Single View Metrology

Desmond Tsoi Yau Chat and Stanley Ng Ho Lun
Department of Computer Science
Hong Kong University of Science and Technology
{desmond, stan}@cs.ust.hk

Abstract

We describe how 3D affine measurements may be computed from a single perspective view of a scene given only minimal geometric information determined from the image. This minimal information is typically the vanishing line of a reference plane, and a vanishing point for a direction not parallel to plane. It is shown that affine scene structure may then be determined from the image, without knowledge of the camera's internal calibration (e.g. focal length), nor of the explicit relation between camera and world (pose).

In particular, we show how to compute (i) the three sets of vanishing points in X, Y, Z directions; (ii) the vanishing lines of the image; (iii) distance between planes that parallel to the reference plane (up to a common scale factor); (iv) image wrapping. Simple geometric derivations are given for these results. We also develop an algebraic representation which unifies the four types of measurement.

We demonstrate the technique for a variety of applications, including height measurements in forensic images and 3D graphical modeling from single images.

Keywords: 3D reconstruction, photogrammetry, image warping.

I. Introduction

In this paper we describe how aspects of the affine 3D geometry of a scene may be measured from a single perspective image. We will concentrate on scenes containing planes and parallel lines.

It is assumed that images are obtained by perspective projection. In addition, we assume that the vanishing points in X, Y, Z direction can be computed from the image. After that, we choose two of them to form the vanishing line and define the reference plane. Then, we need to provide the reference length in X, Y, Z direction. Finally, we can compute the distance between any two reference planes. The measurement method developed is independent of the camera's internal parametric: focal length, aspect ratio, principal point, skew.

We begin in section 2 to give you some geometry background on project geometry. Then in section 3, we describe how to compute the projective matrix P. In section 4, we give some technical details on this project. For instance, hints on how to choose an appropriate input image, which three-dimensional affine information could be extracted from the image; Follow up, we introduce how the vanishing points could be computed from the set of parallel lines. We also illustrate how to compute the vanishing lines from the vanishing points. Then we need to define a reference point as the origin in the world coordinate. After the reference point is defined, we can perform the 3D position calculation based on the 2D image points. Finally, we create a VRML model to demonstrate our work. In section 5, technical difficult are discussed. Summary and conclusion are drawn in section 6.

II. Background geometry and notation

2.1 Introduction

Projective Geometry gives us the mathematical relation between points in 2D image plane and points in 3d world space. In this section, we are going to illustrate the notation and specific details of Projective Geometry, which will be useful in the next section.

2.2 Notation

In the paper, use the following notations for derivation purpose:

- 3D points in general position are vectors that denoted by bold capital letters (e.g. $\mathbf{X}$)
- Image positions are vectors that denoted by bold small letter (e.g. $\mathbf{x}_1$)
- Scalars by normal face symbols (e.g. d)
- Matrices by typewriter style capitals (e.g. $\mathbf{\Pi}, \mathbf{H}$)
- The line through two points $\mathbf{x}_1$ and $\mathbf{x}_2$ is denoted by $< \mathbf{x}_1\mathbf{x}_2 >$

2.3 Pinhole Camera Model

The most general linear camera model is the well-known central projection (pinhole camera).

Description

A 3D point in space is projected onto the image plane by means of straight visual rays. The corresponding image point is the intersection of the image plane with the visual ray connecting the optical center and the 3D point. (c.f. Leonardo's Perspecograph in figure 2.4.2 and the schematic pinhole camera model in figure 2.4.3).

Algebraic interpretation

The projection of a world point **X** onto the image point **x** (figure 2.4.3) is described by the following equation:

$$\mathbf{x} = \mathbf{PX}$$

Where P is the projection matrix, a 3 x 4 homogenous matrix, and "=" is equality up to scale. The world and image points are represented by homogenous vectors as $\mathbf{X} = (X, Y, Z, W)^T$ and $\mathbf{x} = (x, y, w)^T$. The scale of the matrix does not affect the equation, so only the eleven degrees of freedom corresponding to the ratio of the matrix elements are significant.

The camera model is completely specified once the matrix **P** is determined. The matrix can be computed from the relative positioning of the world points and camera center, and from the camera internal parameters; however, it can also be computed directly from image-to-world point correspondences.

2.4 Planar Homography

An interesting specialization of the general central projection described above is a plane-to-plane projection that is a 2D-2D projective mapping.

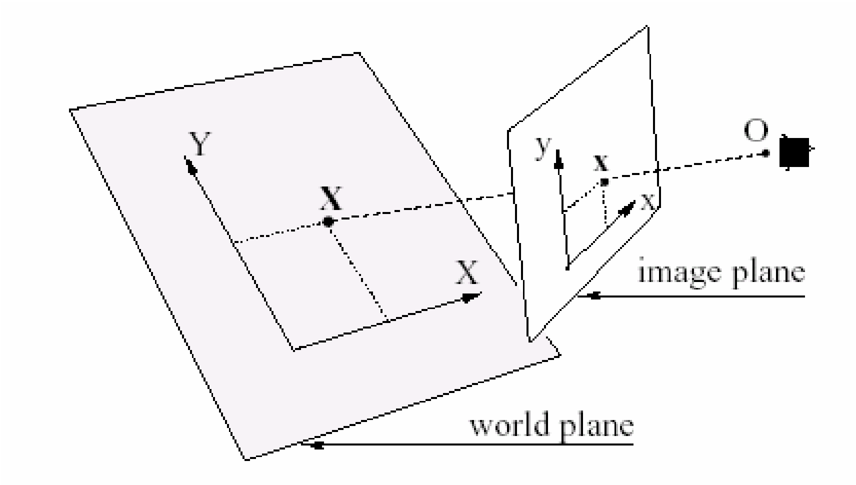


Figure 2.4.1: **Plane-to-plane camera model.**

A point **X** on the world plane is imaged as **x**. Euclidean coordinate X, Y and Z and x, y are used for the world and image planes, respectively. **O** is the viewer's position.

Description

The camera model for perspective image of planes, mapping points on a world plane to points on the image plane (and vice-versa) is well known. Points on a plane are mapped to points on another plane by a plane-to-plane homography, also known as a plane projective transformation. It is a bijective (thus invertible) mapping induced by the star of rays centered in the camera center (center of projection). Planar homographics arise, for instance, when a world planar surface is imaged.

Algebraic parameterization

A homography is described by a 3 x 3 non-singular matrix. Figure 2.4.1 shows the imaging process. Under perspective projection corresponding points are related by:

$$\mathbf{x} = \mathbf{H_i X}$$

Where H_i is the 3 x 3 homogenous matrix which describes the homography, and "=" is equality up to scale. The world and image points are represented by homogenous 3-vectos as $\mathbf{X} = (X, Y, W)^T$ and $\mathbf{x} = (x, y, w)^T$ respectively. The scale of the matrix does not affect the equation, thus only the eight degrees of freedom corresponding to the ratio of the matrix elements are significant.

Since we are interested in recovering world quantities from images and the homography is an invertible transformation, we get:

$$\mathbf{X} = \mathbf{H_i^{-1} x}$$

which is the homography mapping image points into world points. In particular, it can be shown that at least four world-to-image feature (point or line) correspondences (no three points collinear or no three lines concurrent) suffice to define the homography. The relative geometric position of the world features must be known. The computation is described in section 4.6.

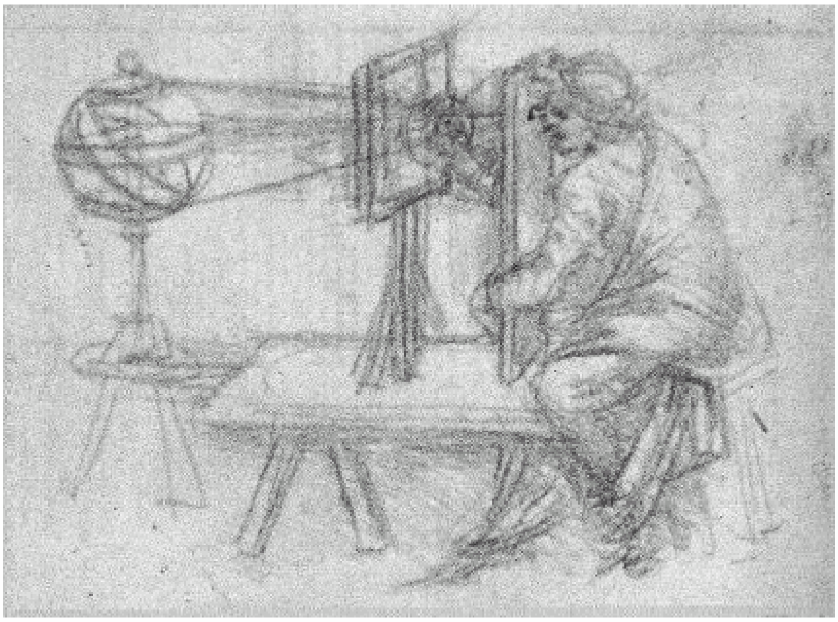


Figure 2.4.2: **Leonardo's perspectograph**.
"The things approach the point of the eye in pyramids, and these pyramids are intersected on the glass plane" Leonardo da Vinci (1452-1519), A.l v.

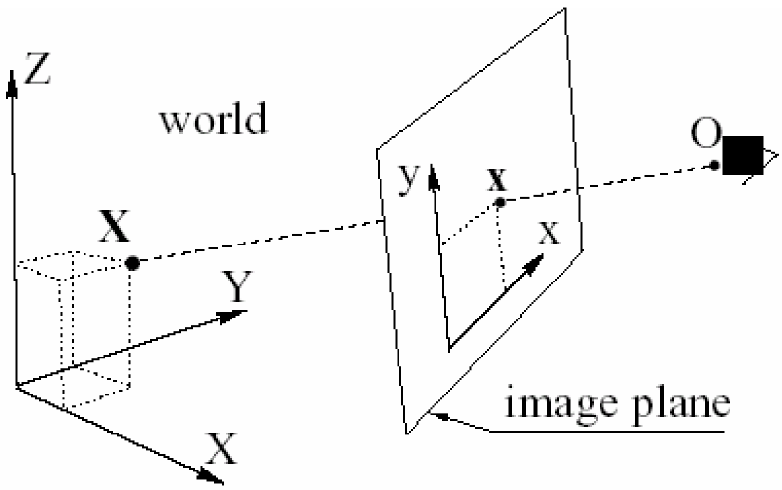


Figure 2.4.3: **Pinhole camera model.**
a point **X** in the 3d space is imaged as **x**. Euclidean coordinates X, Y, Z and x, y are used for the world and image reference system, respectively. **O** is the center of projection, the viewer.

2.5 Vanishing Points

Geometrically the vanishing point of a line is obtained by intersection the image plane with a ray parallel to the world line and passing through the camera center. Thus a vanishing point depends only on the direction of a line, not on its position. If the world line is parallel to the image plane then the vanishing point is at infinity in the image.

Algebraically the vanishing point may be obtained as a limiting point as follows: Points on a line in 3D space through the point ***A*** and with direction ***D*** are written as $\mathbf{x}(\lambda) = \mathbf{A} + \lambda\mathbf{D}$. As the parameter λ varies from 0 to ∞ the point $\mathbf{x}(\lambda)$ is imaged at

$$\mathbf{x}(\lambda) = P\mathbf{X}(\lambda) = P\mathbf{A} + \lambda P\mathbf{D} = \mathbf{a} + \lambda\mathbf{K}d$$

where ***a*** is the image of ***A***. Then the vanishing point ***v*** of the line is obtained as the limit

$$\mathbf{v} = \lim_{\lambda\to\infty} \mathbf{x}(\lambda) = \lim_{\lambda\to\infty}(\mathbf{a} + \lambda\mathbf{K}d) = \mathbf{K}d$$

From the above formula, we get $\mathbf{v} = \mathbf{K}d$. It means that the vanishing point **v** back-projects to a ray with direction d. Note that **v** depends only on the direction d of the line, not on its position specified by **A**.

The vanishing point of lines with direction d in 3-space is the intersection **v** of the image plane with a ray through the camera center with direction d, namely $\mathbf{v} = \mathbf{K}d$.

2.6 Vanishing Lines

Parallel planes in 3D space intersect plane at infinity π_∞ in a common line, and the image of this line is the vanishing line of the plane. Geometrically the vanishing line is constructed by intersecting the image with a plane parallel to the scene plane through the camera center. It is clear that a vanishing line depends only on the orientation of the scene plane; it doesn't depend on its position.

The set of planes perpendicular to the direction n in the camera's Euclidean coordinate frame have vanishing line $\mathbf{l} = \mathbf{K}^{-T}n$.

Since lines parallel to a plane intersect the plane at π_∞, it is easily seen that the vanishing point of a line parallel to a plane lies on the vanishing line of the plane.

