A Novel 3D Reconstruction Approach Based on Camera Perspectives

Muhammad Abuzar Fahiem and Abad Shah

Department of Computer Science and Engineering
University of Engineering and Technology, Lahore, Pakistan
abuzar@uet.edu.pk

Abstract: Engineering industry requires line drawings for manufacturing, machining and production of engineering equipments/objects. The generation of these paper-based drawings or computerized drawings is a complex and time consuming task. Conventionally, these drawings contain three two dimensional (2D) orthographic views, namely *top*, *front* and *side* of an object. Modern trends in engineering industry require three dimensional (3D) engineering drawings. Therefore, to fulfill this requirement the conversion of these 2D drawings to 3D drawings is essential. This conversion is referred to as the *reconstruction*. Various approaches have been proposed for the conversion/reconstruction using existing drawings. In this paper, we propose a novel 3D reconstruction approach which uses camera perspectives in the reconstruction process. Note that in the existing approaches this feature (camera perspective) is not used. Another salient feature of our approach is its underlying mechanism of tangential lines and hypothetical cuboid. Using our proposed approach, manufacturing cost and time can be saved, and it can also be helpful in technology transfer. [Journal of American Science 2010;6(7):342-352]. (ISSN: 1545-1003).

Keywords: 3D Reconstruction, Boundary Representation, Constructive Solid Geometry, Tangential Lines, Hypothetical Cuboid, 3D Modeling

1. Introduction

The area of computer vision deals with the representation, storage and retrieval of the real world objects as computer images. However, these computer images are generally two dimensional (2D) but the real world objects are three dimensional (3D). Therefore, to get real representation and impression of the real world objects, it is necessary to transform 2D computer images into 3D objects (Gingold et al., 2009). These transformations can be categorized into two broad domains; 3D modeling and 3D reconstruction.

The 3D modeling is used in a wide range of applications such as restoration of ancient sculptures, designing of ergonomically suitable costumes, development of aerodynamic vehicle profiles, generation of cockpit helmet designs, modeling of historical monument, and creation of 3D animated movies (Sparacino et al., 1995). It is based on raster images of objects from angle invariant multiple camera images but the main emphasis is on appearance of the 3D models of an object. Number of camera images may range from a few to some hundreds. Moreover, before proceeding to the 3D modeling, a proper view planning is required (Scott et al., 2004), and the 3D model acquisition pipeline needs to be established (Bernardini & Rushmeier, 2002).

The 3D reconstruction primarily focuses on the applications/objects related to engineering industry such as computer aided designing (CAD) and computer aided machining (CAM). The engineering objects are basically components that are composed of different geometric shapes such as triangle, rectangle, cuboid, cylinder, cone, pyramid, sphere, toroid etc. These objects are represented in the form of line drawings in engineering industry. As mentioned earlier that conventionally these drawings contain three 2D orthographic views namely *top*, *front* and *side* of an object. These drawings are either paper based, prepared manually by skilled draftsmen/engineers, or they are computerized, prepared using different CAD tools. With the advent of the modern engineering technologies like rapid prototyping (RP), and computerized numerically controlled (CNC) machines, there is an urgent need to convert these views of 2D drawings into 3D drawings to take full advantage of above technologies. This 2D to 3D conversion is technically termed as a *3D reconstruction*, and for this purpose various approaches have evolved (Wang & Grinstein, 1993; Company et al., 2005, Fahiem, 2009).

There are some situations where drawings of engineering objects are missing or incomplete. Such situations often occur with the third world countries where objects/equipments are supplied but their technology is not transferred in terms of engineering drawings. Redesigning or duplications of these engineering objects require manual reproduction of missing drawings which is a hectic, costly and time consuming job because it requires skilled human resources. The process of reproducing engineering drawings of an object can be made convenient, cost and time effective by automating the job provided these objects are represented in some computerized vector format. For this automation, orthographic camera perspectives of the objects can be used. Then, the 2D drawings are extracted from these perspectives and 2D to 3D reconstruction of drawings is done.

In this paper, we propose a novel 3D reconstruction approach using 2D camera perspectives that makes our approach different from the existing approaches. This approach does not require already built engineering drawings. The prime focus of the proposed approach is on reconstruction and vectorization of these 2D camera perspectives into a 3D engineering drawing. The vectorization process is performed on the drawing exchange file (DXF) format which is recognized by the different CAD/CAM tools. This approach has potential to be helpful in the reverse engineering applications.

The remainder of this paper is organized as follows. A detailed survey of various 3D modeling and 3D reconstruction approaches is given in Section 2. In Section 3, we present our approach while Section 4 is dedicated to our concluding remarks.

2. Related Work

In this section, we give survey of various existing 3D modeling and 3D reconstruction approaches.

2.1. 3D Modeling Approaches

The existing 3D modeling approaches heavily depend on different types of hardware employed, and use of different light sources (Rusinkiewicz et al., 2002; Fleck et al., 2009). The hardware can be 3D cameras and scanners based on different principles. While the different light sources can be laser light, structured light or even an ambient light. In the following sections, more details of existing modeling approaches are given.

2.1.1. Time of Flight Approach

These approaches are based on 3D cameras/scanners that operate on the time of flight (ToF) principle of laser light (Blais et al., 2003). In this type of approaches, the produced laser beam reflects back to a laser detector after striking on the surface of an object. The trip time of the laser beam is recorded and the distance is calculated using Equation (1), c being the velocity of the light.

$$z = (c * t)/2 \qquad (1)$$

These scanners suffer from a drawback that the measurement of so time for fast beam is very difficult. Cost of the measurement is directly proportional to accuracy of the results. Another problem with which these scanners encounter is; to get accurate results, it is necessary that the reflection of laser beam should be at a certain incident angle to the detector from the object surface. A deflection of the beam causes either no or wrong measurement of the time. These scanners are good for the modeling of far objects such as buildings etc.

