www.cad-journal.net
Computer-Aided Design AND APPLICATIONS

A Lightweight Representation of Products: Annotated CAD Model Images in PPT Slides

Qiujiao Wang[1, 2] and Zhijie Xie[3]

[1]Southwest Jiaotong University Hope College, China, wantchoosejoy@163.com
[2]Chengdu Transportation + Tourism Big Data Application Technology Research Base, China
[3]Chengdu Office, Shanghai Tecwin Software Technology Co., Ltd, China, wanzhuxie@163.com

Corresponding author: Zhijie Xie, wanzhuxie@163.com

Abstract. In real-world industrial applications, effective communication of complex product designs at all reporting phases is a significant challenge. Traditional reporting methods that rely on CAD (Computer-Aided Design) models suffer from limitations such as large file sizes, software compatibility issues, and the need for user expertise. To address these challenges, this study proposes a novel approach using PPT (Microsoft PowerPoint) to present 3D model information, providing a lightweight alternative to traditional CAD-centric reporting. Using CATIA as an example, the study first presents techniques for automatically adjusting camera parameters and calculating image coordinates of 3D model elements. screenshots of the CAD model are then embedded into PPT slides and annotated with relevant details. The result is that a PPT file with model images and annotations can express a product in detail. This approach streamlines information transfer and improves product accessibility throughout the product design cycle, and provides a tangible reference for industrial applications.

Keywords: Image-based; Lightweight, PPT; CATIA
DOI: https://doi.org/10.14733/cadaps.2025.1136-1148

1 INTRODUCTION

In real-world industrial applications, a product design cycle typically involves multiple reporting phases. Typically, CAD models are not utilized as reporting mediums due to the following reasons.

(1) The final 3D model of a product can be significantly large, and sometimes exceed hundreds of GBs. Viewing and navigating such models requiring high-performance machines. Furthermore, the CAD model contains numerous process features and parameters [1], resulting in low efficiency when viewing the model.

(2) Engineers often work with and operate between different CAD systems [18]. Additional installation of CAD software is needed when viewing 3D models. The CAD platform may require a dedicated server to support its operation. Moreover, CAD software operation may rely on

expensive licenses [25]. As a result, companies typically install CAD software on a limited number of computers.

(3) Operating the CAD software requires professional skills. Therefore, reporting recipients may need technical personnel assistance to access and interpret the model. Moreover, directly annotating on the 3D model is inconvenient because it may have been locked at this point.

Indeed, lightweight display of CAD models has been a longstanding research topic in product design, and it can be categorized into three different forms. The first method is to use lightweight model formats, such as STL, STEP, and OBJ [6, 16, 17]. These formats alter the composition of the original model, remove the associated data between features, and retain only the appearance of the final product. The second is to layer and partially load the data based on the position and importance of the model nodes in the display environment [11, 20]. The third is to place rendering and computing tasks on the web side to alleviate the computational pressure on the client side [9, 12]. However, the latter two methods cannot be separated from their native software platforms and are not conducive to data sharing. The lightweight data format, as the popular way of data transmitting and sharing, has been widely used [7, 17, 26]. Major commercial CAD platforms, such as CATIA, NX, and SolidWorks, support the import and export of the lightweight format. Additionally, certain office software, such as Microsoft Office, supports displaying 3D models in STL and other lightweight formats.

Transmitting and viewing original 3D CAD models in real-time is inconvenient due to their size and complexity. PPT serves as a widely used file format and offers many built-in trigger action presentations, making it more convenient for use during presentations or reports compared to other ways. This paper proposes a way to present product model data that is more suitable for project reporting. It is more lightweight than other styles because the presentation of CAD model data is no longer limited to the 3D model formats; instead, model images with annotations in PPT slides. There are three advantages in displaying model images in PPT over displaying files in other formats such as STL.

(1) The annotation information can be displayed completely. The STL format itself can only express the structure of the model and cannot contain or show the annotation information.

(2) Product details can be easily displayed. In CAD software, it is easy to locate the local area of a model, and then get the information detail through the screenshot. The exported models in STL and other lightweight formats need to be operated in other independent viewing software, and whether these operations are supported is a matter to be considered.

(3) The size of the PPT file can be controlled. The size of the lightweight format model is still related to the size of the original CAD model. If the lightweight model is placed in every slide of the PPT file, the size of the PPT file with hundreds of slides is still large and unpredictable. In contrast, the size of the fixed-size screenshot is stable, it is unrelated with the size of the original model, and thus, the size of the final generated PPT file is predictable.

Therefore, during the product design cycle, PPT can be chosen to represent 3D model information. Initially, screenshots of the 3D model graphic window can be captured, followed by generating a summary PPT file from these screenshots. Product detail information can then be clearly annotated within the PPT slides. This approach avoids issues related to the transmission, viewing, and annotation of information throughout the product design cycle.