III. How to extract 3D Information

To begin we define an affined coordinate system XYZ in space. Let the origin of the coordinate frame lie on the reference plane, with the X and Y –axes spanning the plane. The z-axis is the reference direction, which is thus any direction not parallel to the plane. The image coordinate system is the usual x-y affine image frame, and a point **X** in space is projected to the image point **x** via a 3x4 projection matrix **P** as:

$$\mathbf{x} = \mathbf{PX} = [\mathbf{p1} \quad \mathbf{p2} \quad \mathbf{p3} \quad \mathbf{p4}]\mathbf{X} \tag{3.1}$$

where **x** and **X** are homogenous vectors in the form: $\mathbf{x} = (x, y, w)^{\mathbf{T}}$, $\mathbf{X} = (X, Y, Z, W)^{\mathbf{T}}$, and "=" means equality up to scale.

If we denote the vanishing points for the X, Y and Z direction as (respectively) **Vx**, **Vy** and **Vz**, then it is clear by inspection that the first three columns of **P** are the vanishing points:

1. Projection of parallel line in X direction:

   $\mathbf{p_1} = \mathbf{P}[1 \quad 0 \quad 0 \quad 0]$

   $\mathbf{p_1} = \alpha_x \mathbf{v_x}$

   Where $\mathbf{p}_1$ is the 1st column in projective matrix **P**, α_x is the unknown scaling factor in x direction and $\mathbf{v}_x$ is the vanishing point in x direction.

2. Projection of parallel line in Y direction:

   $\mathbf{p}_2 = \mathbf{P}[0 \quad 1 \quad 0 \quad 0]$

   $\mathbf{p}_2 = \alpha_y \mathbf{v}_y$

   Where $\mathbf{p}_2$ is the 2nd column in projective matrix **P**, α_y is the unknown scaling factor in y direction and $\mathbf{v}_y$ is the vanishing point in y direction.

3. Projection of parallel line in Z direction:

   $\mathbf{p}_3 = \mathbf{P}[0 \quad 0 \quad 1 \quad 0]$

   $\mathbf{p}_3 = \alpha_z \mathbf{v}_z$

   Where $\mathbf{p}_3$ is the 3rd column in projective matrix P, α_z is the unknown scaling factor in z direction and $\mathbf{v}_z$ is the vanishing point in z direction.

The final column of $\mathbf{P}$ is the projection of the world coordinate system, $\mathbf{o} = \mathbf{p}_4$. Since our choice of coordinate frame has the X and Y axes in the reference plane $\mathbf{p}_1 = \mathbf{v}_x$ and $\mathbf{p}_2 = \mathbf{v}_y$ are two distinct points on the vanishing line. Choosing these fixes the X and Y affined coordinate axes. We denote the vanishing line by $\mathbf{l}$, and to emphasize that the vanishing points $\mathbf{v_x}$ and $\mathbf{v_y}$ lie on it.

Column 1, 2 and 4 of the projection matrix are the three columns of the reference plane to image homography. This homopgraphy must have rank three, otherwise the reference plane to image map is degenerate. Consequently, the final column (the origin of the coordinate system) must not lie on the vanishing line, since if it does then all three columns are points on the vanishing line, an thus are not linearly independent. Hence we set it to be

$$\mathbf{p}_4 = \frac{\mathbf{l}}{\|\mathbf{l}\|} = \bar{\mathbf{l}}, \mathbf{p}_4 = \frac{\mathbf{v}_x \times \mathbf{v}_y}{\|\mathbf{v}_x \times \mathbf{v}_y\|} = \mathbf{l} \qquad (3.2)$$

Therefore, the final parameterization of the projective matrix P is:

$$\mathbf{P} = \lfloor \alpha_x \mathbf{v_x} \quad \alpha_y \mathbf{v_y} \quad \alpha_z \mathbf{v_z} \quad \mathbf{l} \rfloor \qquad (3.3)$$

where $\alpha_x, \alpha_y, \alpha_x$ are the unknown scaling factor, which has an important role to play in the remainder of the paper.

Before calculating the 3D position of the 2D image points, we need to find the scaling factor. The scaling factors can be found by the given reference length. The detail is shown in the following section.

We wish to measure the distance between scene planes specified a point X and a point X' in the scene. These points may be chosen as respectively $\mathbf{X} = (X, Y, 0)^T$ and $\mathbf{X}' = (X, Y, Z)^T$, and their images are **x** and **x'**. If **P** is the projective matrix then the image coordinates are

$$\mathbf{x} = \mathbf{P}\begin{pmatrix} X \\ Y \\ 0 \\ 1 \end{pmatrix} \qquad \mathbf{x}' = \mathbf{P}\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{3.4}$$

The equations above can be rewritten as

$$\mathbf{x} = \rho(X\mathbf{p_1} + Y\mathbf{p_2} + \mathbf{p_4}) \tag{3.5}$$

$$\mathbf{x}' = \rho'(X\mathbf{p_1} + Y\mathbf{p_2} + Z\mathbf{p_3} + \mathbf{p_4}) \tag{3.6}$$

where ρ and ρ' are unknown scale factors, and $\mathbf{p_i}$ is the *ith* column of the **P** matrix.

Since $\mathbf{p_1} \cdot \bar{\mathbf{l}} = \mathbf{p_2} \cdot \bar{\mathbf{l}} = 0$ and $\mathbf{p}_4 \cdot \bar{\mathbf{l}} = 1$, taking the scalar product of (3.5) with $\bar{\mathbf{l}}$ yields $\rho = \bar{\mathbf{l}} \cdot \mathbf{x}$ and therefore (3.6) can be rewritten as

$$\mathbf{x}' = \rho\left(\frac{\mathbf{x}}{\rho} + \alpha Z \mathbf{v}\right) \tag{3.7}$$

By taking the vector product of both terms of (3.7) with **x'** we obtain

$$\mathbf{x} \times \mathbf{x}' = -\alpha Z \rho (\mathbf{v} \times \mathbf{v}') \tag{3.8}$$

And, finally, taking the norm of both sizes of (3.8) yields

$$\alpha Z = -\frac{\|\mathbf{x} \times \mathbf{x}'\|}{(\bar{\mathbf{l}} \cdot \mathbf{x})\|\mathbf{v} \times \mathbf{x}'\|} \tag{3.9}$$

Since αZ scales linearly with α, affine structure has been obtained. If α is known, then a metric value for Z can be immediately computed as:

$$Z = -\frac{1}{\alpha}\frac{\|\mathbf{x} \times \mathbf{x}'\|}{(\mathbf{p_4} \cdot \mathbf{x})\|\mathbf{p_3} \times \mathbf{x}'\|} \tag{3.10}$$

VI. Technical Details

4.1. Image Acquisition

Based on our project assumption, we need a 3-view perspective image. Therefore, we need to find an image that we can extract three vanishing points from sets of parallel lines in X, Y and Z direction of the world space as the input source. In the following section, we prove that 1-view perspective and 2-view perspective are not suitable in our project. The only valid image we want is a 3-view perspective.

4.1.1 One Point Perspective Image

One-point perspective is what you see when you look straight at the side of an object. It uses only one vanishing point, hence its name.

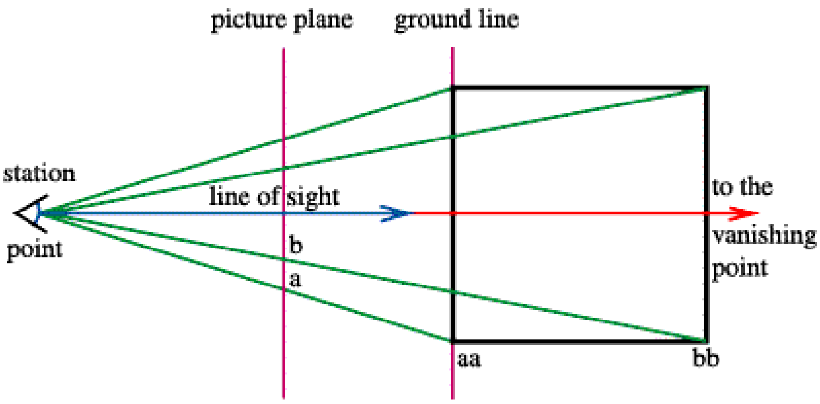


Figure 4.1.1: **One Point Perspective Diagram. Top View.**
It shows the top view of one point perspective. As you can see, the front plane of a cube is parallel to the image plane.

In figure 4.1.1, we find that the line of sight in one-point perspective is perpendicular (at right angle to) the side of the cube in these examples. That means you see the near side in plane view (actual shape undistorted by perspective).

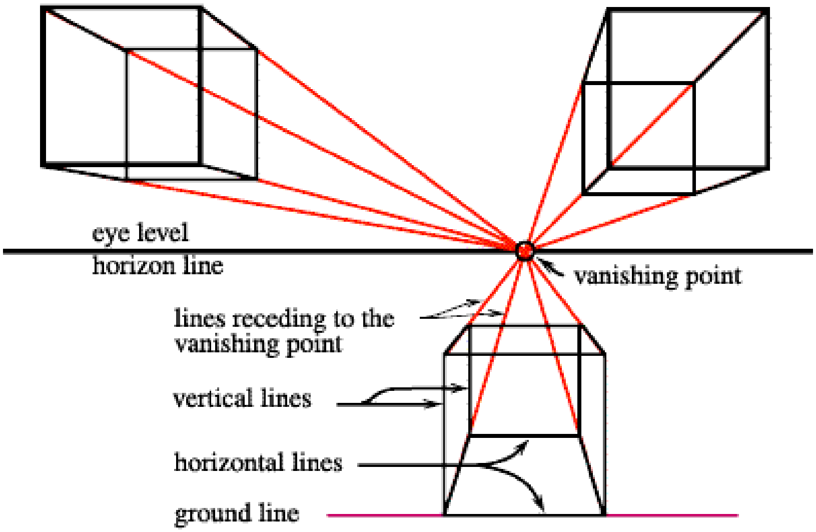


Figure 4.1.2: **One Point Perspective Diagram. 3d View.**
It shows the worldview of one point perspective. The most important thing to know is that there is only one intersection point and all the front plane of a cube is parallel to the image plane.

In figure 4.1.2, it shows that in one point perspective, there are only three kinds of lines used in one-point perspective. They are:

- **Vertical edges** are shown as **vertical lines.**
- **Horizontal edges** (perpendicular to the line of sight and parallel to the ground) are shown as **horizontal lines**.
- **Edges that recede** (are parallel to the line of sight) are on lines that **converge at the vanishing point** on the horizontal line.

In figure 4.1.3, it shows a one-point perspective image in our social life. As you can see in the figure, all the sets of imaged parallel line in the image intersect only one point. We call it vanishing point. If we use this image as our input source, then there is not enough information to calculate the projective matrix **P** and we cannot estimate the X, Y and Z coordinate in 3-space.

4.1.2 Two Points Perspective

Two-point perspective is used when you look at or into the corner of an object. There are two vanishing points since the two sets of sides are receding in two different directions.

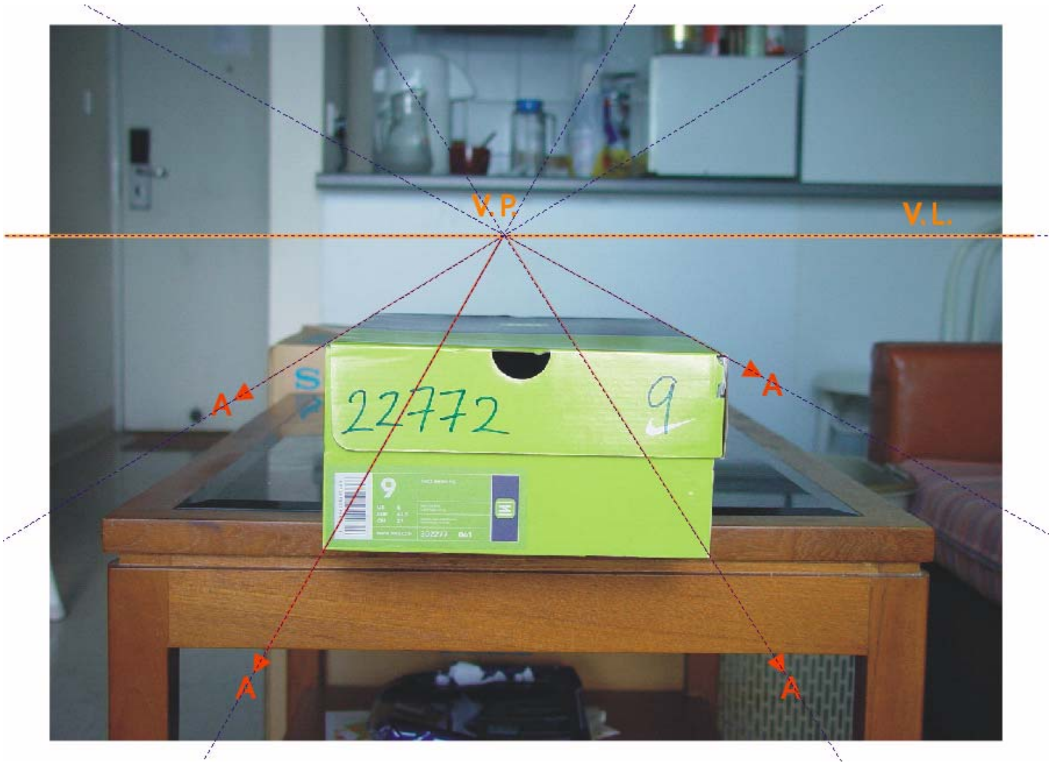


Figure 4.1.3: **One Point Perspective Image.**
In the image, you can notice that there are only one vanishing point. All the lines A in 3-spaces is parallel to each other. But, they are intersected into one common point when we look on the image plane. This point is called vanishing point.