2.1.2. Laser Triangulation Approach

This type of approaches is based on 3D cameras/scanners which operate on the laser triangulation (LT) principle (Xu et al., 1998). In the LT principle, a camera captures the laser dot position when the laser light strikes on an object surface. The laser source, the laser dot on object surface and the camera lens form a triangle (see Figure 1), and the distance z of a point on object surface from the laser source is calculated by solving this triangle using Equation (2).

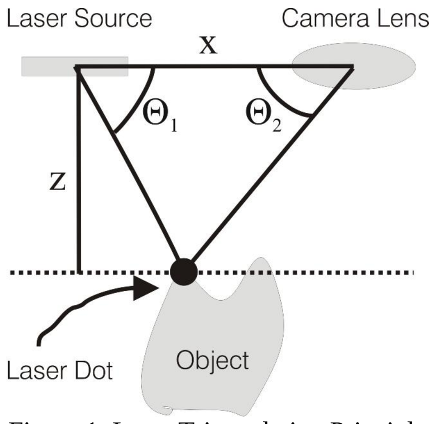

Figure 1: Laser Triangulation Principle

$$z = \frac{x}{1/\tan\theta_1 + 1/\tan\theta_2} \qquad (2)$$

These systems of the scanners operating on the LT principle are accurate but they are expensive. These scanners are limited to a measurement of the distance in the order of a few meters. Moreover, these fail in the situations where deep narrow holes are present in object surface.

2.1.3. Structured Light Approach

Structured light (SL) is usually a strip or an array of strips of while halogen light produced from a projector. When these strips of light intersect with an object present in their path, they reflect back and produce a line of illumination on the surface of the object. A camera captures this line of illumination and the distance of a point on this line of illumination from the camera is calculated through triangulation (Dipanda & Woo, 2004).

SL approach is suitable for retrieving protruded / projected shapes but is not useful in modeling holes/cavities because the line of illumination generated by SL may break or get invisible after striking with the holes/cavities.

2.1.4. Stereoscopy Approach

This type of approaches works on the human vision system and uses two cameras located at a known distance (Fofi et al., 2003). Correspondence of the pictures from both cameras is employed to retrieve the distance information. The correspondence is to find a set of pixels in one picture which can be identified as the same set of pixels in other picture, based upon the matching of certain common features overlapping in both the pictures. The distance information *z* can then be calculated by solving the triangle formed by two cameras and the corresponded pixels, using Equation (2) (see Figure 2).

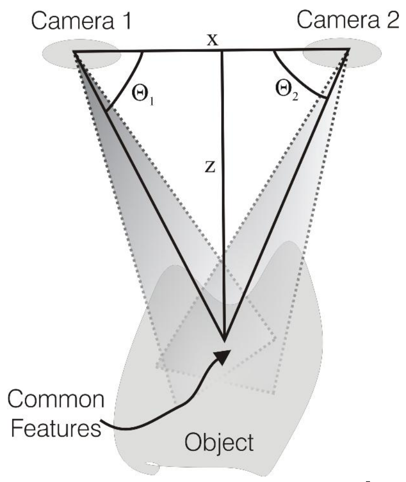

Figure 2: Stereoscopic Approach

This correspondence, itself, is a major problem in such scanners and they require heavy computational time and do not guarantee a reasonable accuracy because the correspondence is based on probabilistic matches of common features.

2. 2. 3D Reconstruction Approaches

The main theme of 3D reconstruction approaches is on vectorized modeling of a solid object from its 2D line drawings. These approaches employee algorithmic computations to generate a 3D solid object while on the other hand 3D modeling approaches are based on intense hardware support. Another major difference between these two kinds of approaches, 3D reconstruction and 3D modeling is that the former is a vector approach while the later is a raster approach. Moreover, 3D modeling is used for visual refinements while 3D reconstruction is used for industrial purposes especially in CAD/CAM industry.

The 3D reconstruction approaches can broadly be divided into two main categories; single-view approaches (discussed in Section 2.2.1) and multi-view approaches (discussed in Section 2.2.2).

2.2.1. Single-view Approach

The single-view approach checks sufficient and necessary conditions in the reconstruction of a 3D solid object from a single view in the drawing (Cooper, 2005; Martin et al., 2005; Feng et al., 2006; Kyratzi & Sapidis, 2009). This approach and its extensions are not ‘reconstruction’ approaches in the real sense because they do not actually perform any process of reconstruction. Rather they only provide labeling schemes to check the correctness of a line drawing or perceive a 3D solid object from 2D lines. These approaches do not meet the requirements of the engineering industry due to their poor accuracy. However, they are helpful in 3D visualization of free hand sketches and artistic strokes. These approaches can further be divided into following categories:

i) Labeling Approach
ii) Gradient Space (GS) Approach
iii) Linear Programming (LP) Approach
iv) Perceptual Approach
v) Primitive Identification (PI) Approach

Labeling approach marks each line in a drawing with either of three labels; convex, concave or occluding (Huffman, 1971; Clowes, 1971). These provide the necessary conditions for a drawing to be reconstructed into a 3D solid object and do not actually perform the 3D reconstruction. Gradient space approach develops a relationship between gradient of a face and slope of a line present in a drawing (Mackworth, 1973). This approach marks the lines with convex or concave labels depending upon its slope with respect to gradient. This approach, like labeling approach, provides only the necessary conditions for a drawing to be reconstructed into a 3D solid object. Linear programming approach provides necessary as well as sufficient conditions for 3D reconstruction of a solid object from its drawing and provides a system of linear equations to perform reconstruction (Sugihara, 1986). Perceptual approach uses adjacency graph along with the labeling scheme to perceive a 3D solid object from the lines in a drawing (Lamb & Bandopadhay, 1990). This approach is capable of correcting the slight roughness in line drawings. The primitive identification approach reconstructs a 3D solid on the basis of some basic primitives (prism, cylinder, sphere) identified in a line drawing (Wang & Grinstein, 1989).

