texture	Yes (https://help.autodesk.com/view/FBX/2018/ENU/?guid=GUID-8A69CD20-B8FB-4B23-925A-372E408390AA)	Yes (http://www.web3d.org/documents/specifications/19775-1/V3.3/Part01/components/texturing.html)	Yes (https://github.com/KhronosGroup/glTF/blob/master/specification/2.0/README.md#textures)	Yes (https://docs.blender.org/manual/en/dev/editors/uv_image/uv_editing/applying_image.html?highlight=texture)	Yes
Q2: File format supports color data, represented as vertex colors	Yes, can be baked though no color management	Yes, defined by colorPerVertex (http://www.web3d.org/documents/specifications/19775-1/V3.3/Part01/components/rendering.html#X3DComposedGeometryNode)	Yes, with COLOR_0 attribute (https://github.com/KhronosGroup/glTF/blob/master/specification/2.0/README.md#meshes)	Yes (https://docs.blender.org/manual/en/dev/physics/dynamic_paint/canvas.html?highlight=vertex%20colors)	Not native, but possible via extension; i.e., “extra”
Q3: File format supports color data, represented as face (polygon) colors	Yes, can be baked though no color management	Yes, defined by colorPerVertex (http://www.web3d.org/documents/specifications/19775-1/V3.3/Part01/components/rendering.html#X3DComposedGeometryNode)	Likely yes, but with extensions (https://github.com/KhronosGroup/glTF/tree/master/extensions#about-gltf-extensions)	No (only workaround available)	Not native, but possible via extension; i.e., “extra”
Q4: File format supports multiple maps (not only texture map, but also, for example, scanner noise map, hole filling map, etc.)	Yes (https://help.autodesk.com/cloudhelp/2018/ENU/FBX-Developer-Help/cpp_ref/class_fbx_texture.html)	Yes (http://www.web3d.org/documents/specifications/19775-1/V3.3/Part01/components/texturing.html#Multitexturing)	Yes (https://github.com/KhronosGroup/glTF/blob/master/specification/2.0/README.md#texture-data)	Yes, as long as they are images of supported formats (https://docs.blender.org/manual/en/dev/render/blender_render/textures/introduction.html)	Not native, but possible via extension; i.e., “extra”
Q5: File format supports storing everything in one file (including textures)	Yes (http://download.autodesk.com/us/fbx/20112/FBX_SDK_HELP/index.html?url=WS1a9193826455f5ff7b1de9f8127315f1b54-5a50.htm,topicNumber=d0e1054)	Yes, textures can be stored with SFImage and MFImage (http://www.web3d.org/documents/specifications/19775-1/V3.3/Part01/fieldsDef.html#SFImageAndMFImage)	Yes, Assets defined in other formats, such as images, may be stored in external files referenced via URI, stored side-by-side in GLB container, or embedded directly into the JSON using data URIs (https://github.com/KhronosGroup/glTF/blob/master/specification/2.0/README.md#gltf-basics)	Yes (https://docs.blender.org/manual/de/dev/editors/info/file.html?highlight=Pack)	Not native, but possible via extension; i.e., “extra”

						FBX	X3D (ISO/IEC 19775/19776/19777)	glTF	BLEND (Blender native format)	DSON (Daz Studio native format)
Q6: Small file size (file format native support of efficient data compression, instead of simple storing of body model file(s) in ZIP-archive)						Yes, has binary form, which is widely adopted (FBX files (.fbx) are normally saved in a binary (or native) format, but they can also be saved in ASCII format, https://help.autodesk.com/view/FBX/2018/ENU/?guid=__files_GUID_274163DA_9E89_4DCC_8AF6_10B0C498582E_htm)	Yes, Fast Infoset and EXI (http://www.web3d.org/working-groups/x3d/compressed-binary-encoding-activity)	Yes (https://github.com/KhronosGroup/glTF/blob/master/specification/2.0/README.md#glb-file-format-specification)	Yes (https://docs.blender.org/manual/en/dev/data_system/files/save.html?highlight=Compress#options)	Binary version is a zlib compressed (aka "zipped") file (http://docs.daz3d.com/doku.php/public/dson_spec/format_description/syntax/start)
Q7: File format native support for global model metadata, both predefined and custom, like model units, coordinate system, certainty values for model overall accuracy, etc. (instead of storing metadata in sidecar file, i.e., in .XMP)						Yes (designers use FBX technology to save models, metadata, and other assets in a file format, https://help.autodesk.com/view/FBX/2018/ENU/?guid=__files_GUID_F8D00075_A916_4AE2_AD7E_ED9DA2E5CF2C_htm)	Yes (http://www.web3d.org/documents/specifications/19775-1/V3.3/Part01/components/core.html#Metadata)	Yes (https://github.com/KhronosGroup/glTF/blob/master/specification/2.0/README.md#asset-1)	Yes (https://blender.stackexchange.com/questions/15729/metadata-notes-comments-in-blend-files)	Not native, but possible via extension; i.e., "extra"
Q8: File format native support for local body model metadata (like body landmarks, certainty values for landmarks placement and measurements values, and other metadata defined by 3DBP Metadata Sub-team)						Yes, with FbxBlob you can embed binary blobs of proprietary data that can be encrypted if desired (https://help.autodesk.com/view/FBX/2018/ENU/?guid=__cpp_ref_class_fbx_blob_html)	Yes (http://www.web3d.org/documents/specifications/19775-1/V3.3/Part01/components/core.html#Metadata)	Yes (https://github.com/KhronosGroup/glTF/blob/master/specification/2.0/README.md#specifying-extensions)	Yes (https://docs.blender.org/manual/en/dev/data_system/custom_properties.html)	Not native, but possible via extension; i.e., "extra"