An engineer can do this job, they can first locate local areas of the model in question and take screenshots of the adjusted model, then place the screenshots into a PPT file, and finally add annotated information of the local parts in the PPT. However, in practical product design scenarios, presenting thousands of positions manually can be time-consuming. Moreover, when presenting a substantial amount of information, operators may find it challenging to identify the viewing perspective of the 3D model. Instead, they sometimes have to consider the features of the model and perform extensive calculations to confirm it. In such cases, the accuracy and efficiency of information summarization can be greatly enhanced by employing an automated method for these steps. It is challenging to locate the position of a 3D model based on its projected image [4, 19],

which may require deep learning [10, 24]. Conversely, the reverse process is relatively easy. The 2D visualization of the information from the CAD model has been widely applied in medical imaging [3] and there have been a small number of applications in the mechanical design area [21]. Moreover, even in the metaverse, there are some exploratory technological applications [23].

CAA (Component Application Architecture), a secondary development interface provided by CATIA, offers extensive function expansion APIs (Application Programming Interface). These APIs can help users develop functions that are tailored to their business needs. Many systems are developed and applied in industry based on CAA, such as papers [5, 22].

Mature commercial CAD software generally provides APIs for image capture in the graphic window, and CAA certainly includes a set of APIs to do this work. The methods for calculating the pixel coordinates in the screenshot from a spatial point are not provided. But the relationship between the spatial length and the pixel length can be deduced from the pixel dimensions of the graphic window and the spatial dimensions displayed in the graphic window. Furthermore, the pixel coordinates of the spatial point in the 3D model can be computed accordingly. Based on this relationship between the pixel length and the spatial length, we propose the method of accurately locating the corresponding 3D model elements in a screenshot image. The proposed method of summarizing information uses the CATIA APIs. However, it is also applicable to other CAD software. First, it automatically adjusts the pose of the 3D model, takes screenshots of the graphic window. Then, it calculates the corresponding pixel coordinates in the image for a 3D point within the model. It then places the image into a PPT slide, calculates the pixel coordinates in the PPT slide. Finally, it further accurately annotates the model information in the PPT slide according to the pixel coordinates.

2 RELATED FUNDAMENTALS

2.1 Model Elements

A model element refers to individual or groups of graphic objects, such as points, lines, surfaces, solids, and their collections. Moreover, individual parts or assemblies may also be considered elements within the overall product. When it comes to the coordinates of a model element, if the element is a point, then the coordinates of the element refer to the coordinates of that point. Otherwise, the coordinates of the elements refer to the coordinates of a point that can represent these elements, such as the weighted centroid of these elements.

2.2 Window Equivalent Size

The size of the graphic window is measured in pixels. This paper defines the graphic window's equivalent size as the smaller one between its height and width. When mapping the spatial dimensions of a 3D model to the image dimensions of a screenshot, the first step is to calculate the maximum bounding sphere that can be displayed in the graphic window. The diameter of this sphere corresponds to the smaller value between the height and the width of the window. This parameter will be used frequently in the algorithm flow described in this paper; hence, it is represented as the equivalent size of the window. In CAA, the 3D model graphic window for has a specific API object class called *CAT3Dviewer*, through which the size of the graphic window can be obtained, and the equivalent size can be further calculated.

2.3 Bounding Sphere

The bounding sphere is a mathematical object that contains two numerical parameters: the center and the radius. CAA uses the *CAT3DBoundingSphere* class to represent the bounding sphere. Based on their purposes, bounding spheres can be classified into two types: window bounding sphere and model bounding sphere. The window bounding sphere refers to the largest sphere that can be fully displayed within the graphic window, where its diameter corresponds to the equivalent

size of the graphic window. On the other hand, the model bounding sphere refers to the smallest sphere that can encompass the entire model, also known as the minimum bounding sphere or minimum zone sphere. The radius of the minimum bounding sphere of the model is not absolutely minimum, and there are various calculation methods [8, 13, 14], and the minimum radius obtained by these different algorithms may vary. As shown in Figure 1, the larger sphere represents the window bounding sphere, while the smaller one represents the model bounding sphere.

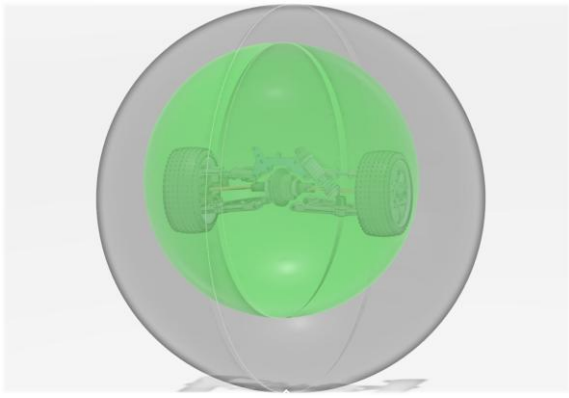

Figure 1: The model bounding sphere and the window bounding sphere.