2.2.2. Multi-view Approach

This type of approaches performs the 3D reconstruction process from three orthographic views of an object. These approaches are conceptually different from single-view approaches as these collect the geometric information from multiple views of an object and correlate these pieces of information with each other to form a 3D solid object. On the other hand, single-view approaches try to perceive/realize the third dimension from a single set of geometric information gathered from a view. Multi-view approaches can be further divided into two categories as listed below:

i) Constructive Solid Geometry (CSG) Approach
ii) Boundary Representation (B-Rep) Approach

Both B-Rep and CSG approaches are described in the next two sections.

2.2.2.1. Constructive Solid Geometry

The constructive solid geometry (CSG) approach tries to construct a solid object by applying set operations (Union, Intersection, and Difference) on some basic primitives/shapes (Aldefeld, 1983). It is just like combining different pieces of a puzzle to generate a desired shape. The basic primitives may be cuboid, pyramid, cylinder, cone, sphere, toroid, etc. The basic primitives are represented at leaf nodes and the set operations are performed at parent/internal nodes of a binary tree (see Figure 3). One set operation is performed at a time on two basic primitives/shapes, so a binary tree is constructed and by performing the set operations recursively in bottom up parsing manner, yields a solid object at the root of the tree.

The CSG representations of solid objects always produce either a solid object or an empty set (i. e., a set having no solid object) because intersection of two non-interacting shapes would produce a null object.

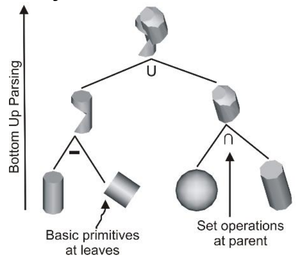

Figure 3: CSG Primitives, Set Operations and Bottom up Parsing

The CSG approaches always suffer from a major drawback because the topological information is not stored in the nodes of the tree. This drawback is inherent in these approaches because they can not predict the relative position of the primitives while performing the set operations. The binary tree, although contains the possible sequence of the set operations to be performed on the primitives during bottom up parsing, but it does not contain or can not compute the position of these primitives with respect to each other. This position information is important to build the desired shape otherwise same primitives may generate different solid objects (see Figure 4). In this Figure, the Union operation on the primitives cone and cuboid is performed at different positions relative to one another and the resultant solid objects are different although the primitives and the operation is same.

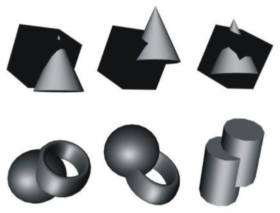

Figure 4: Missing Topological Information in CSG

There is another problem with the CSG approaches that the set operations do not hold the closure property over the CSG domain of primitives.

Aldefeld initially started work on the 3D reconstruction using CSG but this approach is able to handle isolated objects only and requires heavy user interaction (Aldefeld 1983). Then Aldefeld and Richter enhanced Aldefeld's previous approach by eliminating the restriction of isolated objects (Aldefeld & Richter,1984)). Cicek and Gulisen further enhanced the previous approaches by permitting the drawings with hidden lines (Cicek & Gulisen, 2004). This approach made provisions to reconstruct holes and cavities. Dimri and Guromoorthy performed the reconstruction from drawings with sectional views (Dimri & Guromoorthy, 2005).

2.2.2.2. Boundary Representation Approach

In the boundary representation (B-Rep) approach, a solid object is represented by its bounding surfaces (Idesawa, 1973). This approach uses two types of information for the boundary representation, which are topology and geometry. The main topological parameters that are used by this approach are faces, edges and vertices, while the geometry parameters include surfaces, curves and points, which are shown in Figure 5. A face is a bounded portion of a surface, an edge is a bounded piece of a curve, and a vertex is a point. The topology records connectivity of the faces, edges and vertices, while the geometry describes the exact shape and position of each edge, face and vertex. The geometry of a vertex is its position in the space as by its (x, y, z) coordinates. Edges can be straight lines or curves which can be approximated in poly-lines. A face is represented by a set of close curves on a surface. Several curves may lie on a face to represent cavities in a solid object. Different faces interact with each other at the common vertices to inscribe the whole object. It is just like a volume comprising of a set of faces having associated topological information that defines the relationship between these faces.