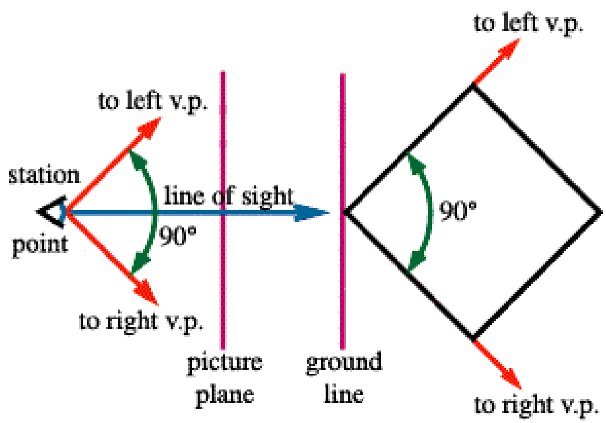


Figure 4.1.4: **Two Point Perspective Diagram. Top View.**
It shows the top view of two points perspective. As you can see, there are two set of parallel lines in 3-space are not parallel to the image plane.

In figure 4.1.4, we find that there are two sets of parallel lines in 3-space are not parallel to the image plane. Two vanishing points are computed by these two sets of imaged parallel lines. In the real world, these vanishing points are very far apart. Imagine strings are streaming out parallel to the edges of a cube going to the horizon as depicted in figure 4.1.5. The horizon is miles away so the vanishing points are many miles apart. When you draw them only a few inches apart on a piece of paper there is going to be some distortion in the image produced.

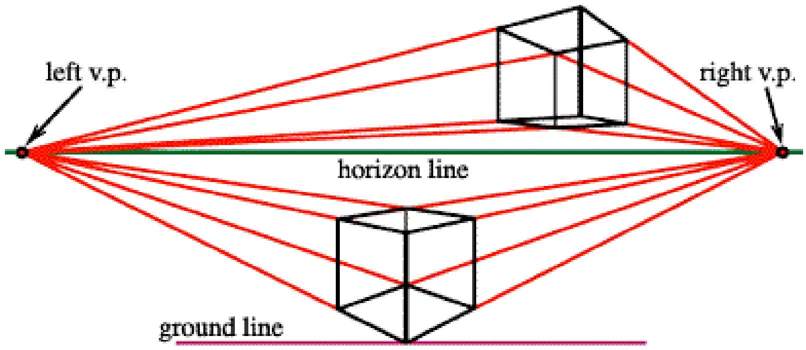


Figure 4.1.5: **Two Point Perspective Diagram. 3d View.**

It shows the worldview of two points perspective. The most important thing to know is that there are two points of intersection in the image plane, namely, left vanishing point and right vanishing point. Different from one point perspective, there is some distortion when you draw the object on the image plane.

Again there are only three different kinds of lines needed to draw in two-point perspective:

- **Vertical edges** are shown as **vertical lines.**
- **Edges of sides that recede toward the right are** on lines converging at the **right vanishing point.**
- **Edges of sides that recede toward the left are** on lines converging at the **left vanishing point.**

Both of the cubes in the example use only the same three kinds of lines. You see the top of the cube below the horizon line (your eye level). You see the bottom of the cube above the horizon line and more of its left side because it is to the right of your position in the center of the vanishing point.

In figure 4.1.6, it shows a two-point perspective image in our social life. There are two sets of parallel line in 3-space, which are not parallel to the image plane. Thus, an intersection point on each set of parallel lines is computed as vanishing point. We call it vanishing point. If we use this image as our input source, then there is not enough information to calculate the projective matrix **P** and we cannot estimate the X, Y and Z coordinate in 3-space.

Figure 4.1.6: **Two Points Perspective Image.**
In the image, you can notice that there are two vanishing points. All the lines A in 3-spaces are parallel to each other to form the right vanishing point. All the lines B in 3-spaces are parallel to each other to from the left vanishing point.

4.1.3 Three Points Perspective

Three-point perspective is used when you play the remaining edge of an object. Usually, it deals with extreme heights or lows. Tall buildings are one example. In the case of looking up at a tall building (worm's eye view) the edges of the building will not only recede to the two vanishing points, but there will be an upward or download recession to a vanishing point. If looking down at an object in three points perspective it is referred to as a bird's eye view. There are three vanishing points in this case since the three sets of sides are receding in three different directions. Figure 4.1.7 illustrates a three points perspective image in social life.

Furthermore, high-resolution images are preferable. Image with dimension at least 800 x 600 are highly recommended. It is because it can improve the final result a little bit when you clicking some control points in the image. Also, you should be sure to choose images that accurately model perspective projection without any fisheye distortions. If the image is fisheye distortions, then some radial distortion correction should be applied on the image, which does not cover in this paper.

In this project, we choose figure 4.18 as the input source image. 3d model are built from this image. We will demonstrate the working steps in the following sections.

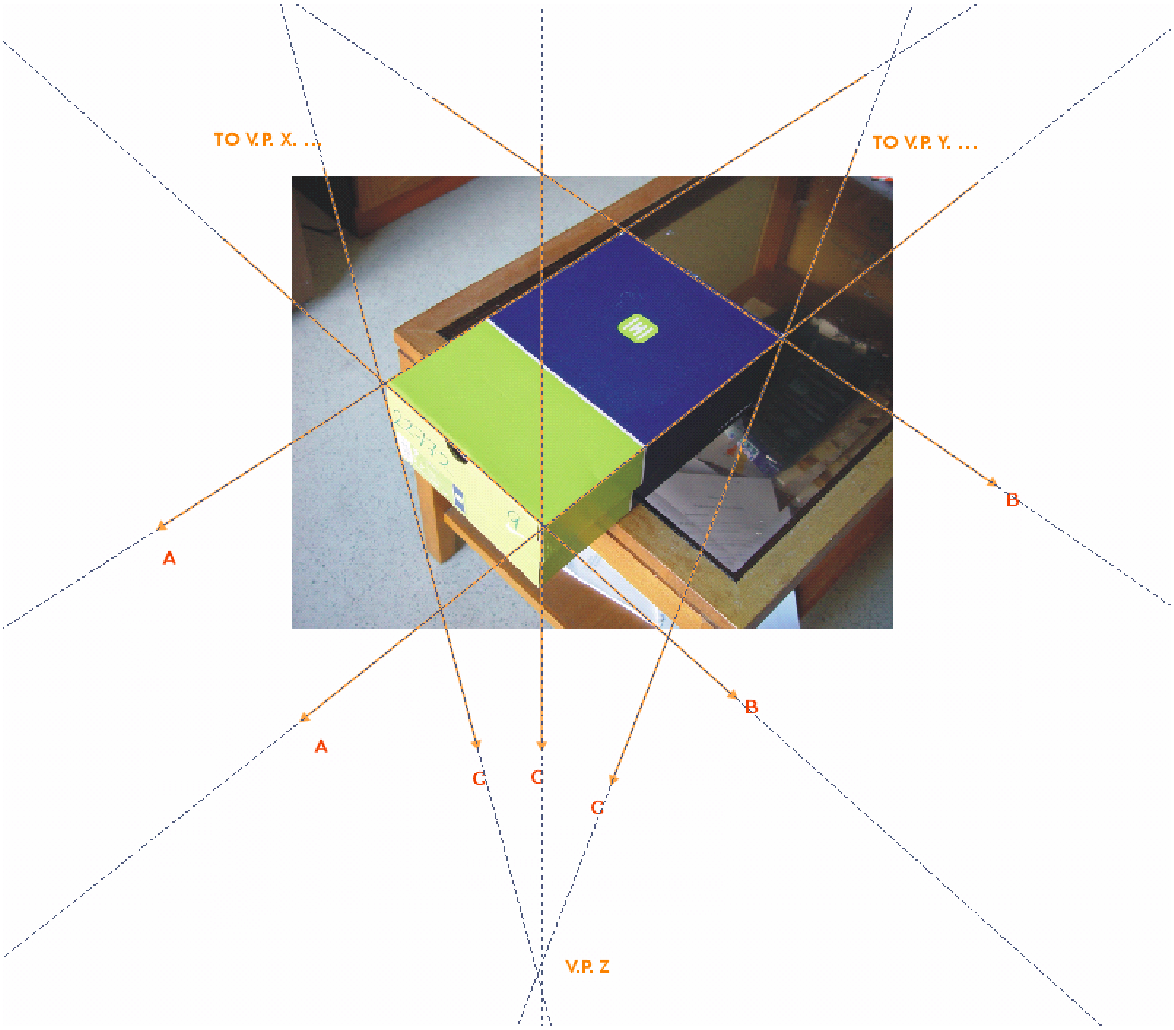


Figure 4.1.7: **Three Points Perspective Image.**

In the image, you can notice that there are three vanishing points. All the lines A in 3-space are parallel to each other to form the vanishing point in Y direction. All the lines B in 3-space are parallel to each other to form the vanishing point in X direction. All the lines C in 3-space are parallel to each other to form the vanishing point in Z direction.

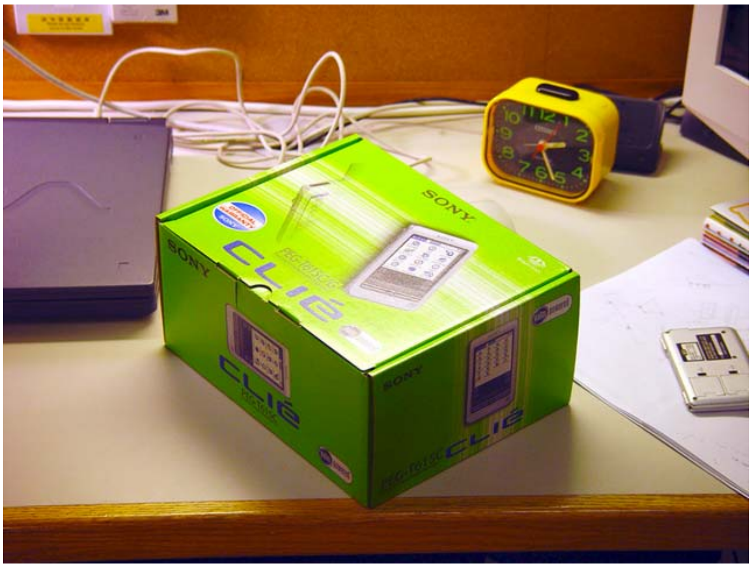
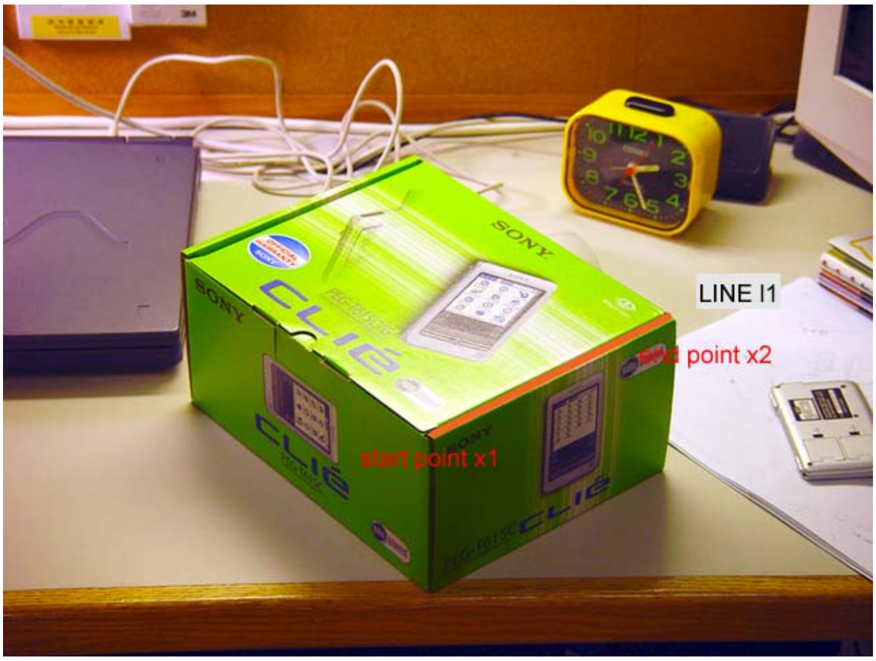


Figure 4.1.8: **Sample Input Source Image.**
We chose this image to do the demonstration in this paper. This input image has three vanishing points. We compute the vanishing line l from vanishing point x and y and leave the vanishing z as the reference direction.