	FBX	X3D (ISO/IEC 19775/19776/19777)	glTF	BLEND (Blender native format)	DSON (Daz Studio native format)
Q9: Extensibility (easy and native way to add extra data: custom vendor data, or some new data accepted in the future versions of the standard)	Yes, with custom metadata in FbxBlob (see Q7 and Q8)	Yes, with custom metadata (http://www.web3d.org/documents/specifications/19775-1/V3.3/Part01/components/core.html#Metadata)	Yes, glTF defines a mechanism that allows the addition of both general-purpose and vendor-specific extensions (https://github.com/KhronosGroup/glTF/blob/master/specification/2.0/README.md#specifying-extensions)	Yes, with custom metadata and custom properties (see 2 answers above), as well as custom objects, custom modifiers and custom scripts	Yes; i.e., "extra"
Q10: Support for storing measurements as a simple table (Measurement name, Value)	Yes, table can be stored as FbxBlob (see Q7) formatted in CSV, XML, JSON or another format	Yes, table can be stored as metadata string formatted in CSV, XML, JSON or another format (http://www.web3d.org/documents/specifications/19775-1/V3.3/Part01/components/core.html#Metadata)	Yes, table can be stored as metadata string formatted in CSV, XML, JSON or another format (https://github.com/KhronosGroup/glTF/blob/master/specification/2.0/README.md#asset-1)	Yes, table can be stored as metadata string (see Q7) formatted in CSV, XML, JSON or another format	Not native, but possible via extension; i.e., "extra"
Q11: Support for storing measurements not just as a simple table, but with curves on the body model surface, along which these measurement were made	Yes, for example with NURBS (https://help.autodesk.com/view/FBX/2018/ENU/?guid=__cpp_ref_class_fbx_nurbs_html)	Yes, for example with NURBS curve (http://www.web3d.org/documents/specifications/19775-1/V3.3/Part01/components/nurbs.html#NurbsCurve)	Yes, for example with LINE_STRIP polyline (https://github.com/KhronosGroup/glTF/blob/master/specification/2.0/README.md#primitivemode)	Yes, if curves are precise, as a curve length can be measured (and thus provide a value). See also a way to do this with https://wiki.blender.org/index.php/Extensions:2.6/Py/Scripts/Neuro_tool/Measurement	Not native, but possible via extension; i.e., "extra"

						FBX	X3D (ISO/IEC 19775/19776/19777)	glTF	BLEND (Blender native format)	DSON (Daz Studio native format)
Q12: File format native support for corresponding DICOM data from medical imaging devices (instead of storing metadata in sidecar file, i.e., in .DCM)						Partly. Can use FbxBlob to embed DICOM data	Yes, X3D includes strongly typed Metadata nodes in a formal component. Current work is mapping metadata correspondences formally for DICOM and other structured metadata representations. X3D is aligning to DICOM requirements through working-group liaison efforts (http://www.web3d.org/working-groups/medical). X3D also supports volume rendering http://www.web3d.org/documents/specifications/19775-1/V3.3/Part01/components/volume.html	Likely yes, as DICOM is a standard for storing and transmitting the images (https://en.wikipedia.org/wiki/DICOM)	Partly, with third-party scripts (https://docs.blender.org/manual/en/dev/search.html?q=DICOM&check_keywords=yes&area=default, https://blenderartists.org/forum/showthread.php?254057-Viewing-DICOM-Medical-Files-in-Blender)	Not native, but possible via extension; i.e., "extra"
Q13: Skeletal animation support for body animation and pose changing (to A-, T-, Y-pose or any other pose)						Yes (https://help.autodesk.com/view/FBX/2018/ENU/?guid=__cpp_ref_class_fbx_skeleton_html)	Yes, existing specification supports full skeletal definition/animation and v2.0 draft supports precise definition of any pose http://www.web3d.org/working-groups/humanoid-animation-h-anim	Yes (https://github.com/KhronosGroup/glTF/blob/master/specification/2.0/README.md#skins)	Yes (https://docs.blender.org/manual/en/dev/rigging/armatures/index.html)	Yes; single animation track
Q14: Morph targets (a.k.a. blend shapes or shape keys) support: for face animation, body shape parameters (like "Left Wrist Circumference" or "Waist Height") or other body model changes						Yes (https://forums.autodesk.com/t5/3ds-max-forum/fbx-export-does-not-export-blendshapes-morph-targets/td-p/7064645)	Partly. Coordinate and index morphing is available. HAnimDisplacer includes relevant functionality that might be adapted. Facial animation is an ongoing area of work. Morph example: http://x3dgraphics.com/examples/X3dForWebAuthors/Chapter07EventAnimationInterpolation/DolphinMorpherIndex.html	Yes (https://github.com/KhronosGroup/glTF/blob/master/specification/2.0/README.md#morph-targets)	Yes (https://docs.blender.org/manual/en/dev/animation/shape_keys/introduction.html)	Yes

	FBX	X3D (ISO/IEC 19775/19776/19777)	glTF	BLEND (Blender native format)	DSON (Daz Studio native format)
Q15: File format is optimized for rendering (no need to convert data structures stored in the file before rendering)	No, a generic file format	Yes, see X3DOM (https://doc.x3dom.org/gettingStarted/background/index.html)	Yes, fast loading is one of format design goals, glTF data structures have been designed to mirror the GPU API data as closely as possible, both in the JSON and binary files, to reduce load times (https://github.com/KhronosGroup/glTF/blob/master/specification/2.0/README.md#design-goals)	Probably yes (https://www.blend4web.com/implementation)	No
Q16: Native whole file encryption (instead of packing it to encrypted ZIP-archive or similar solutions)	No, data is readily readable	Yes (http://www.web3d.org/x3d/content/examples/Basic/Security/X3dSecurityReadMe.html)	Likely this can be achieved, but with extensions (https://github.com/KhronosGroup/glTF/tree/master/extensions#about-gltf-extensions)	No (https://blender.stackexchange.com/questions/672/does-blender-have-a-file-locking-feature)	No
Q17: Native encryption of parts of the file, i.e., vendor specific metadata or attributes (instead of storing them in external encrypted file)	Yes, with FbxBlob you can embed binary blobs of proprietary data that can be encrypted if desired (https://help.autodesk.com/view/FBX/2018/ENU/?guid=__cpp_ref_class_fbx_blob_html)	Yes, metadata strings (see answer to Q8) can be stored encrypted	Yes, metadata strings (see answer to Q8) can be stored encrypted	Yes, custom properties (see answer to Q8) can be stored encrypted	Not native, but possible via extension; i.e., “extra”
Q18: Native document signing support (instead of storing digital signature in external sidecar file)	No, though you could add a signature as metadata	Yes (http://www.web3d.org/x3d/content/examples/Basic/Security/X3dSecurityReadMe.html)	No, but proposed: https://github.com/KhronosGroup/glTF-Validator/issues/4 Meanwhile this likely can be achieved only with extensions (https://github.com/KhronosGroup/glTF/tree/master/extensions#about-gltf-extensions)	Yes, but with a third-party tool, Blender Publisher (http://download.blender.org/documentation/oldsite/oldsite.blender3d.org/215_Blender%20news%20Blender%20Publisher.html)	Not native, but possible via extension; i.e., “extra”