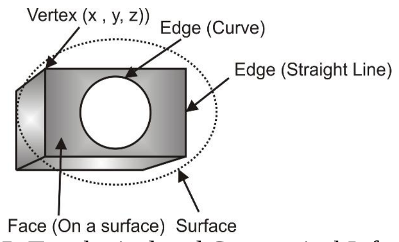

Figure 5: Topological and Geometrical Information in a B-Rep

There are four (4) major steps involved in the B-Rep approach, which are listed as follows:
i) Identification of vertices,
ii) Correspondence of these vertices to form an edge,
iii) Registration of these edges to form a face,
iv) Correspondence of these faces on a surface to form a volume

Idesawa has provided mathematical foundations to the B-Rep, and has set different criteria on the basis of which 3D reconstruction could be performed. But the major problem with this approach is that it can generate ghost figures (Idesawa, 1973). The problem of ghost figures was resolved by Wesley and Markowsky and they formalized the Idesawa's approach with algorithms (Wesley & Markowsky, 1981). Their approach is limited to polyhedrons only. Sakurai extended Wesley and Markowsky's work by incorporating cylinders and cones in addition to polyhedrons but these objects are handled with specific orientation (Sakurai, 1983). This approach is limited to cylinders and cones with axes parallel to one of the coordinate axes only. Moreover, this approach is unable to handle intersecting objects. Gu et al further extended Sakurai's approach to add more orientations of cylinders and cones with the condition that the axis should be parallel to one of the coordinate planes (Gu et al 1986). Their approach is also capable of handling ellipses, parabolas and fourth order curves. Another power of this approach is that it can handle intersecting objects, as well. Liu et al proposed an approach which is independent of orientation, and it can also incorporate quadrics (Liu et al, 2000). Cohen has used undirected graphs for 3D reconstruction but their approach requires heavy storage (Cohen, 2007). Suh and McCasland have reconstructed 3D solid objects from isometric views (Suh & McCasland, 2009).

3. Proposed Reconstruction Approach

In Figure 6, overall working of the proposed approach is shown. There are three (3) main steps in the approach, and they are: *2D Operations, 2D to 3D Transformations,* and *DXF Conversion* (see Figure 6).

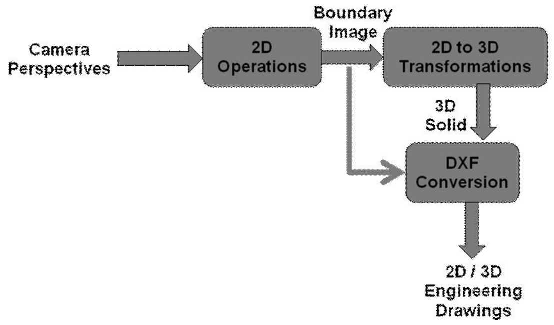

Figure 6: Block Diagram of Proposed Approach

Step 1 (2D Operations) performs the 2D operations on a camera perspective to get an image

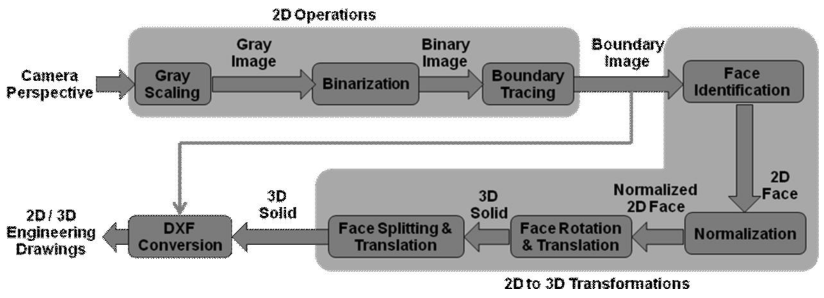

Figure 7: Detailed working of the Proposed Approach

with boundaries, and the image is referred to as a boundary image. Note that three (3) views (i.e., top, front and side) of an engineering object are captured as three corresponding boundary images in Step 1. Step 2 (2D to 3D Transformations) transforms the three boundary images (output of Step 1) of an engineering object into a 3D solid object. Step 3 (DXF Conversion) performs one of the two functions, a single boundary image (from Step 1) is vectorized to generate a 2D engineering drawings, or a 3D solid object (form Step 2) is vectorized to generate a 3D engineering drawing. The detailed working of these three main steps is further shown in Figure 7.

3.1. Step 1: 2D Operations

This step performs three (3) tasks, which are referred to as Gray Scaling, Binarization, and Boundary Tracing. The working of these three tasks we give in the next three (3) sections.

3.1.1. Task 1: Gray Scaling

As it has been mentioned earlier that the proposed approach starts its working by capturing camera perspectives (2D orthographic projections of the object in x-y plane) for each view (*top, front* and *side*) of an engineering object (see Figure 8 (a)), and these three views are converted into gray scale images using weighted average of red (R), green (G) and blue (B) colors for each pixel using Equation (3).

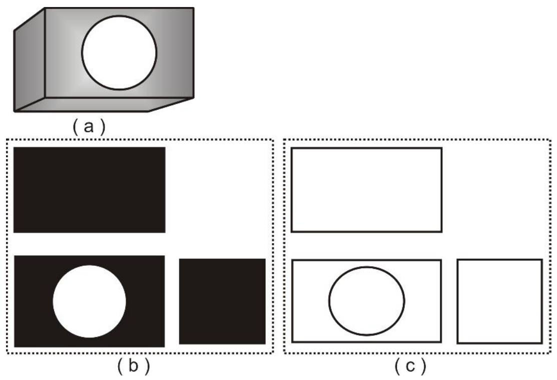

Figure 8: Binarization and Boundary Tracing (a) Engineering Object (b) Binarization (c) Boundary Tracing

$$Gray = 0.3R + 0.59G + 0.11B \qquad (3)$$

3.1.2. Task 2: Binarization

The gray images (from Task 1) are then, binarized to get pure black and white binary images as shown in Figure 8 (b). The algorithm used to perform this task (Binarization) is given in Figure 9.

```
Function Binarize (GrayImage): BinaryImage
{
for each row of GrayImage a: i
    for each column of GrayImage a: j
        if a[i][j] <=threshold
        then b[i][j]=0
        else b[i][j]=255
return b
}
Call Binarize for each GrayImage
```

Figure 9: Algorithm for Binarization

3.1.3. Task 3: Boundary Tracing

This task traces the boundaries from the binary image (from Task 2) to get a boundary image. Figure 8 (c) shows these boundary images corresponding to each of the top, front and side views. These boundary images are vectorized using Step 3 of the proposed approach to generate a 2D engineering drawing, or these images are further processed (Step 2 - Task 1) to identify faces for the 3D reconstruction so that 3D engineering drawings can be generated. 2D engineering drawings are useful for the conventional CAD/CAM applications while on the other hand, 3D engineering drawings are needed in the case of CNC machines and RP applications. This makes the proposed approach useful for a wide range of the engineering industry applications.