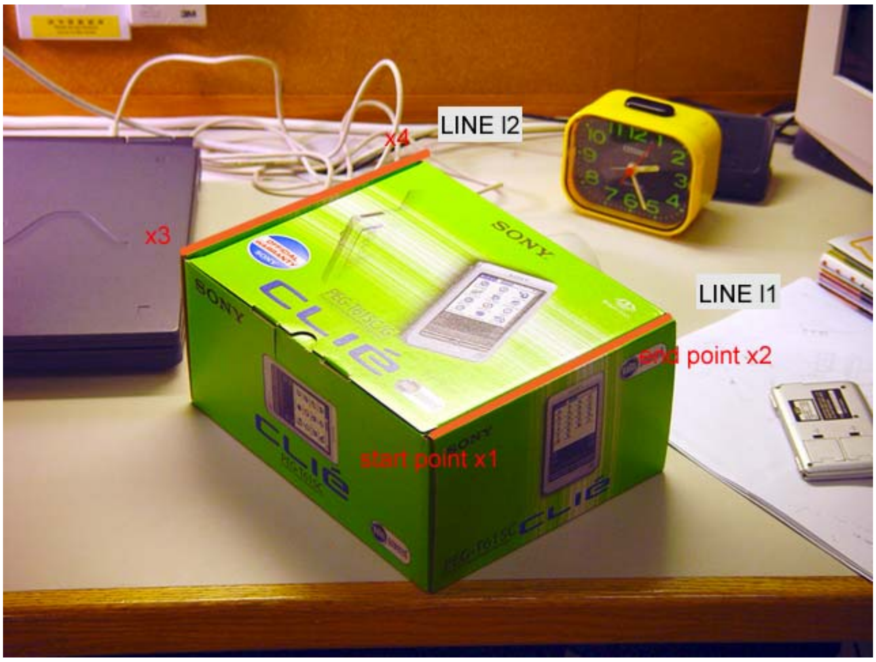
4.2. Calculating Vanishing Points

After choosing appropriate image, we can compute the vanishing points easily by defining four points in each direction. For example, in order to calculate the vanishing point in x direction, we need to draw two parallel lines, for instance, line l_1 and line l_2 that is parallel to x direction in 3-space. To draw line l_1 in the image, we need to specify two end points, which are point A (x_0, y_0) and point B (x_1, y_1) respectively. Then we could obtain the equation of line l_1 by a cross product, **A** x **B** = (x_0, y_0) x (x_1, y_1). We could apply the same technique to get the equation of line l_2. After we get the equation of line l_1 and line l_2, we are ready to find out the vanishing point in **x** direction. The steps for defining those lines are shown below.

STEP 1. In **x**-direction, define a start point and end point to form a line $\mathbf{l_1}$, $\mathbf{l_1} = \mathbf{x_1} \times \mathbf{x_2}$.

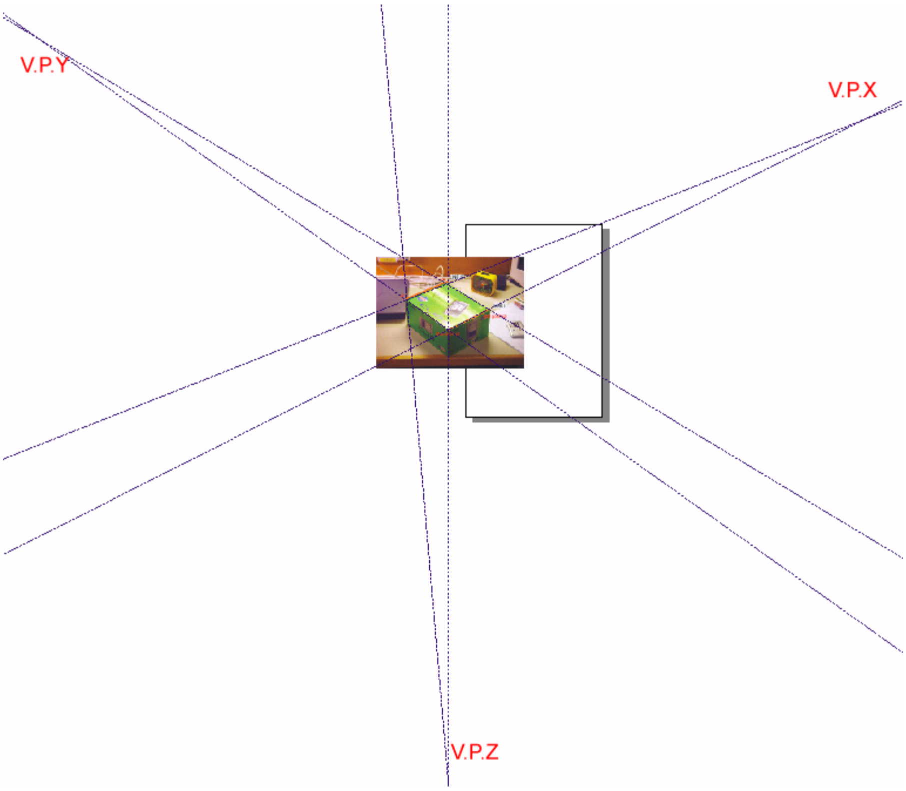


STEP 2. Follow the same technique in STEP 1 we can define line $\mathbf{l}_2$, $\mathbf{l}_2 = \mathbf{x}_3 \times \mathbf{x}_4$.



STEP 3. Make a cross product on $\mathbf{l}_1$ and $\mathbf{l}_2$ we get the vanishing point V.P.X.
$\mathbf{v.px} = \mathbf{l}_1 \times \mathbf{l}_2$.

STEP 4. Repeat STEP 1, STEP 2 and STEP 3 to compute the vanishing point V.P.Y and V.P.Z



In the above, we use a very simple technique to extract the vanishing point in X, Y and Z direction by finding an intersection point in each direction. This method is simple but the error rate is normally high. It is because if one of the parallel lines has error, it could be propagated to the resulting vanishing points. Therefore, in order to get an accurate estimation of the vanishing points, we have made a test by applying Bob Collins method. The steps is shown below:

Bob Collin's method

STEP 1. Specify each line's endpoints $\mathbf{e}_1$ and $\mathbf{e}_2$ in homogenous coordinates

$$\mathbf{e}_1 = (x_{1i} \quad y_{1i} \quad w)$$

$$\mathbf{e}_2 = (x_{2i} \quad y_{2i} \quad w)$$

Where the constant w is often taken as 1.

STEP 2. Compute a homogenous coordinate vector representing the line as the cross product of its two endpoints

$$(a_i \quad b_i \quad c_i) = \mathbf{e}_1 \times \mathbf{e}_2$$

Note that this resulting vector is just the parameters of the equation $a_i x + b_i y + c = 0$ of the 2d infinite line passing through the two endpoints

STEP 3. If you only have two lines, $\mathbf{l_1}$ and $\mathbf{l_2}$, you can compute a homogenous coordinate vector **v** representing their point of intersection as the cross product of these two line vectors

$$\mathbf{v} = \mathbf{v_1} \times \mathbf{v_2}$$

Scaling **v** so that the last coordinate is 1, i.e. $(\mathbf{v_x} \quad \mathbf{v_y} \quad 1)$, and you have $\mathbf{v_x}$ and $\mathbf{v_y}$ as the point in the image that is the vanishing point. It is better to leave the vanishing point as a homogenous coordinate vector.

STEP 4. If you have n lines $l_1, l_2, \cdots\cdots l_n$, you can get the "best fit" vanishing point as follows:

a) Form the 3x3 "second moment" matrix M as

$$\mathbf{M} = \sum_{i=1}^{n} \begin{pmatrix} a_i \cdot a_i & a_i \cdot b_i & a_i \cdot c_i \\ a_i \cdot b_i & b_i \cdot b_i & b_i \cdot a_i \\ a_i \cdot c_i & b_i \cdot c_i & c_i \cdot c_i \end{pmatrix}$$

Where the sum is taken for i = 1 to n. Note that M is a symmetric matrix.

b) Perform an eigendecomposition of **M**, using the Jacobi method, form numerical recipes in C, for example.

c) The eigenvector associated with the smallest eigenvalue is the vanishing point vector **V**

4.3. Calculating Vanishing Line

In the previous section, we define vanishing points by finding set of parallel lines from the image. After we get the vanishing points in X, Y, and Z direction, we can pick two of them to make a cross product to define a vanishing line. Different combination of vanishing points gets different vanishing lines. Once we get the vanishing line, reference plane on the world space is also defined too. For example, the vanishing line of the X-Y plane could be found by the equation $\mathbf{l_{xy}} = \mathbf{vp_x} \times \mathbf{vp_y}$. Figure 4.3.1 illustrates this example.

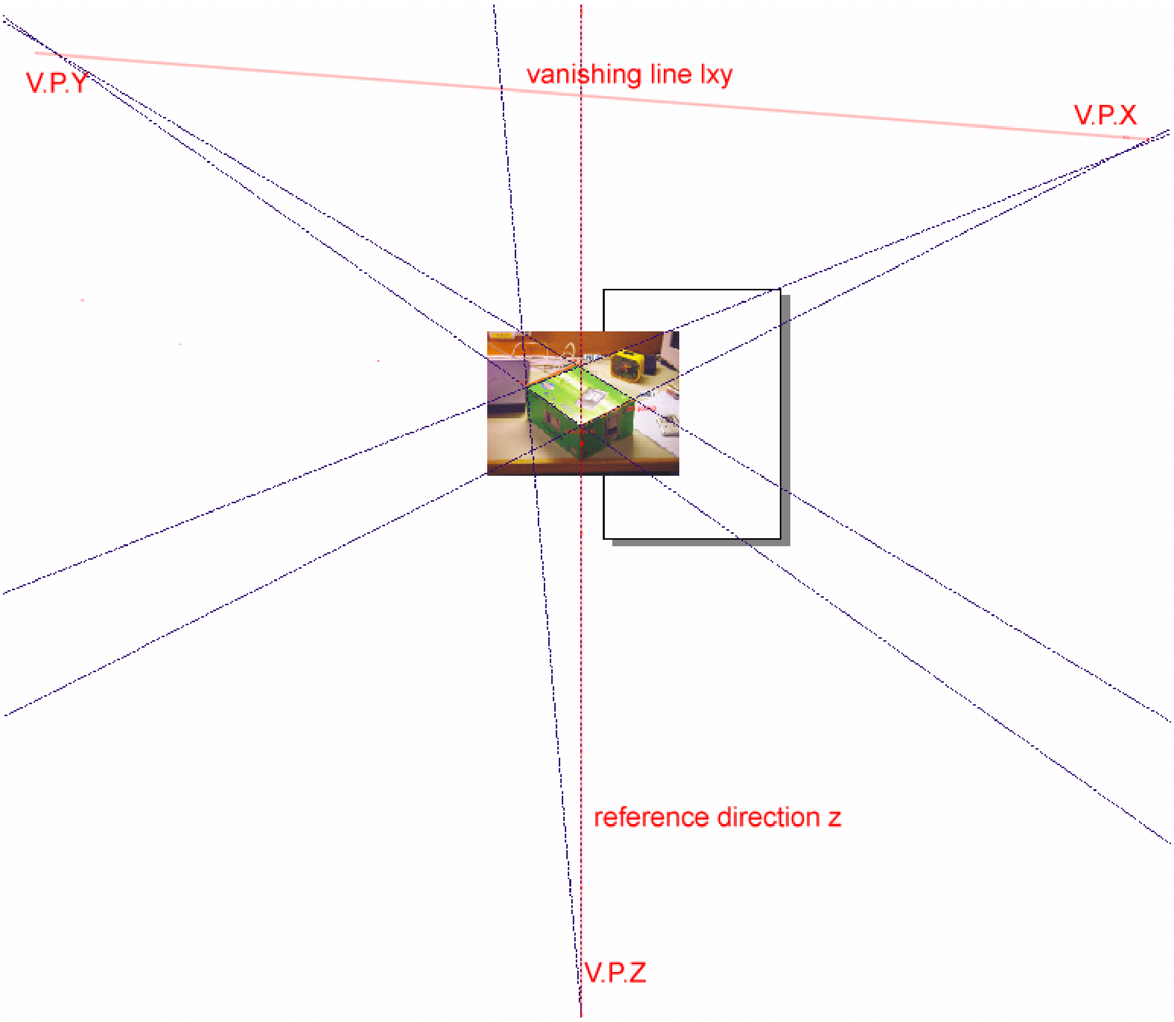


Figure 4.3.1: **Vanishing Line and Reference Direction.**

Vanishing line is computed by the cross product of vanishing point x and vanishing point y. Then, the remaining vanishing point z leaves as a reference direction. In this paper, we assume the direction of x, y and z are orthogonal in 3-space.

4.4. Define 3D scene origin

If reference planes are affined calibrated (we know its vanishing line and its reference direction) then from image measurements, we can compute the ratio of lengths of parallel line segments on the image planes. To extract the 3D coordinate of a particular image point, we can define more than one reference plane (typically is 3 which are along the canonical axes). But for simplicity, we could just define the origin position of the world coordinate frame on the image instead.