	FBX	X3D (ISO/IEC 19775/19776/19777)	glTF	BLEND (Blender native format)	DSON (Daz Studio native format)
Q19: Open specification of file format	No, the FBX file format is proprietary. However, the format description is exposed in the FBX Extensions SDK which provides header files for the FBX readers and writer (https://en.wikipedia.org/wiki/FBX#Limitations)	Yes (http://www.web3d.org/documents/specifications/19775-1/V3.3/Part01/Architecture.html)	Yes (https://github.com/KhronosGroup/glTF/blob/master/specification/2.0/README.md)	Yes (https://wiki.blender.org/index.php/Dev:Source/Architecture/File_Format)	Yes (http://docs.daz3d.com/doku.php/public/dson_spec/start)
Q20: There is ready-to-use open-source implementation of import/export at least in one popular programming language	There is no official open-source implementation, only a SDK interface, SDK is available for several OSes: http://usa.autodesk.com/adsk/servlet/pc/item?siteID=123112&id=26416130.[3]	Yes (Java: http://www.xj3d.org/, WebGL and JavaScript: www.x3dom.org)	Yes (https://github.com/KhronosGroup/glTF#languages)	Yes (http://archive.blender.org/download/source-code/index.html)	No open-source

[3] However, the format description is exposed in the FBX Extensions SDK which provides header files for the FBX readers and writer. There are two FBX SDK bindings for C++ and Python supplied by Autodesk. Blender includes a Python import and export script for FBX, written without using the FBX SDK (https://wiki.blender.org/index.php/Extensions:2.6/Py/Scripts/Import-Export/Autodesk_FBX) and The OpenEnded Group's Field includes a Java-based library for loading and extracting parts from a FBX file (https://web.archive.org/web/20091029223548/http://openendedgroup.com:80/field/wiki/ LoadingFBXFiles). There are third-party open-source FBX parsers, besides aforementioned: https://libraries.io/github/ideasman42/pyfbx_i42. The FBX can be represented on-disk as either binary or ASCII data; its SDK supports reading and writing both. While neither of the formats is documented, the ASCII format (http://download.autodesk.com/us/fbx/2010/fbx_sdk_help/index.html?url= WS1a9193826455f5ff-150b16da11960d83164-6c6f.htm,topicNumber=d0e127) is a tree structured document with clearly named identifiers. For the FBX binary file format, the Blender Foundation published an unofficial specification (https://code.blender.org/2013/08/fbx-binary-file-format-specification/), as well as a higher level unofficial spec (https://wiki.blender.org/index.php/User:Mont29/Foundation/ FBX_File_Structure, work in progress) for how actual data is laid out in FBX (independent of ASCII or binary format). More detail of actual use of data in FBX: https://github.com/mont29/blender-io.

	FBX	X3D (ISO/IEC 19775/19776/19777)	glTF	BLEND (Blender native format)	DSON (Daz Studio native format)
Q21: There are ready-to-use open-source implementations of import/export in a number of popular programming languages (available for integration in applications in C++, C#, Visual Basic, Java, Python, ...)	No, only SDK, see Q20	Yes. Blender support is actively reviewed and of great value. Additional tools included at http://www.web3d.org/getting-started-x3d and http://www.web3d.org/x3d-export-and-import	Yes (https://github.com/KhronosGroup/glTF#languages)	No (https://www.blender.org/forum/viewtopic.php?t=27051)	No open-source
Q22: There is good step-by-step documentation for import/export integration with common use case examples (code for mesh import/export, code for animation import/export, code for metadata import/export and so on)	Yes (http://download.autodesk.com/us/fbx/20102/FBX_SDK_Help/index.html?url=WS1a9193826455f5ff141599191216e0afc23-7407.htm,topicNumber=d0e2511)	Partly, many workflows exist, some documentation of streamlined workflows available, topic of interest and continuing work by Design Printing and Scanning Working Group. http://www.web3d.org/develop-and-deploy-x3d	Yes (https://github.com/KhronosGroup/glTF#languages)	Likely no, see answer above, but the blender Python API is public, and the documentation for the targeted file format should be accessible (https://docs.blender.org/api/2.79/)	Yes; C++ SDK sample
Q23: File format is not dead, but is in active development (with ongoing adoption of new features to the file format specification, like advanced character animation, support for 3D models from CT, MRI, X-Ray scans, etc.)	Yes (https://help.autodesk.com/view/FBX/2018/ENU/?guid=__files_GUID_9B5DFB40_E9B1_477E_B1DA_9259DAC40866_htm)	Yes (http://www.web3d.org/news and https://twitter.com/Web3Dconsortium)	Yes (https://github.com/KhronosGroup/glTF/graphs/contributors)	Yes (https://code.blender.org/)	Yes; actively developed

	FBX	X3D (ISO/IEC 19775/19776/19777)	glTF	BLEND (Blender native format)	DSON (Daz Studio native format)
Q24: The ability to inspect the file with a free simple viewer (compiled for major desktop OSes: Windows, Linux and MacOS)	Partly. Official free FBX Review is available for Windows, Mac and iOS, but no Linux (https://www.autodesk.com/products/fbx/fbx-review). Unofficial FBX reader is available in Blender, which is compiled for all major OSes (https://code.blender.org/2013/08/fbx-binary-file-format-specification/). It is also possible to dump FBX format to plaintext.	Yes (Xj3D with open-source Java viewer: https://sourceforge.net/projects/xj3d/, X3DOM: https://www.x3dom.org/nodes/, Cobweb X3D Browser: http://www.create3000.de/cobweb/)	Yes, with web-based Sketchfab, or Microsoft Paint 3D which is included with Windows 10 Version 1703 (https://github.com/KhronosGroup/glTF#editors-and-modeling-tools)	Yes, in Blender itself (https://www.blender.org/download/). Also, Sketchfab.com can import BLEND files	Partly, with Daz Studio; Windows/MacOS; no Linux
Q25: The ability not only to view but also to edit and work with the file in popular software (either with a native support or with the help of the plugins)	Yes, 3ds Max, Maya, MotionBuilder, Softimage, Blender, Rhino, Cinema4D, Unity, Unreal Engine and others (with some interoperability limitations, see Autodesk charts here: https://knowledge.autodesk.com/search-result/caas/CloudHelp/cloudhelp/ENU/123112/files/fbx-2015-compatibility-chart-html.html)	Yes, but adoption is not widespread: Blender, MeshLab, MATLAB, MAPLE, Samsung GearVR among notable software with built-in support, and 3ds Max, Maya, Rhino with support via third-party plugins (http://www.web3d.org/x3d/content/examples/X3dResources.html)	Yes, but adoption is not yet widespread: native support in Paint 3D and Sketchfab, support via plugins in Blender, Unity, Oculus, Wordpress	Yes, but the only software is Blender	Partly. Daz Studio, Poser (with plug-in), DSON Editor (third-party software), JSON text editor

4. Representation options for human body 3D models

To be able to set compatibility limitations in the future IEEE P3141, Draft Standard for 3D Body Processing, a questionnaire was developed. It contained 38 questions on what constraints should be imposed on human body 3D models.