3.2. Step 2: 2D to 3D Transformations

This step consists of four (4) tasks; i.e., Face Identification, Normalization, Face Rotation and Translation, and Face Splitting and Translation. The working of all these tasks is explained in the next four sections.

3.2.1. Task 1: Face Identification

This task identifies faces from boundary images extracted in Step 1 of the proposed approach (see Section 3.1). Before we proceed with the description of this step (Face Identification), it is necessary to mention that the previous approaches identify faces in a three step process which makes these approaches computationally expensive (Wang & Grinstein, 1993; Fahiem, 2009). The three steps in previous approaches are; identification of vertices, correspondence of these vertices to form an edge, and registration of these edges to form a face.

To reduce the computations, we collect the faces directly in our Face Identification step instead of collecting vertices, edges and faces (as is in previous approaches). For that, we use the concept of tangential lines (Fahiem, 2008). Tangential lines are basically straight lines of infinite length which scan the image from each side (left, top, right and bottom) in the search of a black pixel. We need four tangential lines (TL_1, TL_2, TL_3, and TL_4), one for each of the four sides of a view. These lines start scanning the image row-wise as well as column-wise. The line TL_1 starts scanning the image column-wise from the left most column in the search of a black pixel. If a black pixel is found, then the scanning stops otherwise the line TL_1 scans the next column to the right of previous column in the search of a black pixel. This process continues until a black pixel is encountered. Similarly, the line TL_3 scans the image column-wise from the right most column in the search of a black pixel. It moves to the left until a black pixel is found. The lines TL_2 and TL_4 scan the image row-wise from top and bottom in the search of a black pixel, respectively. The line TL_2 moves towards bottom and the line TL_4 moves towards top until a black pixel is encountered. When this scanning process completes, the tangential lines intersect each other and form a bounding box on a view inscribing a 2D face, as shown in Figure 10. The intersection of TL_1 and TL_2 determines the upper-left corner of the bounding box. Similarly TL_2 and TL_3, TL_3 and TL_4, and TL_4 and TL_1 determine the upper-right, bottom-right, and bottom-left corners of the bounding box, respectively. Now the face is identified by the (x, y) coordinates produced by the intersections of the tangential lines (see Figure 10).

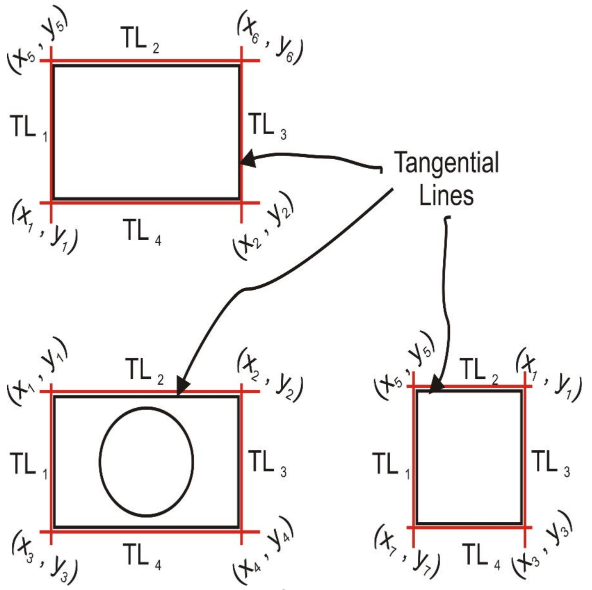

Figure 10: Face Identification using Tangential Lines

The working of Task 1: Face Identification in an algorithm is given in Figure 11.

```
Function FaceIdentify (BoundaryImage): 2DFace
{
BoundaryImage b
while PIXEL!=BLACK
        move $TL_1$ on b from LEFT
        move $TL_2$ on b from TOP
        move $TL_3$ on b from RIGHT
        move $TL_4$ on b from BOTTOM
Mark upper-left corner of bounding box at intersection of $TL_1$ and $TL_2$
Mark upper-right corner of bounding box at intersection of $TL_1$ and $TL_2$
Mark bottom-right corner of bounding box at intersection of $TL_1$ and $TL_2$
Mark bottom-left corner of bounding box at intersection of $TL_1$ and $TL_2$
Collect 2D face inscribed in bounding box
return 2DFace
}
Call FaceIdentify for each BoundaryImage
```

Figure 11: Algorithm for Face Identification

3.2.2. Task 2: Normalization

After the 2D faces identification (see Step 2 - Task 1), these faces are normalized so that a correct alignment of these faces with each other can be carried out, and it is done in the next step (Step 2 – Task 3). Three parameters, i.e., size, angle and perspective, of the 2D faces are normalized to make them size, angle and perspective invariant as it is explained in the following paragraph.