The parameters of the model bounding sphere can generally be calculated according to the actual conditions. For regular models such as cubes, cones, etc., their bounding spheres can be calculated through mathematical conditions. For irregular models, the bounding sphere can be roughly calculated on the basis of its actual conditions. Alternatively, through CAA's API, the model bounding sphere can be obtained using the following technical approach: first, transfer the relevant model elements to the *CATI3DGeoVisu* interface; then, use its parent class method *CATIVisu::GiveRep* to obtain the model's visual representation object *CAT3DRep*, and finally, use the *CAT3DRep::GetBoundingElement* method to obtain the model's bounding sphere.

When only focusing on local models or elements, the *CAT3DViewer::ReframeOn* method can be used to center the model and maximize it within the window range. In this case, the parameters of model bounding sphere will be the same as those of the window bounding sphere.

2.4 Camera Parameters

The graphic window displays a 2D image, which represents the result of observing the 3D model from a certain viewpoint, or equivalently, the image results of photographing the 3D model with a camera placed at a certain position. The camera parameters refer to the parameter set defining the pose for observing the model. It comprises several parameters as follows:

(1) Camera position
(2) Target position
(3) Target direction
(4) Sight distance
(5) Up direction
(6) Near plane
(7) Far plane

Some parameters are mathematically equivalent. For example, using [Camera position, Target position and Up direction] and [Camera position, Target direction, Focus distance, and Up direction] can both determine the camera object. In CAA, the camera parameters can be expressed using viewpoints. Each window object has a main viewpoint and CATIA uses the main

viewpoint by default to observe the model. The main viewpoint object of CATIA's graphic window can be obtained by the function *CATViewer::GetMain3DViewpoint*.

3 IMAGE COORDINATES OF MODEL ELEMENTS

3.1 Length-Pixel Relationship

When taking a screenshot of a 3D model in the graphic window, the 2D image coordinates corresponding to the 3D points in the model will change with the camera parameters. The same spatial length in the model may have different pixel lengths in various images, leading to distinct pixel coordinates for elements across images. The coordinates of the centroid of model elements or parts in 3D space are easily obtained, and CAA provides API for calculating the 3D coordinates of model elements and their distances from other elements. Therefore, in the context of a 3D model, it is quite easy to obtain the relative coordinates between elements. In this case, for each image obtained through the graphic window, if we have the correspondence between the actual length in the 3D model and the pixel length in the image, as shown in Figure 2, the pixel coordinates of each 3D element in the image can be further calculated. Here, we term the connection between length and pixel as the length-pixel relationship, or simply the LP relationship. It represents the actual length of the model corresponding to each unit of pixels.

$$Lp = L / P \tag{1}$$

where, L represents the spatial length, and P represents the pixel length corresponding to L.

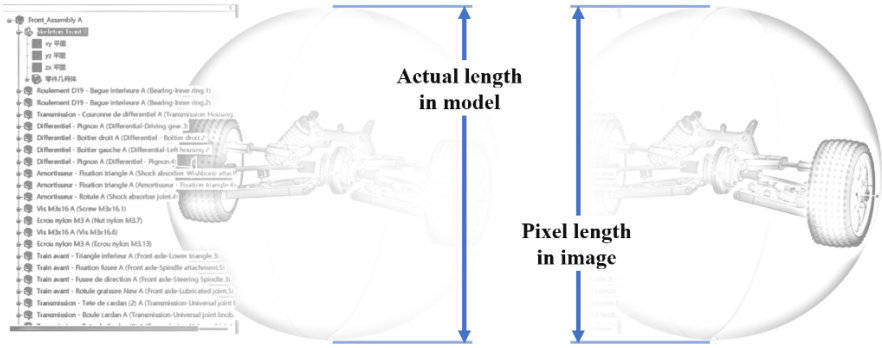

Figure 2: The actual length in the 3D model and the pixel length in the screenshot of the graphic window.

When centering the model and filling the model bounding sphere into the entire graphic window, the diameter of the window bounding sphere (actual length) corresponds to the equivalent size of the window (pixel length). Consider that if we can determine the radius of the window bounding sphere and the window's size, then the LP relationship can be readily calculated. With the help of the LP relationship, the position of any 3D element in the image can be determined, and thus the image coordinates of the element can be further computed. Therefore, the problem can be transformed into that how to obtain the radius of the window bounding sphere and the equivalent size of the graphic window.

The window bounding sphere can be obtained using the *GetGlobalBoundingSphere* function of the graphic window *CAT3DViewer*, which includes the radius value. Both the primary CATIA window and the smaller window controller generated through CAA belong to the *CATViewer* object, They can utilize the *CATViewer::GetGraphicSize* function to retrieve the graphic window's pixel size. At this point, the value of the LP relationship can be calculated as follows.

$$Lp = 2R / S \tag{2}$$

where, R is the radius of the window bounding sphere, and S is the equivalent size of the graphic window.

3.2 The Calculation of Image Coordinates

Assuming that the 3D model has been positioned correctly, either automatically or manually. We take the point $A(X,Y,Z)$ in the 3D world coordinate system $OXYZ$ as an example to illustrate how to calculate the image coordinates when the LP value is known. Point A can be regarded as a feature point, a part, or the centroid of an element. Regardless, assuming that we need to calculate its image coordinates.