Every question had an optional Other/Comment field, to collect the ideas that were not specified in the questionnaire initially. First, the questionnaire was sent to a narrow group of experts. The questionnaire was updated based on their feedback. Then the updated questionnaire was sent to a wide group of specialists. The answers collected in Other/Comment field were discussed and summarized in a unified manner.

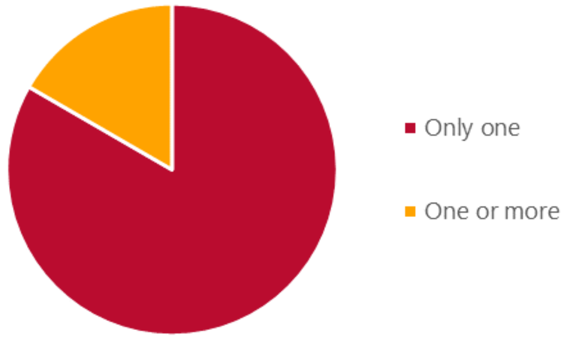
Thirty-four people representing the following organizations have answered:

- AIST
- Avametric / UC Berkeley
- Bauerfeind AG
- Body2Garment Solutions
- Browzwear
- Elasizer
- ELSE Corp
- Gneiss Concept
- Human Solutions
- Instituto de Biomecánica de Valencia
- Intel
- Lectra
- Metail Ltd.
- Novaptus Systems, Inc.
- NSRDEC, Anthropometry Team
- Picanova (3D.me)
- Polytechnic University of Tirana, Faculty of Mechanical Engineering, Textile & Fashion Department
- Size Stream
- Target
- Texel
- True Fit
- TU Delft
- University of Michigan
- US Air Force
- Web3D Consortium
- ZelusFX

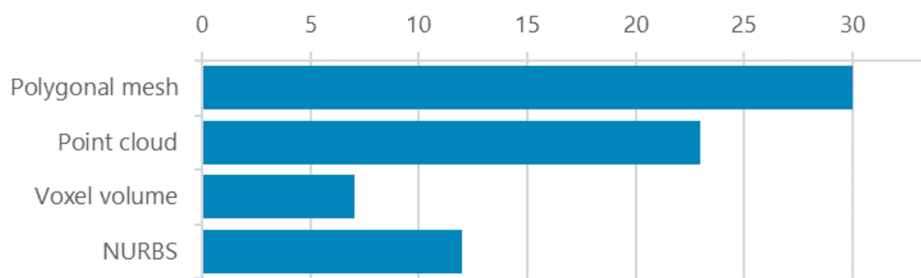
Results of the survey on representation options for human body 3D models

- Q1: Number of files per one 3D model

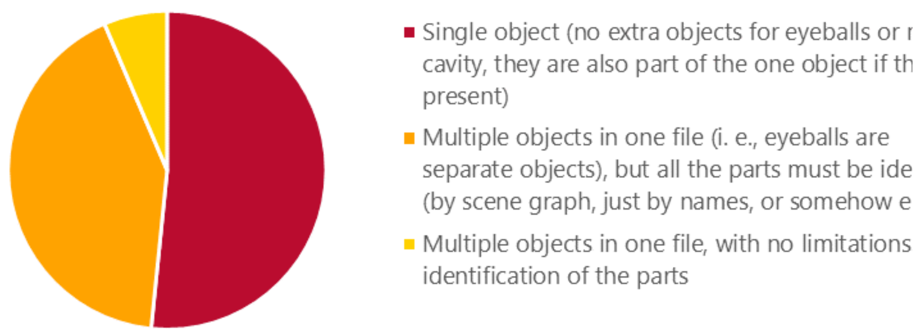
- Q2: Allowed number of human models in one file



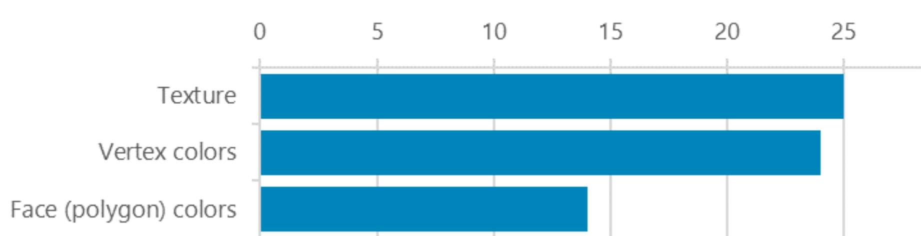
- Q3: Allowed geometry representations



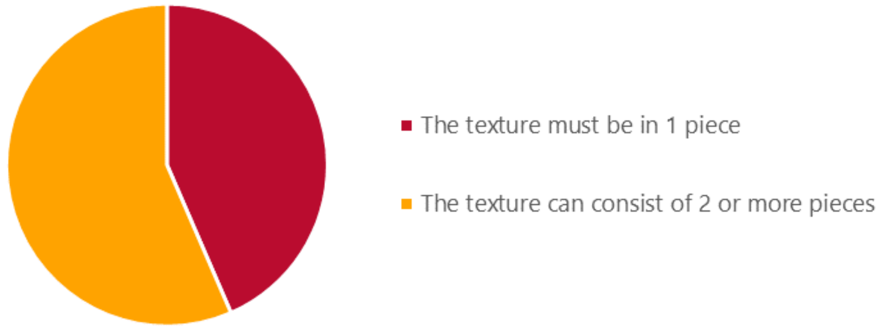
- Q4: Allowed number of objects for one human model



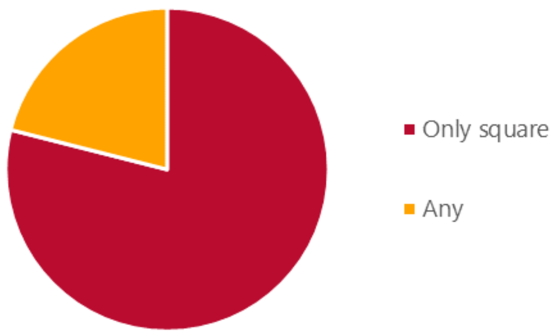
- Q5: Allowed color representations



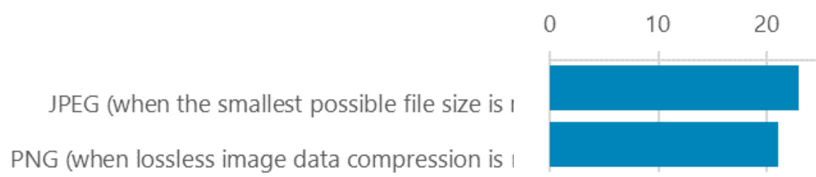
- Q6: Allowed number of textures per one object