Before computing the 3D position of a particular point on the image plane, we need to fill in some unknown parameters. They are the imaged origin **o** of 3-space on the image plane, and the scaling factor $\alpha_x, \alpha_y, \alpha_z$.

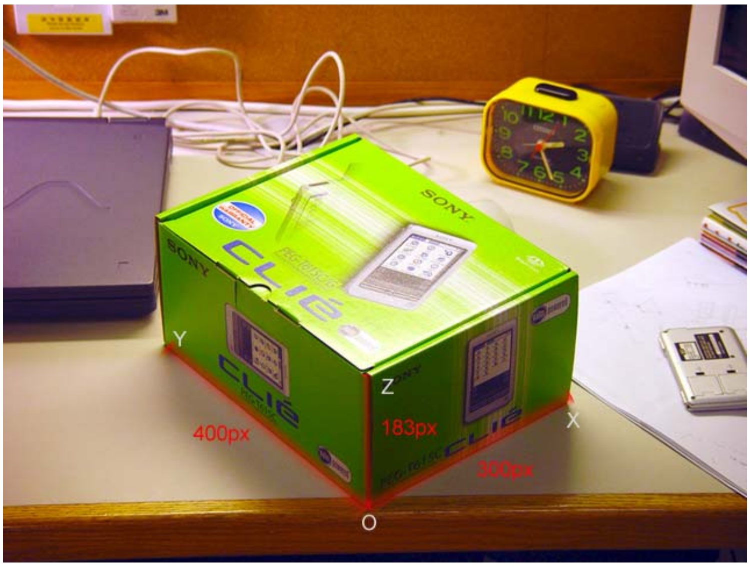


Figure 4.4.1: **Origin and Scaling Factor.**
In the figure, we define the origin at the lower left corner of the box. The direction X has magnitude 300 pixels. The direction Y has 400 pixels and the direction Z has magnitude 183 pixels.

In figure 4.41, we show an example on how to define the imaged origin of 3-space on the image plane. We pick the lower corner of the box as origin, named O. Also, the scaling factor $\alpha_x, \alpha_y, \alpha_z$ can be computed from the real measurement. The X magnitude is 300 pixels. The Y magnitude is 400 pixels. The Z magnitude is 183 pixels.

4.5. Compute 3d Positions

In previous section, we not only define the imaged origin, but also compute the scaling factor $\alpha_x, \alpha_y, \alpha_z$. Then we can compute the 3D points with the corresponding 2D points on the image plane.

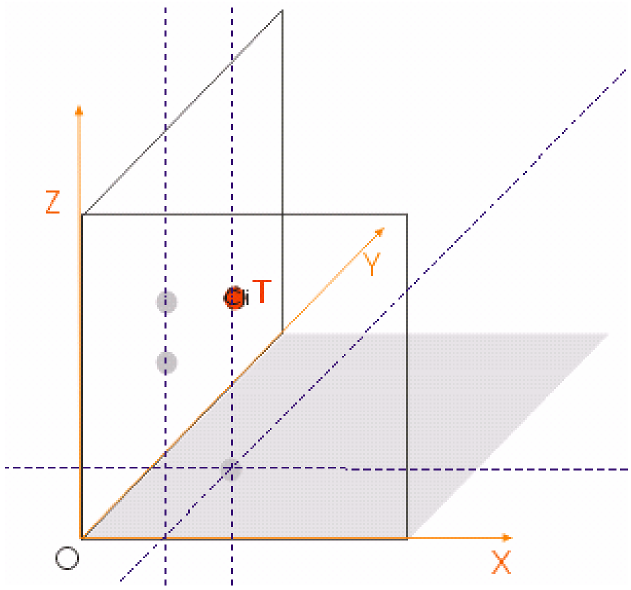


Figure 4.5.1: **Compute affine 3d reconstruction from single images.**
The 3d space is parametrized as three pencils of parallel planes. The location of a world point T can be computed in the following formula.

In 3-space, the general X, Y, Z location of a 3D point $\mathbf{T} = \lfloor \alpha_x X \quad \alpha_y Y \quad \alpha_z Z \rfloor$ may be computed from single images apply the following formula:

$$\alpha_x X = -\frac{\mathbf{o} \cdot \mathbf{l}_{yz} \| \mathbf{b}_{yz} \times \mathbf{t} \|}{\mathbf{b}_{yz} \cdot \mathbf{l}_{yz} \| \mathbf{v}_x \times \mathbf{t} \|}, \quad \alpha_y Y = -\frac{\mathbf{o} \cdot \mathbf{l}_{xz} \| \mathbf{b}_{xz} \times \mathbf{t} \|}{\mathbf{b}_{xz} \cdot \mathbf{l}_{xz} \| \mathbf{v}_y \times \mathbf{t} \|}, \quad \alpha_z Z = -\frac{\mathbf{o} \cdot \mathbf{l}_{xy} \| \mathbf{b}_{xy} \times \mathbf{t} \|}{\mathbf{b}_{xy} \cdot \mathbf{l}_{xy} \| \mathbf{v}_z \times \mathbf{t} \|}$$

Where **t** is the image of the world point **T**, $\mathbf{l_{ij}} = \mathbf{v_i} \times \mathbf{v_j}$ and $\mathbf{t_{ij}}$ is the intersection of the plane spanned by the axis i and j with the line through **T** parallel to the direction k. In previous section, we have shown that how to calculate the α_i value. If the α_i parameters are known then metric structure can be computed for the point T; otherwise only affine structure can be obtained.

- Compute the Z coordinate of T in 3-space (Figure 4.5.2)

  Base on the formula $\alpha_z Z = -\frac{\mathbf{o} \cdot \mathbf{l}_{xy} \| \mathbf{b}_{xy} \times \mathbf{t} \|}{\mathbf{b}_{xy} \cdot \mathbf{l}_{xy} \| \mathbf{v}_z \times \mathbf{t} \|}$, we set that $\mathbf{t} = \mathbf{t_z}$, $\mathbf{b_{xy}} = \mathbf{b_z}$. Also, we know the value of the scaling factor α_z. Finally, we get the value $Z = \frac{\mathbf{o} \cdot \mathbf{l}_{xy} \| \mathbf{b}_z \times \mathbf{t}_z \|}{\mathbf{b}_z \cdot \mathbf{l}_{xy} \| \mathbf{v}_z \times \mathbf{t}_z \|} \frac{1}{\alpha_z}$, which is the Z coordinate of T in 3-space.

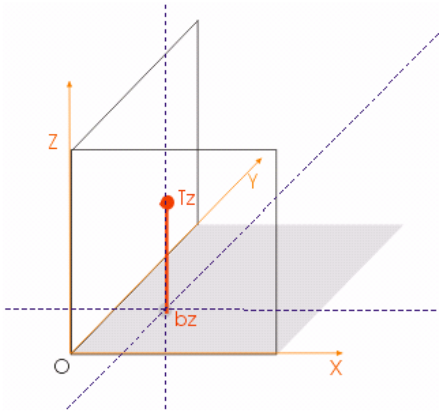


Figure 4.5.2: **Compute Tz coordinate.**

- Compute the X coordinate in 3-space (Figure 4.5.3)

  Base on the formula $\alpha_x X = \frac{\mathbf{o}\cdot\mathbf{l}_{yz}\|\mathbf{b}_{yz}\times\mathbf{t}\|}{\mathbf{b}_{yz}\cdot\mathbf{l}_{yz}\|\mathbf{v}_x\times\mathbf{t}\|}$, we set that $\mathbf{t} = \mathbf{t}_x$, $\mathbf{b}_{yz} = \mathbf{b}_x$. Also, we know the value of the scaling factor α_x. Finally, we get the value $X = \frac{\mathbf{o}\cdot\mathbf{l}_{yz}\|\mathbf{b}_x\times\mathbf{t}_x\|}{\mathbf{b}_x\cdot\mathbf{l}_{yz}\|\mathbf{v}_x\times\mathbf{t}_x\|}\frac{1}{\alpha_x}$, which is the X coordinate of T in 3-space.

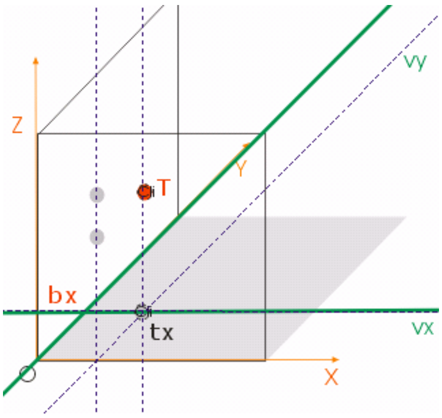


Figure 4.5.3: **Compute Tx coordinate.**

- Compute the Y coordinate in 3-space (Figure 4.5.4)

  Base on the formula $\alpha_y Y = \frac{\mathbf{o}\cdot\mathbf{l}_{xz}\|\mathbf{b}_{xz}\times\mathbf{t}\|}{\mathbf{b}_{xz}\cdot\mathbf{l}_{xz}\|\mathbf{v}_y\times\mathbf{t}\|}$, we set that $\mathbf{t} = \mathbf{t}_y$, $\mathbf{b}_{yz} = \mathbf{b}_y$. Also, we know the value of the scaling factor α_y. Finally, we get the value $Y = \frac{\mathbf{o}\cdot\mathbf{l}_{xz}\|\mathbf{b}_y\times\mathbf{t}_y\|}{\mathbf{b}_y\cdot\mathbf{l}_{xz}\|\mathbf{v}_y\times\mathbf{t}_y\|}\frac{1}{\alpha_y}$, which is the Y coordinate of T in 3-space.

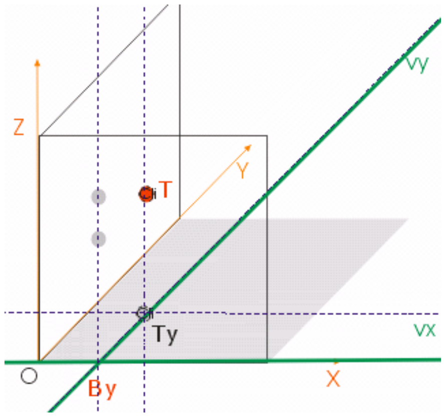


Figure 4.5.4: **Compute Ty coordinate.**

4.6. Compute Texture Maps

After we get the 3D position of the model, we can extract the texture by image wrapping or perspective transformation.

4.6.1 Perspective Transformation

The general representation of a perspective transformation is

$$[x', y', w'] = [u, v, w]\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \tag{4.6.1}$$

where x = x' / w' and y = y' / w'.

A perspective transformation, or projective mapping, is produced when $[a_{13} \quad a_{23}]^T$ is nonzero. It is used in conjunction with a projection onto a viewing plane in what is known as a perspective or central projection. Perspective transformations preserve parallel lines only when they are parallel to the projection plane. Otherwise, lines converge to a vanishing point. This has the property of foreshortening distinct lines, a useful technique for rendering realistic images. For perspective transformations, the forward mapping functions are

$$x = \frac{x'}{w'} = \frac{a_{11}u + a_{21}v + a_{31}}{a_{13}u + a_{23}v + a_{33}} \qquad (4.6.2a)$$

$$y = \frac{y'}{w'} = \frac{a_{12}u + a_{22}v + a_{32}}{a_{13}u + a_{23}v + a_{33}} \qquad (4.6.2b)$$

They take advantage of the fact that w' allowed to vary at each point and division by w' is equivalent to a project using rays passing through the origin. Note that affine transformations are a special case of perspective transformations where w' is constant over the entire image, i.e. $a_{13} = a_{23} = 0$.

Perspective transformations share several important properties with affine transformations. They are planar mappings, and thus their forward and inverse transforms are single-valued. They preserve lines in all orientations. That is, lines map onto lines (although not of the same orientation). As we shall see, this desirable property is lacking in more general mappings. Furthermore, the nine degrees of freedom in Eq. 4.6.1 is sufficient to permit planar quadrilateral-to-quadrilateral mappings. In contrast, affine transformations offer only six degree of freedom and thereby facilitate only triangle-to-triangle mappings.