For the size normalization, the faces are scaled such that length of the top face is equal to length of the front face; width of the top face is equal to length of the side face; and width of the side face is equal to width of the front face. The faces are made angle and perspective invariant by adjusting the orientation of these faces such that the tangential lines TL_1 and TL_3 are parallel to y-axis and the tangential lines TL_2 and TL_4 are parallel to x-axis. The algorithm for this step is given in Figure 12.

```
Function Normalization (2DFace): Normalized2DFace
{
2DFace Top, Front, Side
scale Top, Front, Side such that
        TopLength=FrontLength
        TopWidth=SideLength
        SideWidth=FrontWidth
rotate & shear Top, Front, Side such that
                $TL_1$ & $TL_3$ // y-axis
                $TL_2$ & $TL_4$ // x-axix
return Normalized2DFace
}
```

Figure 12: Algorithm for Normalization

3.2.3. Task 3: Face Rotation and Translation

The normalized 2D faces produced by the previous step are still in the 2D x-y plane as the original projections (*top, front* and *side*) were. Before we do the reconstruction process of the 3D solid object, it is necessary that the top face should be in x-z plane and the side face should be in the y-z plane so that these faces could be aligned with the sides of the hypothetical cuboid (HC). To achieve this objective, the top face is rotated at -90^0 about the x-axis to transforms it into the x-z plane from the x-y plane.

Similarly, the side face is rotated at -90^0 about the z-axis to transform it into the y-z plane from the x-y plane. These rotations of the top and side faces are performed using Equation (4) and Equation (5), respectively (Gonzales & Woods, 2008).

$$x' = x\cos\theta - y\sin\theta \quad , \quad z' = x\sin\theta + y\cos\theta \qquad (4)$$

$$y' = x\cos\theta - y\sin\theta \quad , \quad z' = x\sin\theta + y\cos\theta \qquad (5)$$

Moreover, we introduce the third dimension as well, by setting the x, y and z coordinates to 0 for the side, top and front faces, respectively. At this stage, although the reconstructed solid object has 3D coordinates but the object lacks in the depth as the third dimension is 0.

The above process converts the 2D faces into 3D faces (see Figure 13(a)). This figure shows the top, front and side faces of the object after rotations and the introduction of the third dimension.

Now the dimensions of HC are determined. The length (L_{HC}), height (H_{HC}) and depth (D_{HC}) of HC are determined by Equation (6), Equation (7) and Equation (8), respectively.

$$L_{HC} = \lvert x_2 - x_1 \rvert \qquad (6)$$

$$H_{HC} = \lvert y_3 - y_1 \rvert \qquad (7)$$

$$D_{HC} = \lvert z_5 - z_1 \rvert \qquad (8)$$

After determining and setting the dimensions of the HC, the faces are translated to align the sides of HC so that a 3D solid object can be reconstructed. To do this, the top face is translated into x-z plane along x-axis and y-axis of the HC using Equation (9). The side face is translated in y-z plane along y-axis of the HC using Equation (10) (Gonzales & Woods, 2008).

$$x' = x - x_{1(top)} \quad , \quad y' = y + y_{1(front)} \qquad (9)$$

$$y' = (y - y_{1(side)}) + y_{1(front)} \qquad (10)$$

In this way, the coordinates (0, y_1, z_1), (x_1, y_1, 0), (x_1, 0, z_1) of the top, front and sides faces (without depth – third dimension) are aligned with each other to produce a 3D coordinate (x_1, y_1, z_1) with the depth. Similarly, other 3D coordinates (without depth) are also aligned with each other and 3D coordinates (with depth) are generated for whole of the solid (see Figure 13 (b)). The algorithm of this task (Face Rotation and Translation) is given in Figure 14.

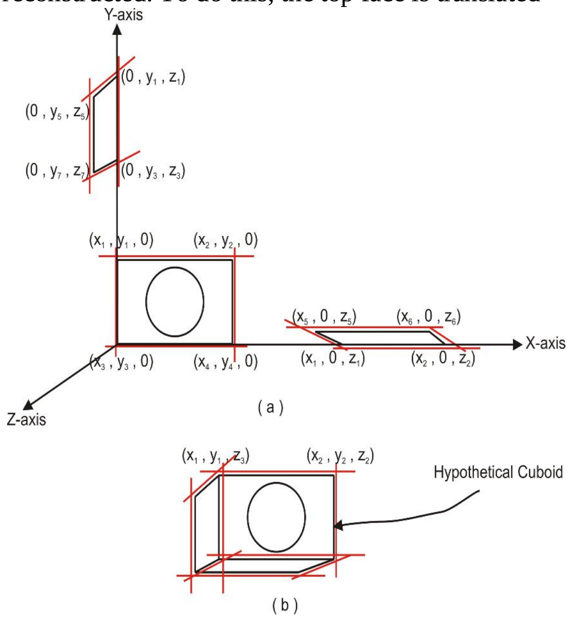

Figure 13: Face Rotation and Translation (a) Face Rotations (b) Face Translations on Hypothetical Cuboid

```
Function FaceRT (Normalized2DFace):
3DSolid
{
Normalized2DFace Top, Front, Side
rotate Top at -90⁰ about x-axis
rotate Side at -90⁰ about z-axis
translate Top in x-z plane along x-axis of HC
translate Top in x-z plane along y-axis of HC
translate Side in y-z plane along y-axis of HC
return 3DSolid
}
```

Figure 14: Algorithm for Face Rotation and Translation

3.2.4. Task 4: Face Splitting and Translation

The 3D solid object that has been reconstructed by the previous task (Step 2 – Task 3) may not be a true 3D solid because it may contain protruding faces. A protruding face is basically a part of a face which is projected beyond its plane as shown in Figure 15(a). Here right hand parts of top and front faces are basically protruding from their planes. Due to these protruding faces, the 3D solid object reconstructed in the previous task (Step2 – Task 3) is not a true 3D solid (see Figure 15(b)). These protruding faces are not translated to their correct positions and are hanging just like a cantilever beam. We will call such protruding hanging faces as simply a 'hanging face'. A hanging face is splitted and translated to the next edge of the front face to form a true 3D solid object (see Figure 15(c)) using algorithm of 'Face Splitting and Translation' as presented in Figure 16.