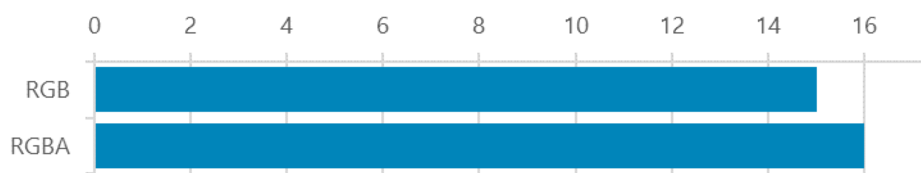
- Q7: Allowed texture aspect ratio



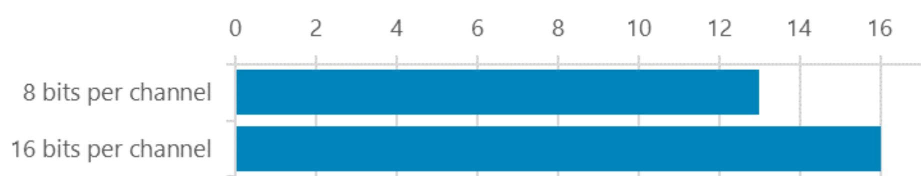
- Q8: Allowed texture image formats

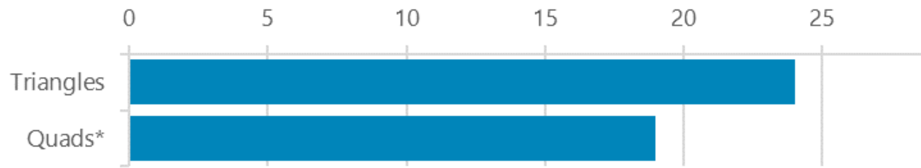


- Q9: Allowed texture channels

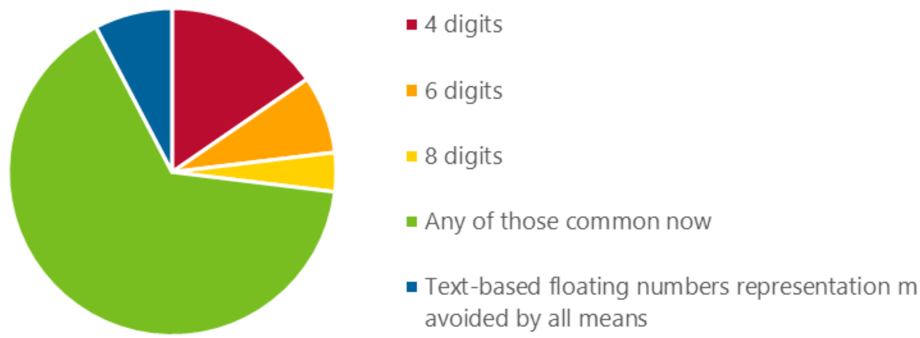


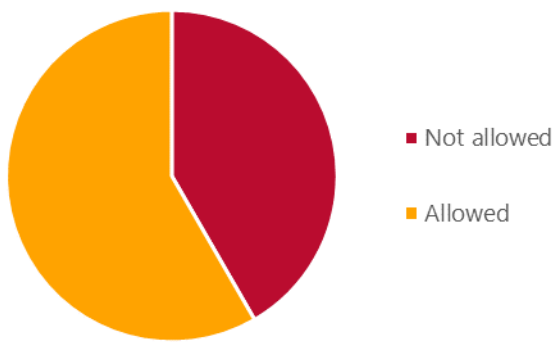
- Q10: Allowed texture color depth

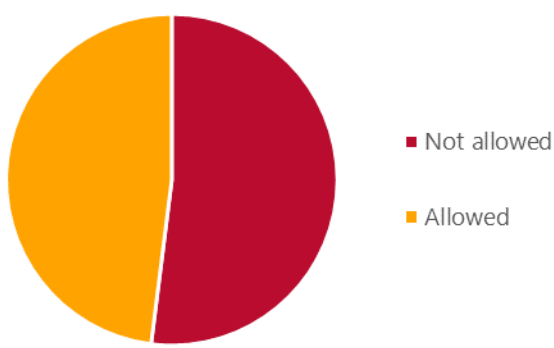


- Q11: Allowed polygons in meshes
  - 

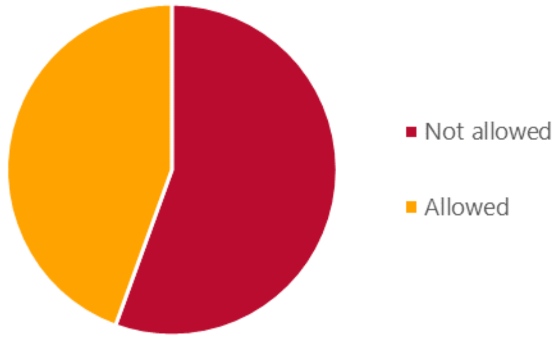
*Quads may allow meshes with anatomically correct edge loops

- Q12: Required floating numbers precision (must be specified for text-based file formats)
  - 

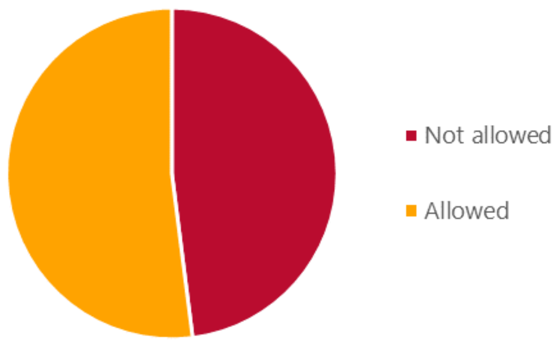
- Q13: Non-manifold vertices, non-manifold edges
  - 