Step 1: Center the model, acquire the bounding sphere and the equivalent size of the graphic window, and calculate the LP value.

Step 2: Employ the *CATNavigation3DViewer::GrabPixelImage* function to take a screenshot of the graphic window and obtain a 2D image.

Step 3: Construct a plane P parallel to the screen, with the center O of the window bounding sphere as the origin.

Step 4: Project the point A onto the plane P to derive the projected point $A'(X_L,Y_L)$. The coordinate system of the point A' is a 2D plane coordinate system $O'X'Y'$, where the origin corresponds to the center point O' of the graphic window.

Step 5: Convert the coordinates of point A' from length to pixels (X_P,Y_P):

$$\begin{cases} X_P = 1/L_P * X_L \\ Y_P = 1/L_P * Y_L \end{cases} \tag{3}$$

Step 6: Transform the coordinate system of point A' $(O'X'Y')$ into the image coordinate system $O"X"Y"$. The coordinate system where point A' is located, $O'X'Y'$, takes the center of the graphic screen as the origin, with the X-axis pointing horizontally to the right along the screen and the Y-axis pointing vertically upward along the screen. The image coordinate system takes the upper left corner of the screenshot image as the origin, with the X-axis pointing horizontally to the right along the image and the Y-axis pointing vertically downwards along the image. It is necessary to calculate the image coordinates $(x",y")$ in the coordinate system $O"X"Y"$ for the transformation of the point A', and the point $A"(x",y")$ is just the corresponding image coordinates of the 3D model point.

In fact, testing revealed that after centering the model using the *CAT3DViewer::ReframeOn* method, the window bounding sphere did not fully extend to the boundary of the graphic window. The radius R of the window bounding sphere is slightly smaller than the equivalent size S of the graphic window, and the relationship is given by:

$$R = 0.95 * S \tag{4}$$

4 AUTOMATIC ADJUSTMENT OF CAMERA PARAMETERS

Before capturing screenshots of the graphic window, it's essential to determine the displayed model content in the graphic window, such as which part to showcase and the specific area to include. In fact, the content displayed in the graphic window is determined by the camera parameters, so it is necessary to first determine the camera parameters of the window before taking a screenshot. Figure 4, utilizing 3D imaging software by Song Ho Ahn [2], provides an intuitive illustration of the correlation between the planar result of the 3D model and the camera

parameters. Figure 4(a) and Figure 4(b) present different results, with the same 3D model positions, but different camera positions.

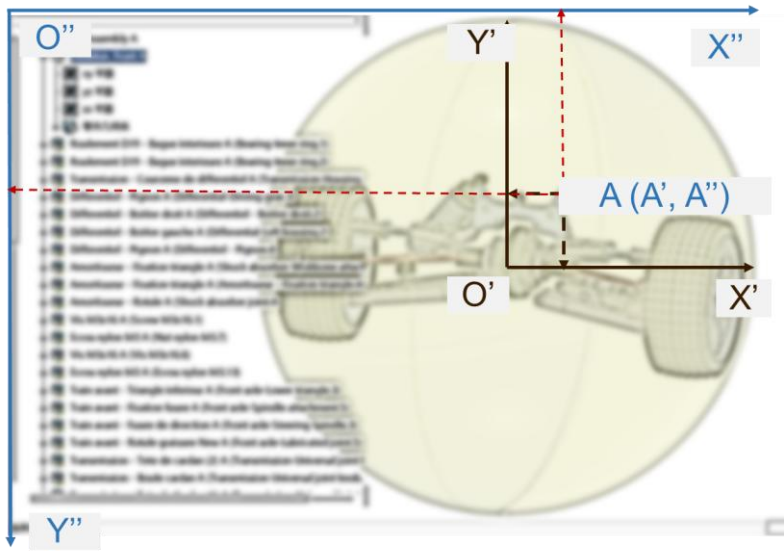

Figure 3: The coordinates of a point in different coordinate systems.

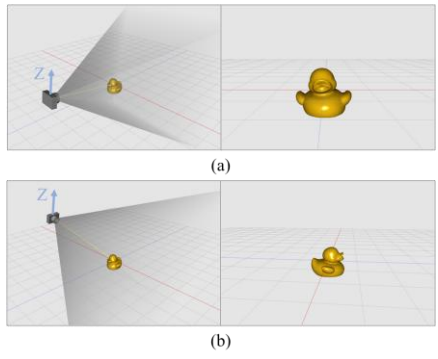

Figure 4: Results of the model observed with different camera parameters.

Setting the camera parameters typically occurs in two scenarios: firstly, when the user manually adjusts the model pose through translating, scaling, and rotating, then the window's camera parameters are determined accordingly; secondly, using API to automatically set the camera parameters based on the parts of the model that need to be displayed, so that the graphic window shows the appropriate area of the 3D model. When it is necessary to automatically set the camera parameters, it is necessary to calculate these parameters automatically and then set them to the graphic window. As shown in Figure 5, after setting the camera parameters, there is a relationship between the Focus distance F of the camera and the radius R of the window bounding sphere:

$$R = c * F \tag{5}$$

where, c is the coefficient between the two parameters. For the default main viewpoint of the graphic window in CATIA, this coefficient is 0.268, and the value is related to the camera parameters.