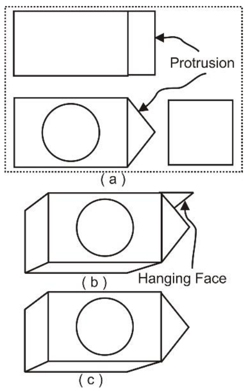

Figure 15: Face Splitting and Translation (a) 2D Faces (b) 3D Solid with Hanging Face (c) 3D Solid Object after Face Splitting and Translation

```
Function FaceST (Normalized2DFace, 3DSolid): 3DSolid
{
Normalized2DFace Top, Front, Side
3DSolid S
subtract Top from x-z plane of S
translate remaining in y-direction to next edge
subtract Front from x-y plane of S
translate remaining in z-direction to next edge
Subtract Side from y-z plane of S
translate remaining in x-direction to next edge
return 3DSolid
}
```
Figure 16: Algorithm for Face Splitting and Translation

This algorithm (see Figure 16) takes normalized 2D top, the front and side faces (from Step 2 – Task 2) and 3D solid (from Step 2 – Task 3) as input, and subtracts these faces from this solid object, and splits the hanging faces from the base face. These hanging faces, after splitting, are translated to the next edge. The top hanging face is translated to the next edge of front face of the 3D solid object. Similarly, other hanging faces (front and side) are also translated to the next edge of the 3D solid object. The output of the proposed algorithm is a complete 3D solid object.

3.3. Step 3: DXF Conversion

In the last step of the proposed approach, the 3D solid object is vectorized to generate a 3D engineering drawing. The vectorization is performed in drawing exchange file (DXF) format recognized by most of the CAD tools (Fahiem & Farhan, 2007). Typically, the DXF format comprises of eight (8) sections as it has been described in Table 1.

Table 1: Sections in DXF Format

Section	Description
Header	Contains the drawing's general information
Classes	Contains the application specific class definitions
Tables	Contains the information about Layer, Line type, etc.
Blocks	Contains the information about blocks
Entities	Contains drawing entities such as Line
Objects	Contains the application specific data
Thumbnail Image	Contains the preview image for the DXF
End of File	Marks the end of DXF

In our proposed approach, the vectorization process needs 'Line Entity' to model different shapes in a 3D solid object with 3D coordinates (generated in Step 2). Different entities in DXF format are specified by their numeric codes and the code for 'Line Entity' is 6. A line is specified by two 3D points with numeric codes for (x, y, z) coordinates of the starting and ending points. These codes are shown in Table 2.

Table 2: Numeric Codes for Line Starting and Ending Points

Point	Coordinate	Numeric Code
Starting	x	10
	y	20
	z	30
Ending	x	11
	y	21
	z	31

A typical line with (x_1, y_1, z_1) and (x_2, y_2, z_2) coordinates is represented as 6 10 x_1 20 y_1 30 z_1 11 x_2 21 y_2 31 z_2 in DXF format. We have converted the 3D solid object with the 3D coordinates generated in Step 2 in the DXF format using 'Line Entity'.

The output of this step is an ASCII file in DXF format containing the numeric values of 3D coordinates corresponding to different lines forming a 3D solid object. This file can be loaded in a CAD tool to regenerate a 3D engineering drawing automatically, from the numeric values corresponding to 3D coordinates of a 3D solid object stored in the file.

4. Conclusion

The novelty of the proposed 3D reconstruction approach is its reconstruction of 3D solid objects form 2D camera perspectives of engineering objects. This feature is new and is not available in the existing approaches (Wang & Grinstein, 2008). This approach is useful in reconstructing and redesigning of engineering objects when 2D drawings are not available for some reason. The approach is a reverse engineering process and helpful in technology transfer, and is effective in terms of cost and time. This can also be helpful to engineering industry as it supports modern trends in manufacturing, machining and production of engineering equipments.

Corresponding Author:
Muhammad Abuzar Fahiem
Department of Computer Science and Engineering
University of Engineering and Technology
Lahore 54000, Pakistan
E-mail: abuzar@uet.edu.pk