- Q14: Surface self-intersections
  - 

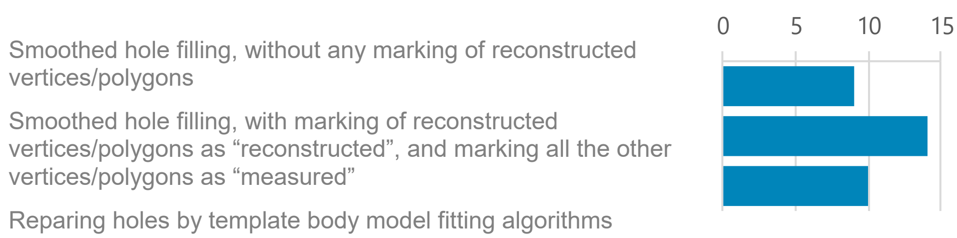
- Q15: Small objects in the air (like detached parts of earrings, small hair curls or just noise)



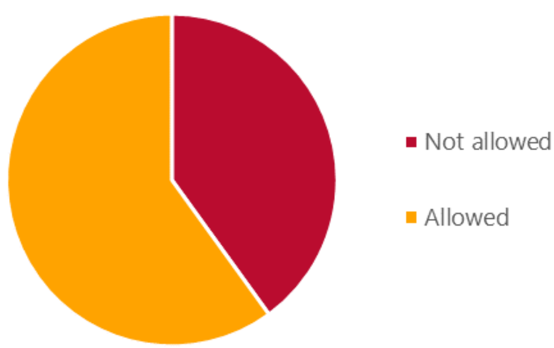
- Q16: Surface holes



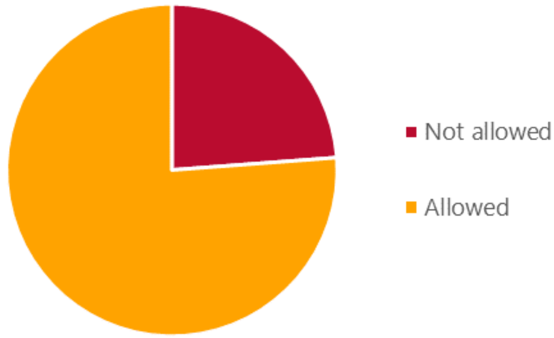
- Q17: If holes are not allowed, what hole filling methods are allowed