4.6.2 Inferring Perspective Transformations

A perspective transformation is expressed in terms of the nine coefficients in the general 3x3 matrix T_1. Without loss of generality, T_1 can be normalized so that $a_{33} = 1$. This leaves eight degrees of freedom for a projective mapping. The eight coefficients can be determined by establishing correspondence between four points in the reference and observed images. Let (u_k, v_k) and (x_k, y_k) for k = 0,1,2,3 be these four points in the reference and observed images, respectively. Assuming $a_{33} = 1$, Eqs. (4.6.2a) and (4.6.2b) can be rewritten as

$$x = a_{11}u + a_{21}v + a_{31} - a_{13}ux - a_{23}vx \qquad (4.6.3a)$$

$$y = a_{12}u + a_{22}v + a_{32} - a_{13}uy - a_{23}vy \qquad (4.6.3b)$$

Applying Eqs. (4.6.3a) and (4.6.3b) to the four pairs of correspondence points yields the 8x8 system of equations shown in Eq. (4.6.4).

$$\begin{pmatrix} u_0 & v_0 & 1 & 0 & 0 & 0 & -u_0x_0 & -v_0x_0 \\ u_1 & v_1 & 1 & 0 & 0 & 0 & -u_1x_1 & -v_1x_1 \\ u_2 & v_2 & 1 & 0 & 0 & 0 & -u_2x_2 & -v_2x_2 \\ u_3 & v_3 & 1 & 0 & 0 & 0 & -u_3x_3 & -v_3x_3 \\ 0 & 0 & 0 & u_0 & v_0 & 1 & -u_0y_0 & -v_0y_0 \\ 0 & 0 & 0 & u_1 & v_1 & 1 & -u_1y_1 & -v_1y_1 \\ 0 & 0 & 0 & u_2 & v_2 & 1 & -u_2y_2 & -v_2y_2 \\ 0 & 0 & 0 & u_3 & v_3 & 1 & -u_3y_3 & -v_3y_3 \end{pmatrix} \mathbf{A} = \mathbf{X} \tag{4.6.4}$$

The coefficients are determined by solving the linear system. This yields a solution to the general (planar) quadrilateral-to-quadrilateral problem. Speedups are possible when considering several special cases: square-to-quadrilateral, quadrilateral-to-square, and quadrilateral-to-quadrilateral using the results of the last two cases. We now consider each case individually. A detailed exposition is found in [Heckbert 89].

4.6.2.1 Case 1: Square-to-Quadrilateral

Consider mapping a unit square onto an arbitrary quadrilateral. The following four-point correspondences are established from the uv-plane onto the xp-plane.

$$\begin{aligned} (0,0) &\rightarrow (x_0, y_0) \\ (1,0) &\rightarrow (x_1, y_1) \\ (0,1) &\rightarrow (x_2, y_2) \\ (1,1) &\rightarrow (x_3, y_3) \end{aligned}$$

In this case, the eight equations become

$$
\begin{aligned}
a_{31} &= x_0 \\
a_{11} + a_{31} - a_{13}x_1 &= x_1 \\
a_{11} + a_{21} + a_{31} - a_{13}x_2 + a_{23}x_2 &= x_2 \\
a_{12} + a_{13} - a_{23}x_3 &= x_3 \\
a_{32} &= y_0 \\
a_{12} + a_{32} - a_{13}y_1 &= y_1 \\
a_{12} + a_{22} + a_{32} - a_{13}y_2 + a_{23}y_2 &= y_2 \\
a_{22} + a_{32} - a_{23}y_3 &= y_3
\end{aligned}
$$

The solution can take two forms, depending on whether the mapping is affine or perspective. We define the following terms for our discussion.

$$
\begin{aligned}
\Delta x_1 &= x_1 - x_2 \\
\Delta x_2 &= x_3 - x_2 \\
\Delta x_3 &= x_0 - x_1 + x_2 - x_3 \\
\Delta y_1 &= y_1 - y_2 \\
\Delta y_2 &= y_3 - y_2 \\
\Delta y_3 &= y_0 - y_1 + y_2 - y_3
\end{aligned}
$$

4.6.2.2 Case 2: Quadrilateral-to-Square

This case is the inverse of the mapping already considered. As discussed earlier, the adjoint of a projective mapping can be used in place of the inverse. Thus, the simplest solution is to compute the square-to-quadrilateral mapping coefficients described above to find the inverse of the desired mapping, and then take its adjoint to compute the quadrilateral-to-square mapping.

4.6.2.3 Case 3: Quadrilateral-to-Quadrilateral

The results of the last two cases may be cascaded to yield a fast solution to the general quadrilateral-to-quadrilateral mapping problem.

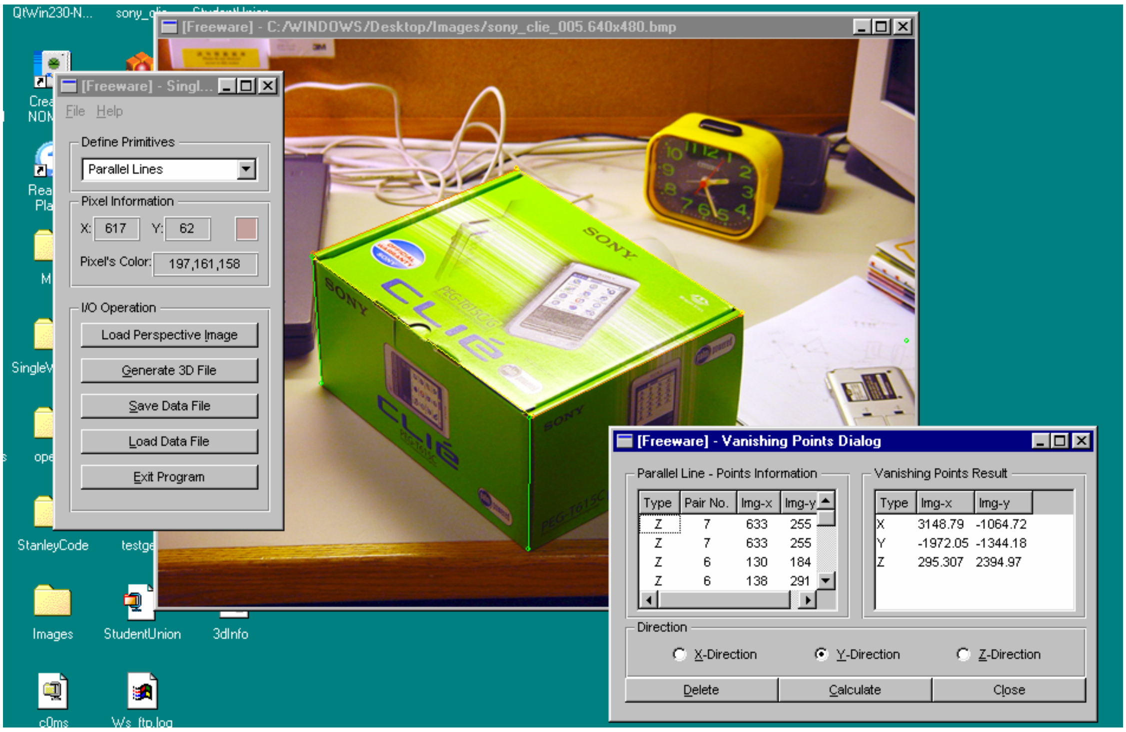
The general quadrilateral-to-quadrilateral problem is also known as four-corner mapping. Perspective transformations offer a planar solution to this problem. When the quadrilaterals become non-planar, however, more general solutions are necessary.

Bilinear solution transformations are an example of the simplest mapping functions that address four-corner mappings for nonplanar quadrilaterals.

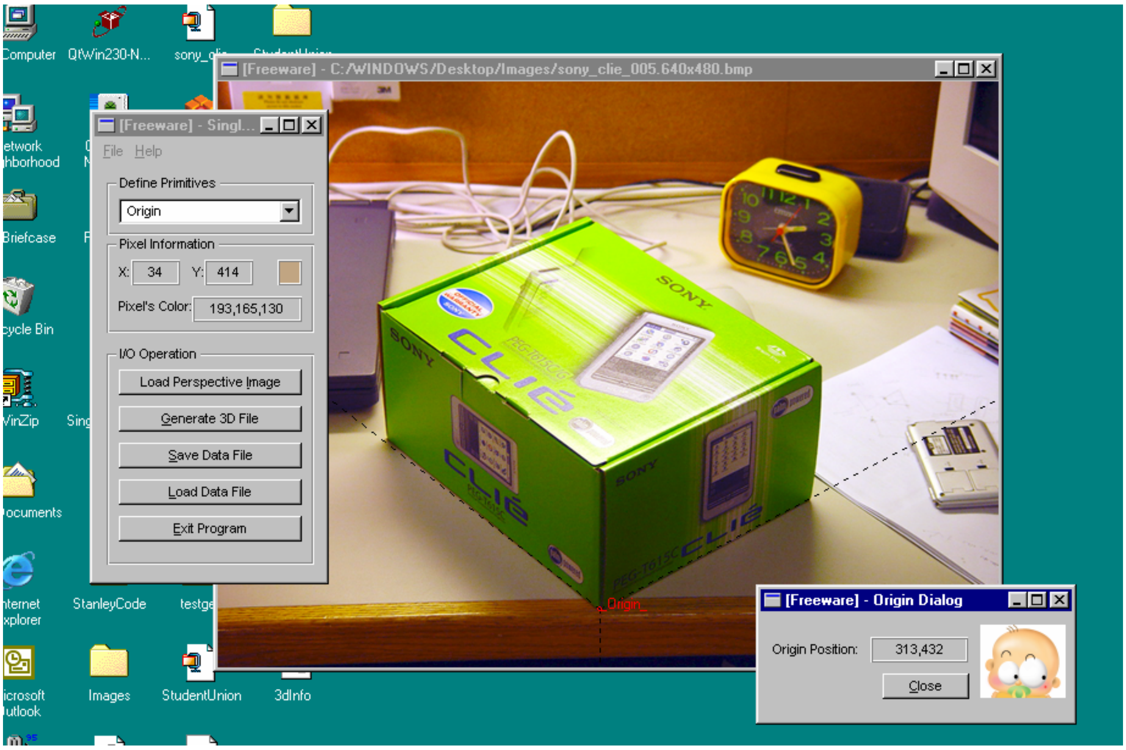
4.7 Create a VRML Model

After getting the 3D model vertices and their textures, we can build a VRML model to display the output. VRML is one of the well-known languages to present 3D model in either client machine or even on the Internet through the web browser with appropriate plug-in installed.