We called the formular FR relationship.

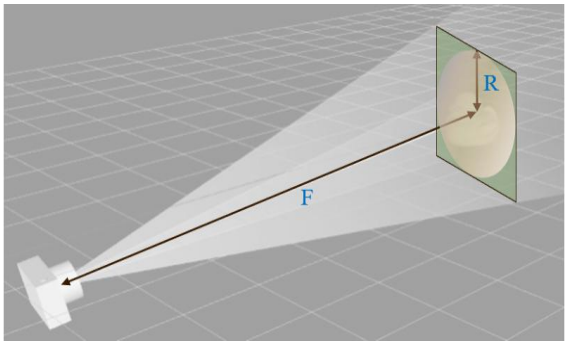

Figure 5: The Focus distance and the radius of the window bounding sphere.

Below is an explanation of automatic calculation and setting of the camera parameters, illustrated with a specific example depicted in Figure 6(a). The example involves marking several points on a particular element, with the following requirements:

(1) All points should be included and centered within the graphic window.

(2) The camera should be facing the plane formed by the points 1, 2, and 3 (assuming that these three points are not collinear).

(3) Line L_{12} formed by the points 1 and 2 should be horizontal, with the point 2 to the right of the point 1.

(4) Point 3 should be above the line L_{12}.

Based on the given conditions, it is necessary to establish four parameters for the camera: Camera position, Sight direction, Focus distance, and Up direction. The Sight direction, the Up direction, and the Right direction formed by points 1 and 2 should be mutually orthogonal. As the Figure 6 shows, the specific steps are as follows:

Step 1: Calculate the Sight direction perpendicular to the projection plane. Specifically, as shown in Figure 6(b), construct the direction V_{12} from point 1 to point 2 and the direction V_{13} from point 1 to point 3 (V_{12} and V_{13} may not be orthogonal). Take the cross product of V_{12} and V_{13}, resulting in the camera's Sight direction. At this stage, V_{12} and the Sight direction are orthogonal.

Step 2: Transform the coordinate system to make V_{12} horizontal to the right. By combining the camera's Sight direction with the orthogonal relationship between V_{12} and the Sight direction, the camera's Up direction can be obtained, as shown in Figure 6(b). At this point, the virtual camera's posture has been determined, and the subsequent steps are equivalent to adjusting the camera's Focus distance.

Step 3: Center all points within a group in the graphic window, as illustrated in Figure 6(c), and then determine the center point C and the radius R by obtaining the bounding sphere of the graphic window.

Step 4: Calculate the Focus distance F based on the Formula (5). Then, calculate the Camera position based on the center of the sphere C, the Focus distance F, and the Sight direction, as shown in Figure 6(c).

Step 5: At this stage, all four necessary parameters of the camera have been calculated. Set the parameters to the graphic window and the window will display the model that as shown in Figure 6(d).

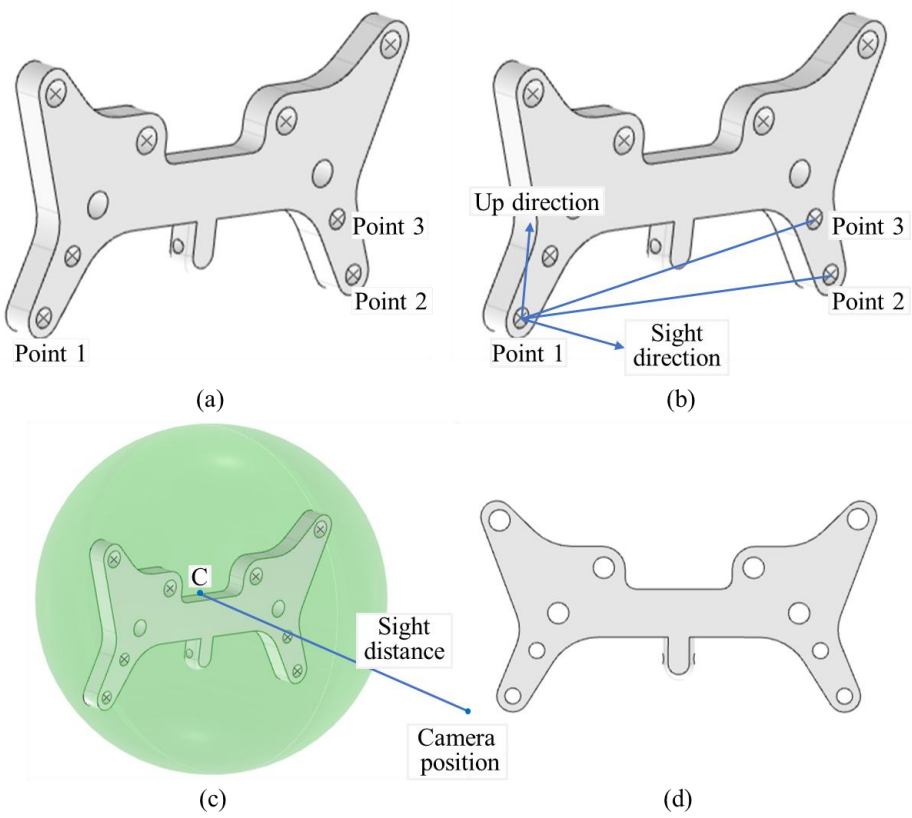

Figure 6: The process of automatic calculating and setting the camera parameters.