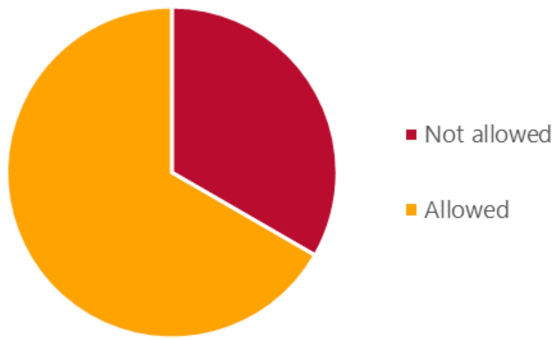
- Q18: Small tunnels



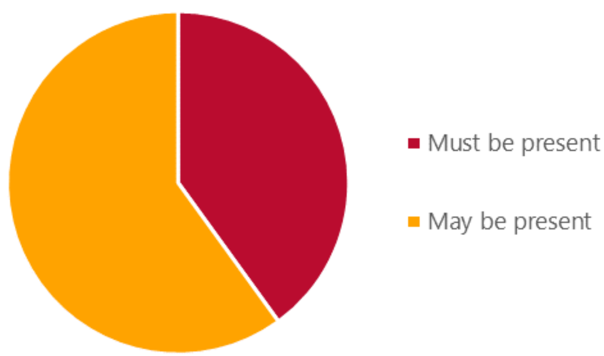
- Q19: Highly creased edges



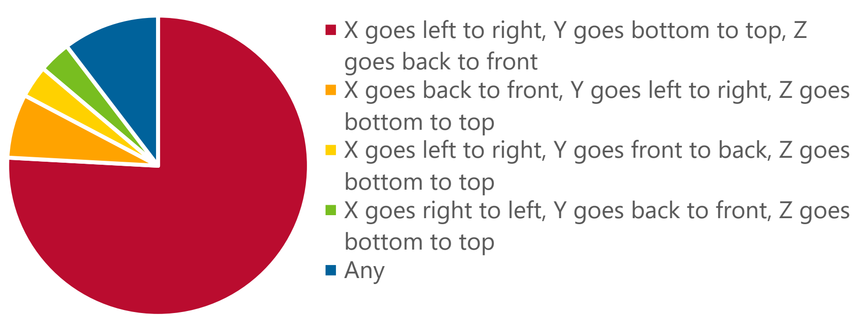
- Q20: Surface spikes



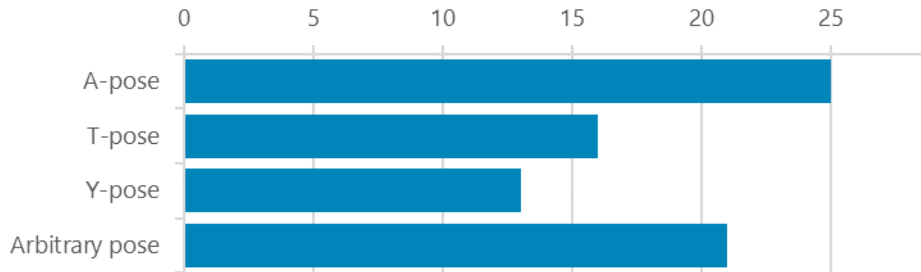
- Q21: Normals



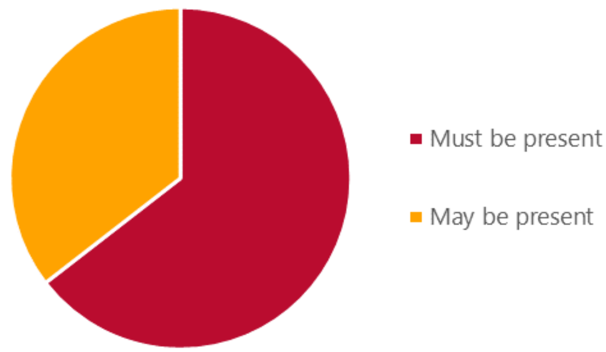
- Q22: Allowed coordinate system



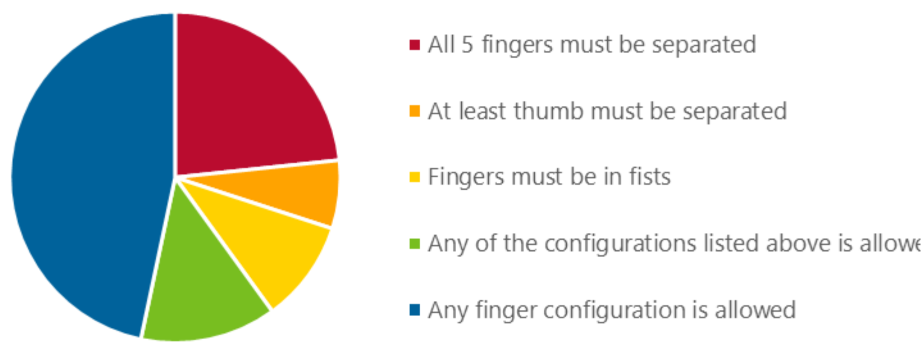
- Q23: Allowed poses



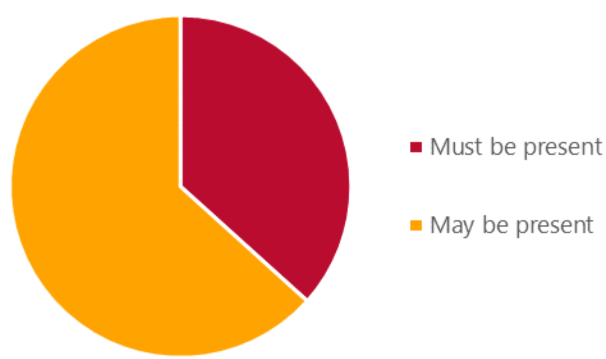
- Q24: Metadata, describing what pose is in the file



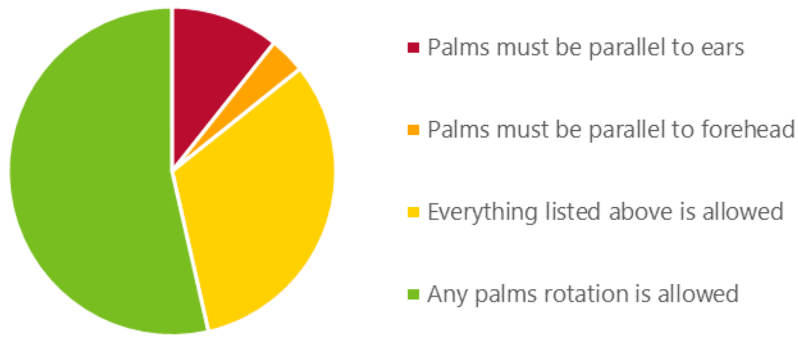
- Q25: Allowed fingers configurat ion



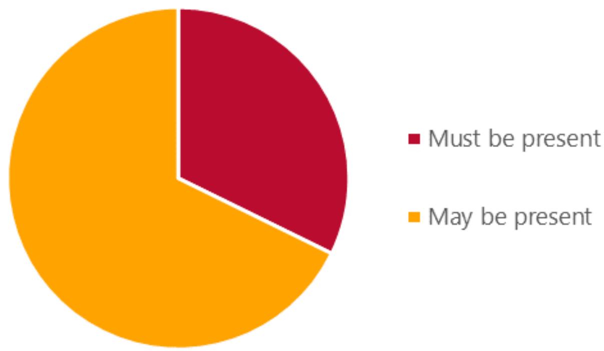
- Q26: Metadata, describing fingers configurat ion in the file



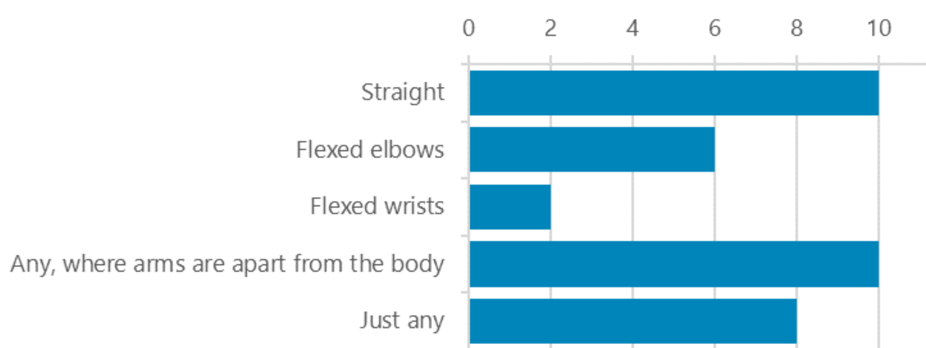
- Q27: Allowed palms rotation



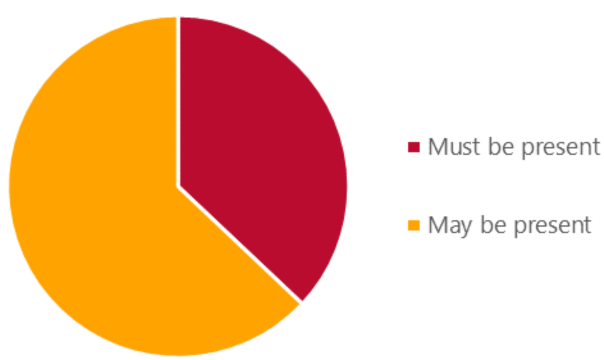
- Q28: Metadata, describing palms rotation in the file



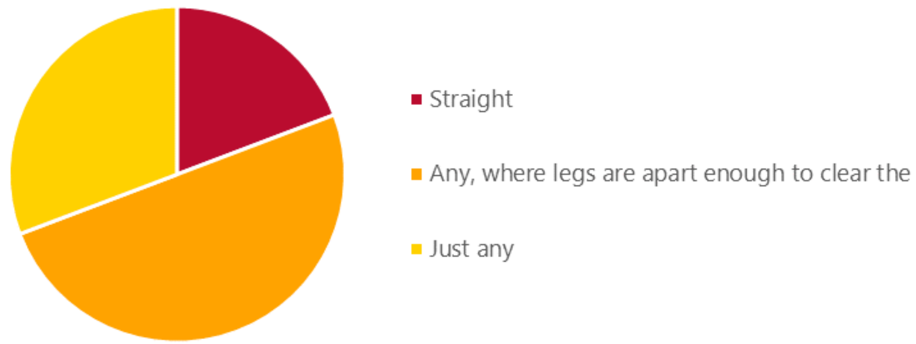
- Q29: Allowed arm bending



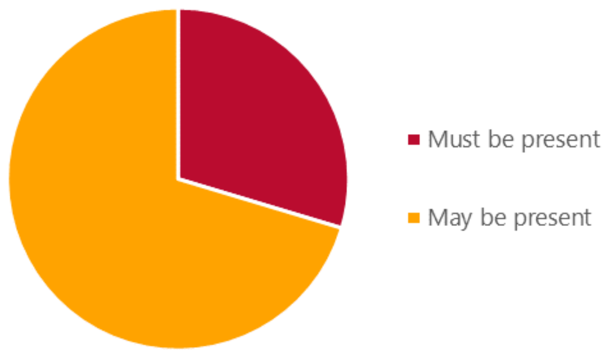
- Q30: Metadata, describing arm bending in the file