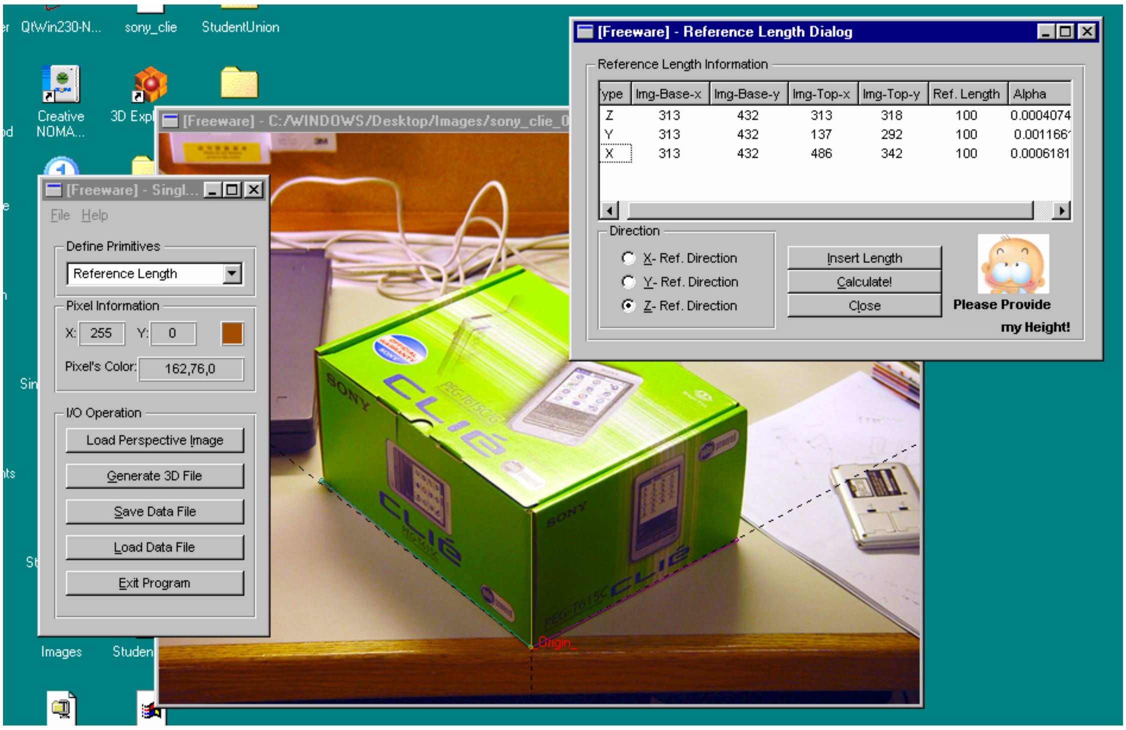
IV. Demonstration



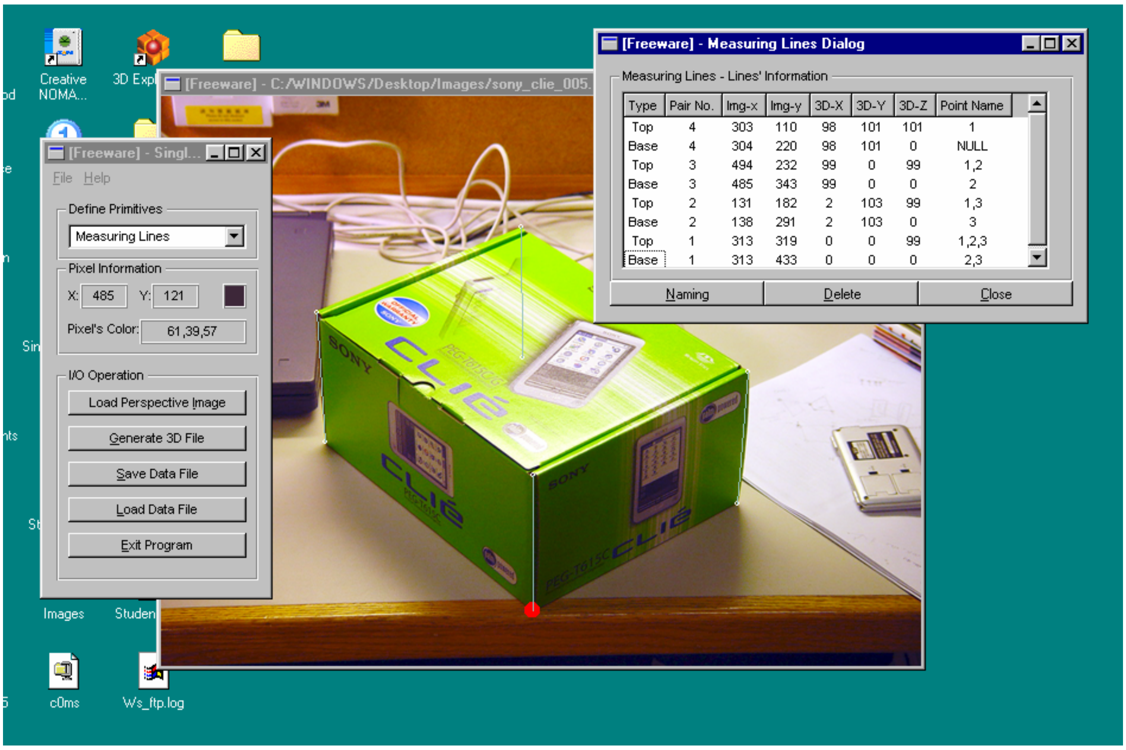
Step1: The user needs to load an image which is high resolution, accurately model perspective projection without fisheye distortions. Actually it should be complex enough to create an interesting model with at least ten textured polygons, yet not so complex that the resulting mode is hard to digitize or approximate.Then, a scene coordinate frame is constructed by defining lines in the scene that are parallel to the X, Y and Z axis. For each axis, digitize more than two lines parallel to that axis. The intersections of these lines in the images define the corresponding vanishing point. Since the accuracy of the model depends on the precision of the vanishing points, implement a robust technique for computing vanishing points that uses more than two lines is needed. So the method that proposed by Collins is used in the project.



Step2: To extract the 3D coordinate of a particular image point, we can define more than one reference plane (typically is 3 which are along the canonical axes). But for simplicity, we could just define the origin position of the world coordinate frame on the image instead.

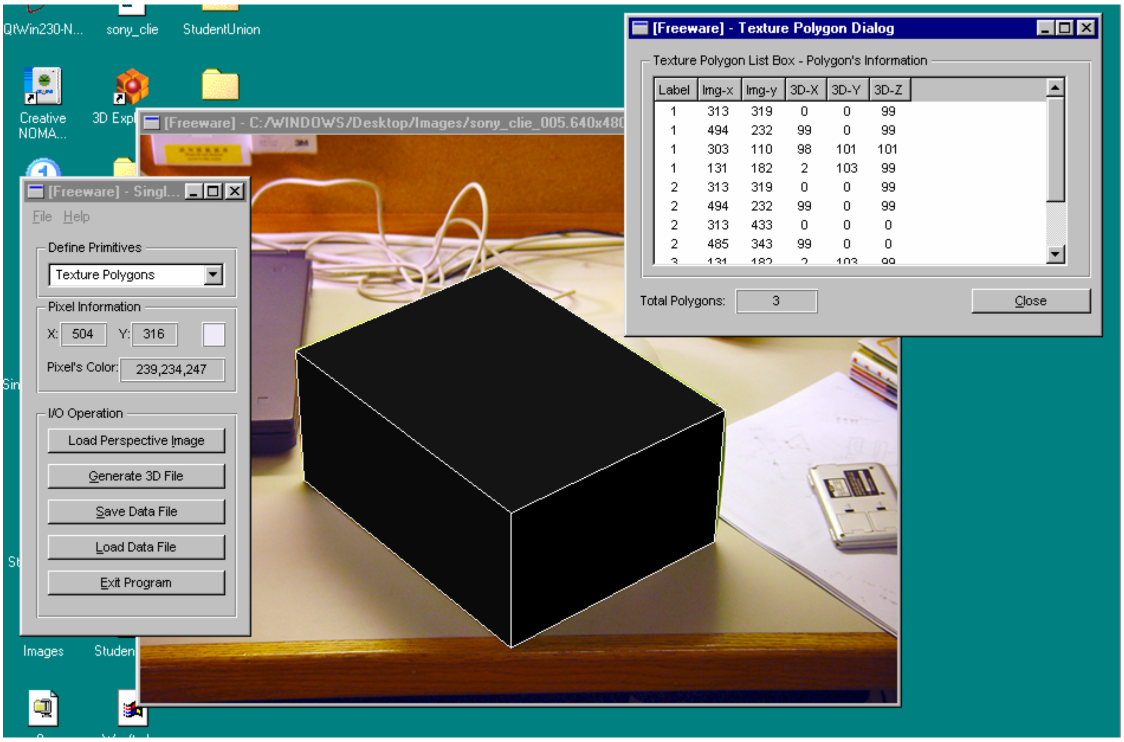


Step3: To avoid affine distortions in the model, user will need to set the scale parameters as described in the paper. One way of doing this is to measure, in 3-D, when user shoot the picture, the positions of 4 points on the reference plane and one point off of that plane. The 4 reference plane points and their image projections define a 3x3 matrix **H** that maps u-v points to X-Y positions on the plane. The fifth point determines the scale factor alpha off of the plane, as described in the paper. Alternatively, user can specify **H** and alpha without physical measurement by identifying a regular structure such as a cube and choosing its dimensions to be unit lengths. This latter approach is necessary for paintings and other scenes in which physical measurements are not feasible.

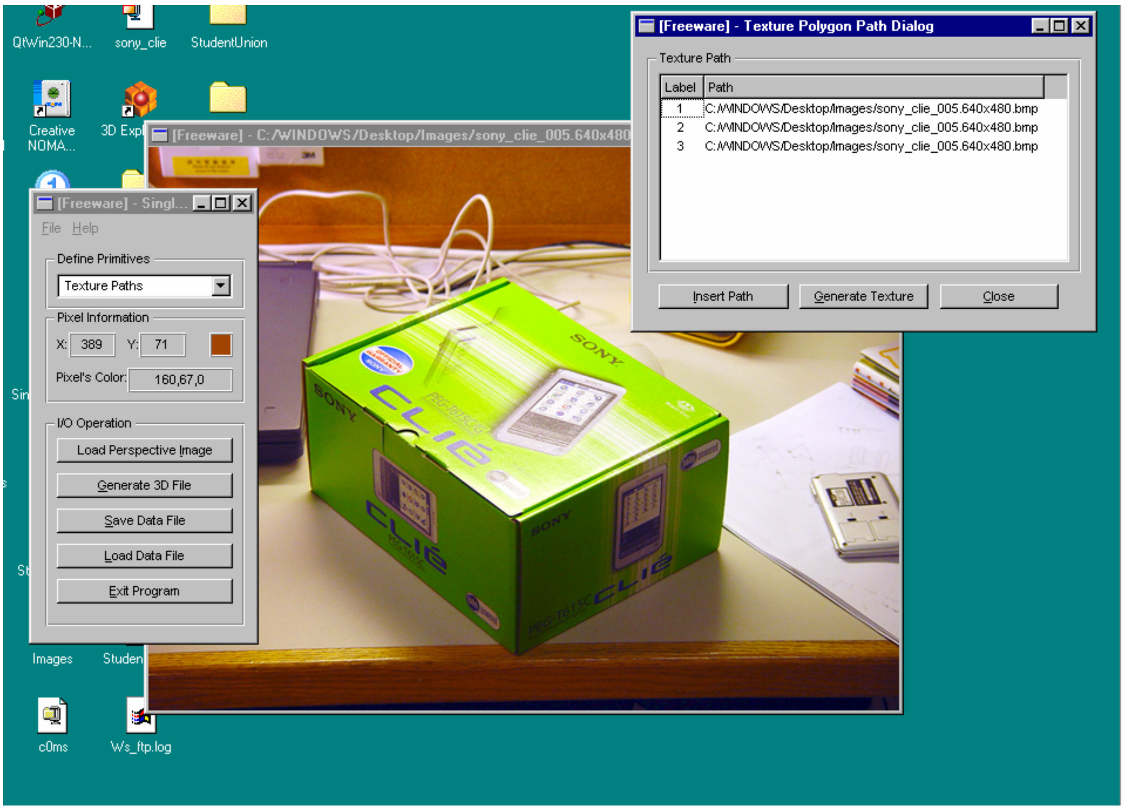


Step 4: The 3D space is parameterized as three pencils of parallel planes which are orthogonal to each other. The location of a world point Q can be computed by applying

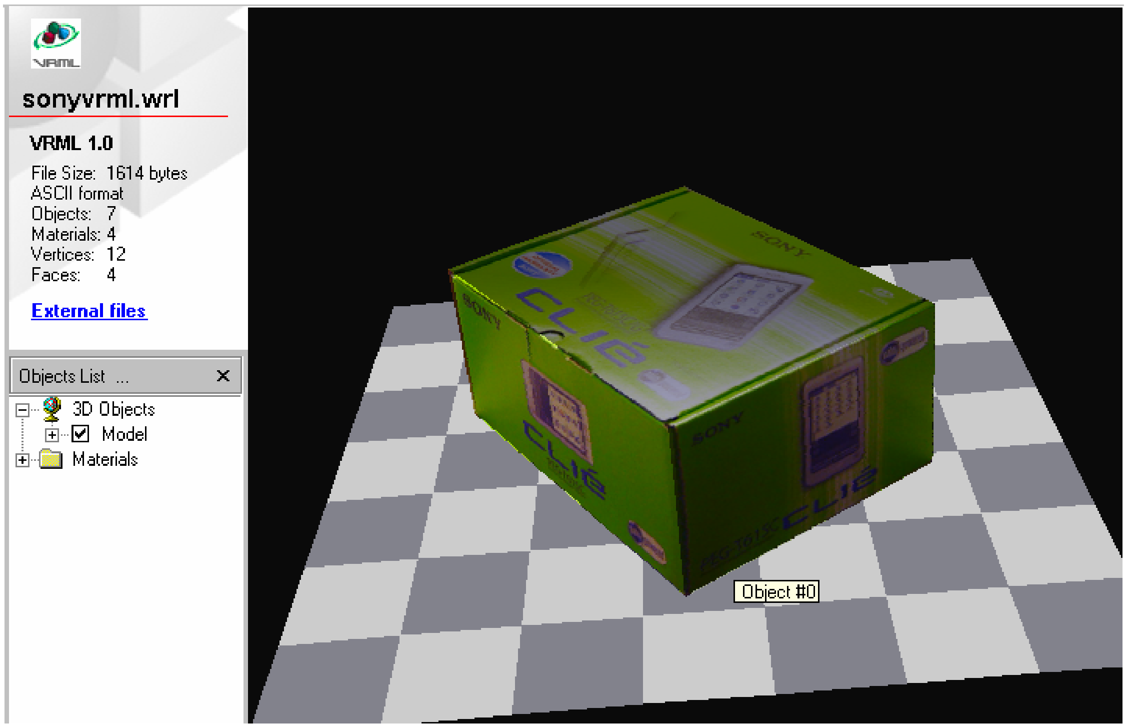
$$\alpha_x X = -\frac{\mathbf{o}\cdot\mathbf{l}_{yz}\|\mathbf{b}_{yz}\times\mathbf{t}\|}{\mathbf{b}_{yz}\cdot\mathbf{l}_{yz}\|\mathbf{v}_x\times\mathbf{t}\|},\ \alpha_y Y = -\frac{\mathbf{o}\cdot\mathbf{l}_{xz}\|\mathbf{b}_{xz}\times\mathbf{t}\|}{\mathbf{b}_{xz}\cdot\mathbf{l}_{xz}\|\mathbf{v}_y\times\mathbf{t}\|},\ \alpha_z Z = -\frac{\mathbf{o}\cdot\mathbf{l}_{xy}\|\mathbf{b}_{xy}\times\mathbf{t}\|}{\mathbf{b}_{xy}\cdot\mathbf{l}_{xy}\|\mathbf{v}_z\times\mathbf{t}\|}$$



Step5: Constructed patches could be observed in Texture Polygon View.