5 PPT ANNOTATION

The size of a PPT slide is measured in pixels and is determined by the slide settings [15]. The coordinate system of a PPT slide is same to the image coordinate system, with the origin at the upper left corner, the X-axis pointing horizontally to the right, and the Y-axis pointing vertically downward. When annotating a 3D points on a PPT slide with a model screenshot, the coordinates of the 3D points in the PPT slide should be calculated using the following formula:

$$(X,Y)_{\text{3D-in-ppt}} = (X,Y)_{\text{3D-in-image}} * Scale + (X,Y)_{\text{image-in-ppt}} \quad (6)$$

where:

$(X,Y)_{\text{3D-in-ppt}}$ represents the 2D coordinates of the 3D point in the PPT slide.

$(X,Y)_{\text{3D-in-image}}$ represents the 2D coordinates of the 3D point in the image, as calculated in the previous section.

$Scale$ represents the scaling factor when inserting the image into the PPT slide. For example, if the original size of an image is 600 * 400, and it is inserted into a PPT with a size of 300 * 200, then the scaling factor is 0.5. In fact, the scaling factors for the width and height of an image can be different, but that can cause distortion and deformation of the image; therefore, here we set the scaling factors for both the width and height of the image to the same value.

$(X,Y)_{\text{image-in-ppt}}$ represents the PPT coordinates of the image, indicating the position where the upper left corner (or other reference locations, it is related to the current settings of PPT) of the image is inserted into the PPT slide.

For instance, if the coordinates of a 3D point in the image are (200, 120), and the original image size is 600 * 600. When inserting the image into the PPT at position (30, 40) with an inserted size of 300 * 300, the scaling factor is 0.5. The coordinates of the 3D point in the PPT slide would then be calculated as (130, 100)

Figure 7 displays one of the final PPT slides, featuring 3D coordinate annotations of each key position point. In fact, the content of the annotations is related to the actual information that needs to be displayed, and other information about the model can also be annotated according to the business requirements, such as procurement information, process parameter, and so on.

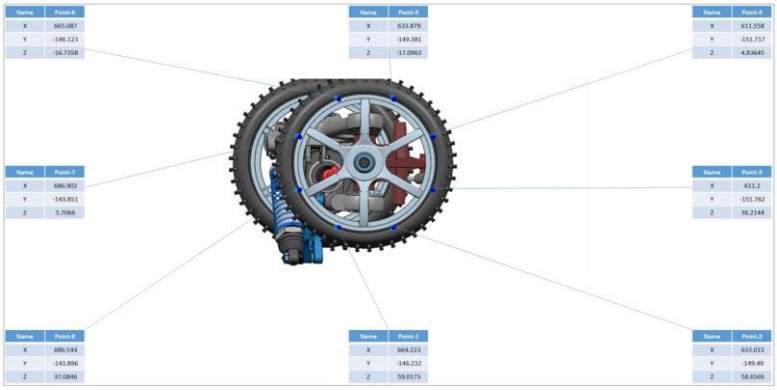

Figure 7: The results of the created PPT slide.

6 THE TOOL AND WORKFLOW

A demonstration tool has been developed to implement the technology described in this paper. An instructional video demonstrating the tool's basic operations has also been uploaded to https://github.com/wantchoosejoy/CAD-Camera. The tool includes additional functions beyond those described in the paper. For instance, it can detect whether a key point falls within the graphic window. Certain functions, not central to the paper's focus, were not detailed discussed in this paper. However, these functions can be further explored using the techniques outlined in this paper.

The preceding explanations detail the procedures for automatically setting camera parameters, computing the image coordinates of 3D points, and establishing the PPT coordinates of these points. Below, we provide a summary of the overall workflow, as illustrated in Figure 8.

7 CONCLUSIONS

With the window bounding sphere of the CAD software and the pixel size of the graphic window, the correspondence between the spatial length within the 3D model and the pixel size in the graphic window screenshot can be calculated. Using this correspondence, the pixel length in the graphic window screenshot corresponding to the spatial length can be calculated. Based on these principles, the 2D image coordinates of any 3D point can be calculated, and any 3D point in a model screenshot can be located. When this screenshot is placed in a PPT slide, since the size of a PPT slide is measured in pixels, the 2D coordinates of a 3D point in the slide can be calculated. As a

result, extended text, graphics, tables, and other annotations can also be included in the PPT slides, and precisely pointe to the 2D position in the slide that corresponds to the 3D point.