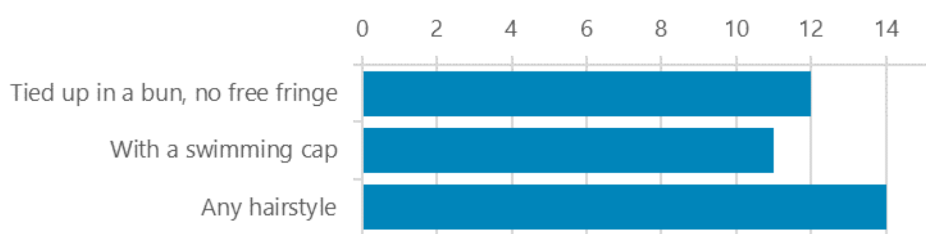
- Q31: Allowed leg bending



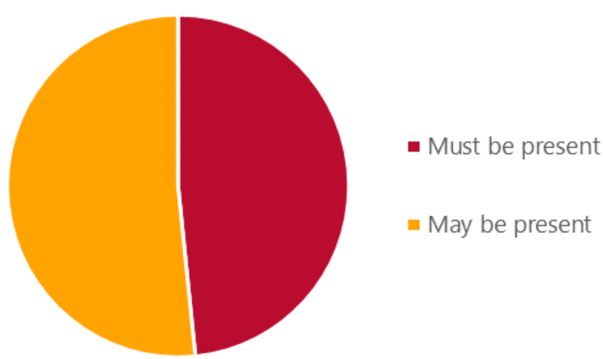
- Q32: Metadata, describing leg bending in the file



- Q33: Allowed hairstyles

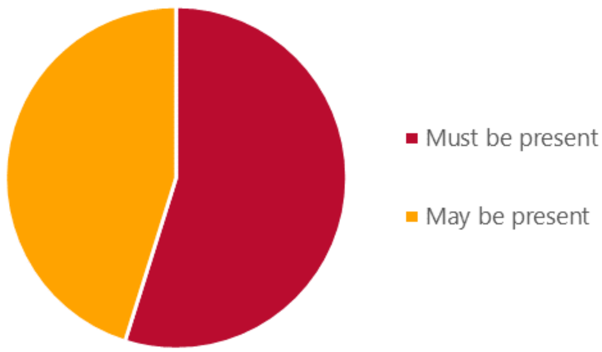


- Q34: Metadata, describing hairstyle in the file

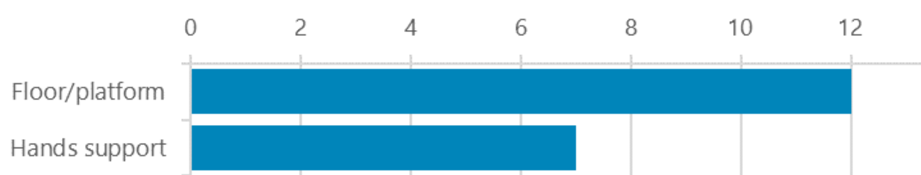


- Q35: Allowed clothes

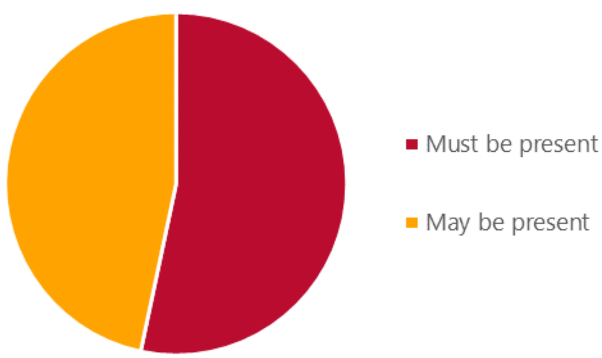
- Q36: Metadata, describing clothes in the file



- Q37: Allowed extra objects



- Q38: Metadata, describing extra objects in the file



5. Discussion

Dealing with feature-rich file formats, application of the standard limitations on representation options would be beneficial to avoid incompatibility. Here it should be noted that there may be recommendations of different representation options for different use cases. Let's consider an example of treating surface holes. Raw 3D models obtained with a 3D scanner usually have surface holes. So, when we scan dressed people for 3D printing, 3D printer slicing software usually does not accept surface holes. The common approach is to make a smooth reconstruction of surface hole areas.

But when we scan people in tight clothes and we intend to obtain body measurements from these scans, for many modern body measurements estimation algorithms it is much better to have real data with surface holes than smoothed reconstructions in slit areas (armpits, crotch).

There is also an option to introduce metadata that specify for each vertex/polygon whether it is measured, reconstructed or perhaps user-defined. In this case the future IEEE P3141, Draft Standard for 3D Body Processing may require obligatory hole filling and obligatory marking all smoothed reconstructed surfaces as "reconstructed", and all original data as "measured". So, body measurement estimation algorithms will be able to take only "measured" data into account, but animation and 3D printing algorithms will simply use all the vertices and polygons, both "measured" and "reconstructed".

Another option to consider is to provide different recommended subsets in the standard. An operator participating in a 3D body model processing pipeline may declare that it complies with a chosen subset depending on its capabilities and resources. Subsets may be as follows: basic (the easiest to implement, with the most limitations on representation options), full (the hardest to implement, supporting a wide range of options), and industry specific for additive manufacturing, apparel, medical and other industries.

6. Conclusion

The file format requirements for human body 3D models are provided. Two suitable file formats are recommended. As of 2019, these are X3D and glTF. The representation options for feature-rich file formats are considered as well.

The next steps are to provide metadata specifications and to set compatibility limitations on feature-rich file formats for the future IEEE P3141, Draft Standard for 3D Body Processing, as well as to cover file formats for clothing, footwear, accessories and equipment, with emphasis on materials and especially textiles. This will be elaborated upon in future papers.

The authors of this white paper encourage experts in the covered topics to provide feedback on relevant issues, results and prospects, as well as to join IEEE 3D Body Processing initiative. The group, including the File Format subgroup can be contacted by the following web form: https://standards.ieee.org/industry-connections/3d/bodyprocessing.html.

RAISING THE WORLD'S STANDARDS

IEEE

3 Park Avenue I New York, NY 10016-5997 USA http://standards.ieee.org
Tel.+1732-981-0060 Fax+1732-562-1571