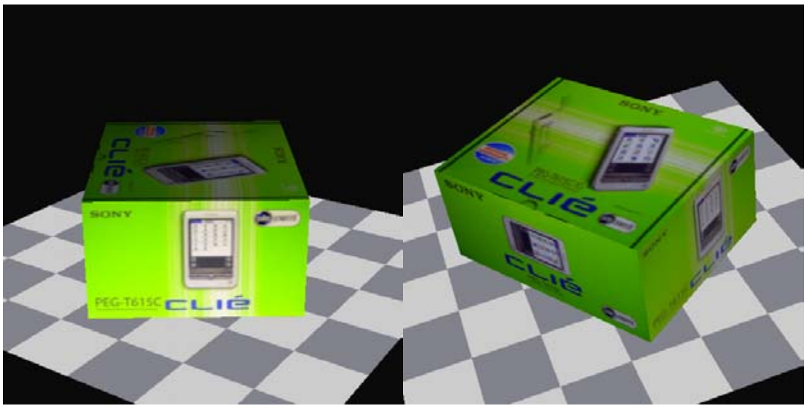
Step 6: The last step is to compute texture maps for each of these patches. If the patch is a rectangle in the scene, e.g., a wall or door, all which is needed, is to warp the quadrilateral image region into a rectangular texture image. It is best to choose the width and height of the texture image to be the about the same as that of the original quadrilateral, to avoid loss of resolution. If the patch is a non-rectangular region such as the outline of a person, user needs to perform the following steps: (1) edit the texture image and mark out "transparent" pixels by hand using image editing software (e.g. Adobe Photoshop). (2) select the processed image path that corresponding to the particular patch. (3) warp this into a rectangular texture image, as before.

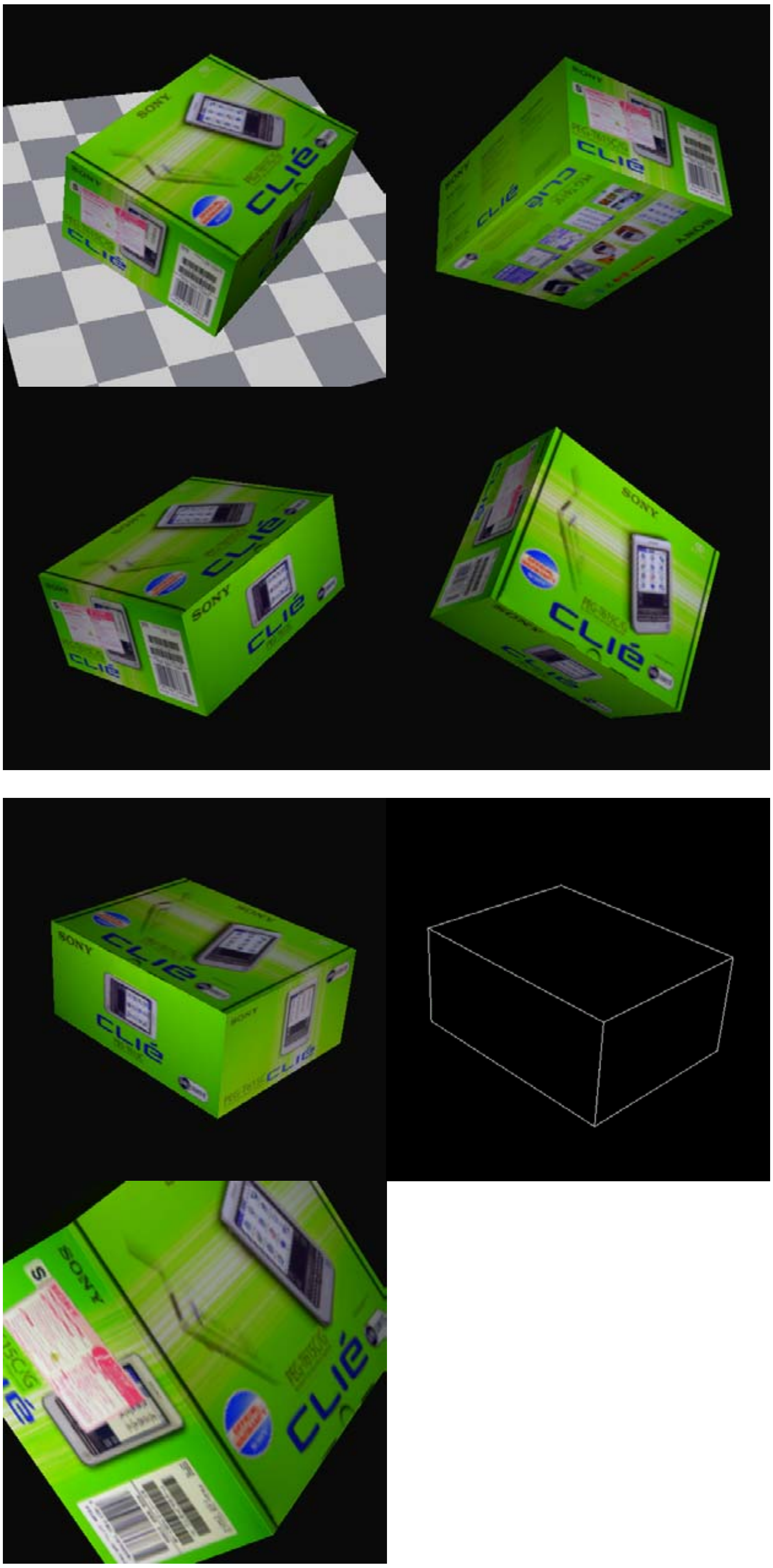


Step 7: A 3D VRML model is generated.

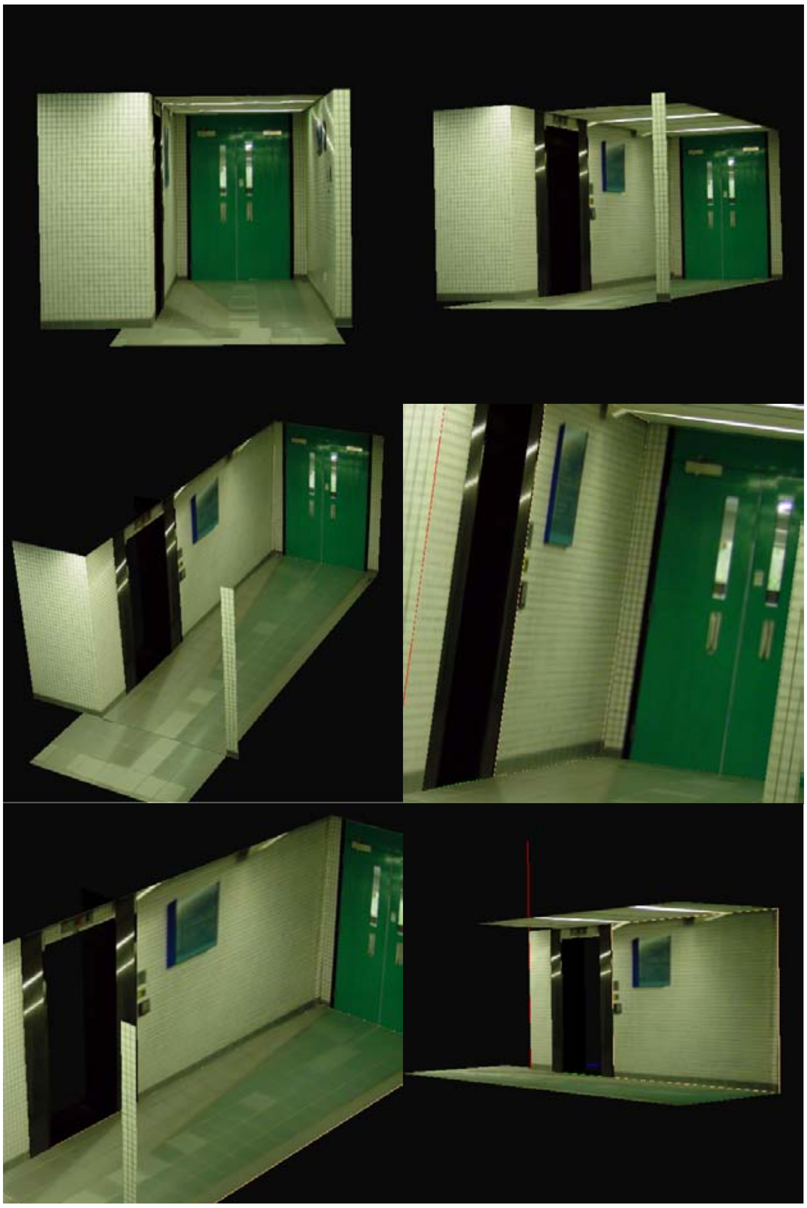
More Examples.

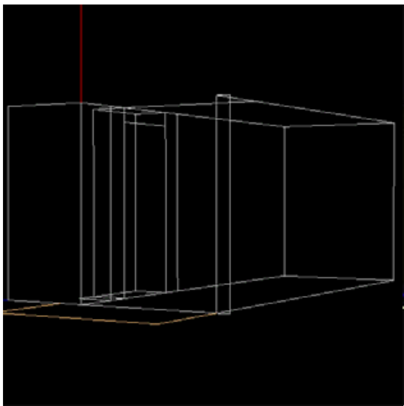
Original Image.

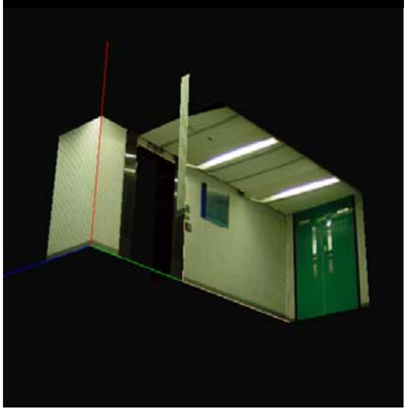




Original Image.







V. Technical Difficulties

Firstly, the original paper focus on the calculation of z component and doesn't say much about x and y component of the measurement calculation. But the solution could be found in the Antonio Criminisi's thesis, "Accurate Visual Metrology from Single and Multiple Uncalibrated Images" and other related papers.

Secondly, the result 3D re-construction by the original implementation of that paper is not satisfying because the origin is defined in a straightforward way. So we could improve the original equation from

$$\alpha_x X = \frac{\|\mathbf{b}_{yz} \times \mathbf{t}\|}{\mathbf{b}_{yz} \cdot \mathbf{l}_{yz} \|\mathbf{v}_x \times \mathbf{t}\|}, \ \alpha_y Y = \frac{\|\mathbf{b}_{xz} \times \mathbf{t}\|}{\mathbf{b}_{xz} \cdot \mathbf{l}_{xz} \|\mathbf{v}_y \times \mathbf{t}\|}, \alpha_z Z = \frac{\|\mathbf{b}_{xy} \times \mathbf{t}\|}{\mathbf{b}_{xz} \cdot \mathbf{l}_{xy} \|\mathbf{v}_z \times \mathbf{t}\|}.$$

to

$$\alpha_x X = -\frac{\mathbf{o} \cdot \mathbf{l}_{yz} \|\mathbf{b}_{yz} \times \mathbf{t}\|}{\mathbf{b}_{yz} \cdot \mathbf{l}_{yz} \|\mathbf{v}_x \times \mathbf{t}\|}, \ \alpha_y Y = -\frac{\mathbf{o} \cdot \mathbf{l}_{xz} \|\mathbf{b}_{xz} \times \mathbf{t}\|}{\mathbf{b}_{xz} \cdot \mathbf{l}_{xz} \|\mathbf{v}_y \times \mathbf{t}\|}, \ \alpha_z Z = -\frac{\mathbf{o} \cdot \mathbf{l}_{xy} \|\mathbf{b}_{xy} \times \mathbf{t}\|}{\mathbf{b}_{xy} \cdot \mathbf{l}_{xy} \|\mathbf{v}_z \times \mathbf{t}\|}$$

Finally, as the equations that mentioned in the paper mainly deal with the positive quadrant (which is positive), negative parts needed to be managed by adding more constraints. Specifically, we need to check the direction of the current calculating vector to see whether it is same as or opposite to the one that defined by the user. If the direction is the same, we will define it as positive, other negative value will be assigned.

VI. Summary and Conclusions

We have explored how the affine structure of three-dimensional space may be partially recovered from perspective images in terms of a set of planes parallel to reference plane and a reference direction not parallel to the reference plane.

For the further improvement, we can try to compute and detect the vanishing points and lines automatically. We think that it is not hard as there is some well-known papers are published on the topic “vanishing point detection and computation”.

VII. Reference

[Heckbert 89] Heckbert, Paul, Funderamentals of Texture Mapping and Image Wrapping, Masters Thesis, Dept. of EECS, U. of California at Berkeley, Technical Report No. UCB/CSD 89/516, June 1989.

A. Criminisi, I. Reid and A. Zisserman Department of Engineering Science University of Oxford, Single View Metrology. 1999

Dr Antonio Criminisi, PhD Thesis, University of Oxfordm, Accurate Visual Metrology from Single and Multiple Uncalibrated Images. 1999