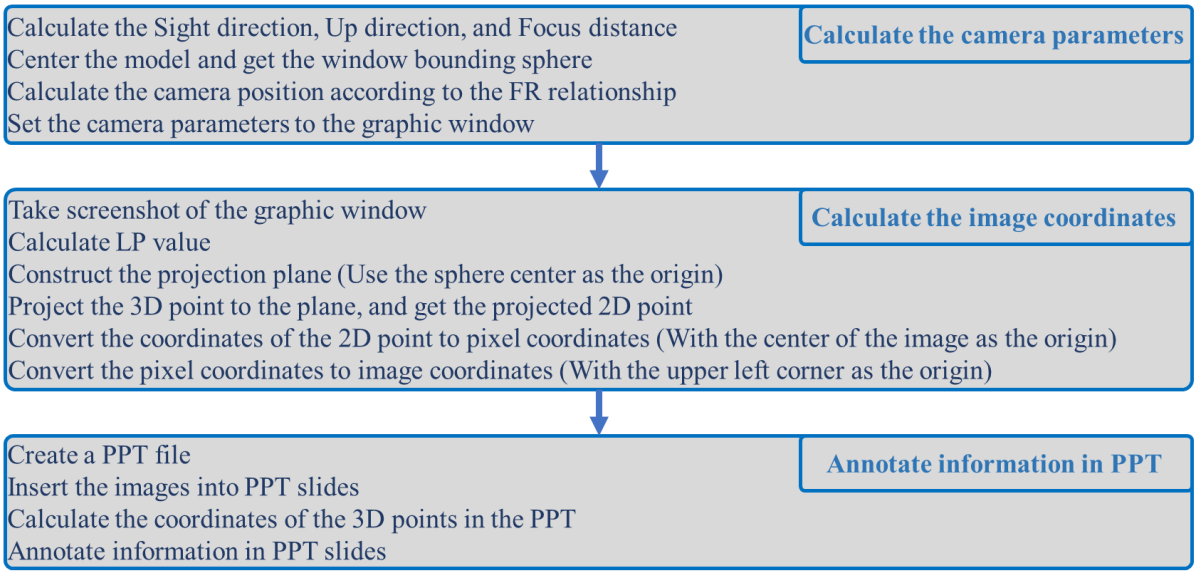

Figure 8: The overall workflow for summarizing 3D model information to a PPT file.

Therefore, a PPT file with model images and annotations can express a product in detail. This approach effectively tackles challenges such as the high memory consumption of CAD models and the limitations of free viewing on different devices when summarizing product information. In addition, this paper provides a specific reference for industrial applications by explaining the CATIA API and an examples of add-in tool.

8 ACKNOWLEDGEMENT

We thank the reviewers for their helpful suggestions. We thank Lei Tang and Baoqin Zhao of Tecwin Software for their technical advice. This work was supported by the Chengdu Philosophy and Social Science Key Research Base, China (No. 20231003).

Qiujiao Wang, https://orcid.org/0000-0003-0346-1198
Zhijie Xie, https://orcid.org/0009-0007-0031-2273

REFERENCES

[1] Agarwal D.; Christos K.; Robinson T. T., Armstrong C. G.: Using parametric effectiveness for efficient CAD-based adjoint optimization, Computer-Aided Design and Applications, 16(4), 2019, 703-19. http://doi.org/10.14733/cadaps.2019.703-719
[2] OpenGL Camera, https://www.songho.ca/opengl/gl_camera.html, Ahn S. H.
[3] Alavez N. B.; Hurtado-Esquivel C. I.; Trujillo-Romero C. J., Rios A. A.: Semiautomatic Generation of a Three-Dimensional Human Anatomical Model of Bone for Biomedical Applications: First Approach, Villahermosa Tabasco, Mexico, Springer Science and Business Media Deutschland GmbH, 2024, 258-74. http://doi.org/10.1007/978-3-031-46933-6_28