References

1. Gingold, Y., Igarashi, T., and Zorin, D. 2009. Structured Annotations for 2D-to-3D Modeling. ACM Transactions on Graphics. 28(5): 1-9.
2. Sparacino, F., Wren, C., Pentland, A., and Davenport, G. 1995. HyperPlex: a World of 3D Interactive Digital Movies. Proceedings of International Joint Conference on Artificial Intelligence.
3. W. R., Roth, G., and Rivest, J. F. 2003. View Planning for Automated Three-Dimensional Object Reconstruction and Inspection. ACM Computing Surveys. 35(1): 64-96.
4. Bernardini, F., and Rushmeier, H. 2002. The 3D Model Acquisition Pipeline. Computer Graphics Forum. 21(2): 149-172.
5. Wang, W., and Grinstein, G. G. 1993. A Survey of 3D Solid Reconstruction from 2D Projection Line Drawings. Computer Graphics Forum. 12 (2): 137-158.
6. Company, P., Piquer, A., Contero, M., and Naya, F. 2005. A Survey on Geometrical Reconstruction as a Core Technology to Sketch-Based Modeling. Computers and Graphics. 29 (6): 892-904.
7. Fahiem, M. A. 2009. 3D Reconstruction of Solid Models from 2D Camera Perspectives. Lecture Notes in Electrical Engineering. 27: 241-250.
8. Rusinkiewicz, S., Hall-Holt, O., and Levoy, M. 2002. Real-Time 3D Model Acquisition. ACM Transactions on Graphics. 21(3): 438-446.
9. Fleck, S., Busch, F., Biber, P., and Strasser, W. 2009. Graph Cut Based Panoramic 3D Modeling and Ground Truth Comparison with a Mobile Platform – The Wagele. Image and Vision Computing. 27: 141–152.
10. Blais, F., Beraldin, J. A., El-Hakim, S., and Godin, G. 2003. New Developments in 3D Laser Scanners: From Static to Dynamic Multi-Modal Systems. Proceedings of 6th Conference on Optical 3-D Measurement Techniques.
11. Xu, Y., Xu, C., Tian, Y., Ma, S., and Luo, M. 1998. 3-D Face Image Acquisition and Reconstruction System. Proceedings of IEEE Instrumentation and Measurement Technology Conference.
12. Dipanda, A., and Woo, S. 2004. Structured Light System Configuration Determination for Efficient 3D Surface Reconstruction. Proceedings of International Conference on Image Processing.
13. Fofi, D., Salvi, J., and Mouaddib, E. M. 2003. Uncalibrated Reconstruction: An Adaptation to Structured Light Vision. Pattern Recognition. 36(7): 1631-1644.
14. Cooper, M. C. 2005. Wireframe Projections: Physical Realisability of Curved Objects and Unambiguous Reconstruction of Simple Polyhedra. International Journal of Computer Vision. 64 (1): 69-88.
15. Martin, R. R., Suzuki, H., and Varley, P. A. C. 2005. Labeling Engineering Line Drawings Using Depth Reasoning. Journal of Computing and Information Science in Engineering. 5(2): 158-167.
16. Feng, D., Lee, S., and Gooch, B. 2006. Perception-Based Construction of 3D Models from Line Drawings. Proceedings of ACM SIGGRAPH International Conference on Computer Graphics and Interactive Techniques.
17. Kyratzi, S., and Sapidis, N. 2009. Extracting a Polyhedron from a Single-View Sketch: Topological Construction of a Wireframe Sketch with Minimal Hidden Elements. Computers and Graphics. 33(3): 270-279.
18. Huffman, D. A. 1971. Impossible Objects as Nonsense Sentences, Machine Intelligence. 6: 295-323.
19. Clowes, M. B. 1973. On Seeing Things. Artificial Intelligence. 2(1): 79-116.
20. Macworth, A. K. 1973. Interpreting Pictures of Polyhedral Scenes. Artificial Intelligence. 4(2): 121-137.
21. Sugihara, K. 1986. Machine Interpretation of Line Drawings. The MIT Press.
22. Lamb, D., and Bandopadhay, A. 1990. Interpreting a 3D Object from a Rough 2D Line Drawing. Proceedings of the 1st IEEE conference on Visualization.
23. Idesawa, M. 1973. A System to Generate a Solid Figure from Three Views. Bulletin of Japan Society of Mechanical Engineers. 16(92): 216-255.
24. Wesley, M. A., and Markowsky, G. 1981. Fleshing out Projections. IBM Journal of Research and Development. 25(6): 934-954.

25. Sakurai, H. 1983. Solid Model Input Through Orthographic Views. Computer Aided Design. 17(3): 243-252.
26. Gu, K., Tang, Z., and Sun, J. 1986. Reconstruction of 3D Objects from Orthographic Projections. Computer Graphics Forum. 5(4): 317-323.
27. Liu, S., Hu, S., Tai, C., and Sun, J. 2000. A Matrix-Based Approach to Reconstruction of 3D Objects from Three Orthographic Views. Proceedings of 8th Pacific Conference on Computer Graphics and Applications.
28. Cohen A. B. 1995. Mechanical Engineering in the Information Age. Mechanical Engineering. 117(12): 66-70.
29. Suh, Y. S., and McCasland, J. 2009. Interactive Construction of Solids from Orthographic Multiviews for an Educational Software Tool. Computer-Aided Design and Applications. 6(2): 219-229.
30. Fahiem, M. A. 2008. A Deterministic Turing Machine for Context Sensitive Translation of Braille Codes to Urdu Text. Lecture Notes in Computer Science. 4958: 342-351.
31. Gonzalez, R. C., and Woods, R. E. 2008. Digital Image Processing. Prentice Hall.
32. Aldefeld, B. 1983. Automatic 3D Reconstruction from 2D Geometric Part Descriptions. Proceedings of IEEE Conference on Computer Vision and Pattern Recognition.
33. Aldefeld, B., and Richter, H. 1984. Semiautomatic Three-Dimensional Interpretation of Line Drawings. Computer-Aided Design. 8(4): 371-380.
34. Cicek, A., and Gulesin, M. 2004. Reconstruction of 3D Models from 2D Orthographic Views Using Solid Extrusion and Revolution. Journal of Material Processing Technology. 152(3): 291-298.
35. Dimri, J., and Gurumoorthy, B. 2005. Handling Sectional Views in Volume-Based Approach to Automatically Construct 3D Solid from 2D Views. Computer-Aided Design. 37(5): 485-495.
36. Fahiem, M. A., and Farhan, S. 2007. Representation of Engineering Drawings in SVG and DXF for Information Interchange. Proceedings of 6th WSEAS International Conference on Circuits, Systems, Electronics, Control & Signal Processing.
37. Wang, W., and Grinstein, G. G. 2008. A Survey of 3D Solid Reconstruction from 2D Projection Line Drawings. Computer Graphics Forum. 12(2): 137-158.

6/18/2010