[4] Bauza M.; Bronars A.; Rodriguez A.: Tac2Pose: Tactile Object Pose Estimation from the First Touch., 2022. http://doi.org/10.48550/arXiv.2204.11701
[5] Dong Y.; Su F.; Sun G.; Liu Y., Zhang F.: A feature-based method for tire pattern reverse modeling, Adv Eng Softw, 124 2018, 73-89. http://doi.org/10.1016/j.advengsoft.2018.08.008
[6] Hallmann M.; Goetz S.; Schleich B.: Mapping of GD&T information and PMI between 3D product models in the STEP and STL format, Cad Computer Aided Design, 115 2019, 293-306. http://doi.org/10.1016/j.cad.2019.06.006
[7] Huang L.; Zhang X.: Automatic identification of features from CAD models based on STL files, Taiyuan, China, Trans Tech Publications, 2012, 2524-7. http://doi.org/10.4028/www.scientific.net/AMM.220-223.2524
[8] Huang X.; Zhang S.; Zhou L., Huang L.: A hybrid algorithm for the minimum bounding sphere problem, Oper Res Lett, 50(2), 2022, 150-4. http://doi.org/10.1016/j.orl.2022.01.013
[9] Lee H.; Jauhar T. A.; Kim I.; Kwon S., Han S.: A web-based solution for collaborative design supporting multiple CAD systems, Poznan, Poland, Association for Computing Machinery, 2018, ACM SIGGRAPH; EuroVR Association; Khronos Group. http://doi.org/10.1145/3208806.3208822
[10] Li F.; Wu Z.; Li J.; Lai Z.; Zhao B., Min C.: A multi-step cnn-based estimation of aircraft landing gear angles, Sensors-Basel, 21(24), 2021. http://doi.org/10.3390/s21248440
[11] Liu W.; Zhou X.; Zhang X., Niu Q.: Three-dimensional (3D) CAD model lightweight scheme for large-scale assembly and simulation, Int J Comput Integ M, 28(5), 2015, 520-33. http://doi.org/10.1080/0951192X.2014.880811
[12] Lupinetti K.; Cabiddu D.; Giannini F., Monti M.: CAD3A: A web-based application to visualize and semantically enhance CAD assembly models, Sorrento, Italy, Institute of Electrical and Electronics Engineers Inc., 2019, 462-9. http://doi.org/10.1109/SITIS.2019.00080
[13] Meng F.; Xu C.; Hao J., Xiao D.: Sphericity evaluation based on minimum circumscribed sphere method, Singapore, Singapore, Trans Tech Publications, 2012, 3146-51. http://doi.org/10.4028/www.scientific.net/AMR.433-440.3146
[14] Meng F.; Xu C.; Li H., Hao J.: The relationship between minimum zone sphere and minimum circumscribed sphere and maximum inscribed sphere, Hong Kong, Hong Kong, Trans Tech Publications, 2012, 658-65. http://doi.org/10.4028/www.scientific.net/AMM.157-158.658
[15] PowerPoint VBA reference, https://learn.microsoft.com/en-us/office/vba/api/overview/powerpoint, Microsoft
[16] Nguyen C. H. P.; Choi Y.: Triangular Mesh and Boundary Representation Combined Approach for 3D CAD Lightweight Representation for Collaborative Product Development, J Comput Inf Sci Eng, 19(1), 2019. http://doi.org/10.1115/1.4041777
[17] Nishida I., Shirase K.: Automated process planning system for end-milling operation by cad model in STL format, Int J Auto Tech-Jpn, 15(2), 2021, 149-57. http://doi.org/10.20965/IJAT.2021.P0149
[18] Sadler J. E.; Day R. D.; Bronson P. G.; Tovar J. L., Salmon J. L.: A neutral XML design framework for generating parametric parts in multiple CAD systems, Computer-Aided Design and Applications, 16(5), 2019, 923-35. http://doi.org/10.14733/cadaps.2019.923-935
[19] Stiller P. F.: Aligning images with CAD models via quaternion optimization, San Diego, CA, United States, SPIE, 2011, The Society of Photo-Optical Instrumentation Engineers (SPIE). http://doi.org/10.1117/12.894930
[20] Tang R.; Liu F.; Xiao X.: A Lightweight Approach for Large CAD Models Based on Lazy Loading, 99 Min An Dong Lu, Yinzhou District, Zhejiang Province, Ningbo, China, Institute of Electrical and Electronics Engineers Inc., 2023, 1977-82. http://doi.org/10.1109/ICIEA58696.2023.10241576
[21] Wong H.: Bitmap generation from computer-aided design for potential layer-quality evaluation in electron beam additive manufacturing, Rapid Prototyping J, 26(5), 2020, 941-50. http://doi.org/10.1108/RPJ-05-2019-0146

[22] Zhang W.; Cao Q.; Zhang L.; Zhao Y.; Lei P.; Wu C.; Tan H.; Wang Q., Zou X.: Research on digital rapid design of modularized assembly tooling based on CAA, Cirp J Manuf Sci Tec, 46 2023, 157-77. http://doi.org/10.1016/j.cirpj.2023.08.009

[23] Zhao J.; Wang K.: A Metaverse Scene Generation and Optimization Algorithm Based on Artificial Intelligence and CAD, Computer-Aided Design and Applications, 21(S14), 2024, 31-45. http://doi.org/10.14733/cadaps.2024.S14.31-45

[24] Zheng Y.; Mamledesai H.; Imam H., Ahmad R.: A novel deep learning-based automatic damage detection and localization method for remanufacturing/repair, Computer-Aided Design and Applications, 18(6), 2021, 1359-72. http://doi.org/10.14733/cadaps.2021.1359-1372

[25] Zhimin T.; Qi L.; Guangwen Y.: Integrating cloud-computing-specific model into aircraft design, Beijing, China, Springer Verlag, 2009, 632-7. http://doi.org/10.1007/978-3-642-10665-1_64

[26] Zhou H.; Wu J.: Research on CAD/CAM integration methods based on the STL model, Mexico City, Mexico, Springer Verlag, 2016, 1201-7. http://doi.org/10.1007/978-3-662-48768-6_134