1 The Pinhole Camera Model

1.1 General Introduction

A pinhole camera is a simple camera without a lens but with a tiny aperture, a pinhole – effectively a light-proof box with a small hole in one side. Light from a scene passes through the aperture and projects an inverted image on the opposite side of the box, which is known as the camera obscure effect.

A camera could approach a projective model, often called pinhole projection. The simplest representation of a camera is a photosensitive surface (sensor): a plane image, a lens (projective projection) and a certain position and an orientation in space.

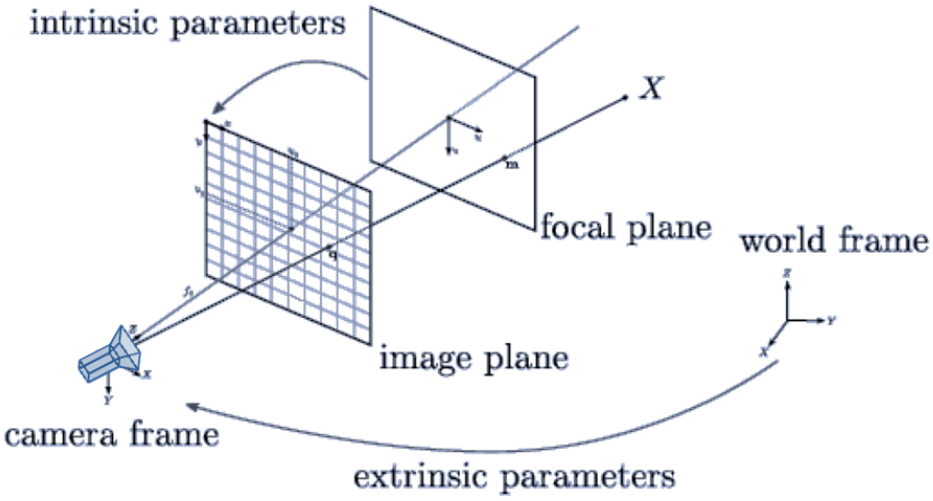


Figure 1 – The pinhole camera model

1.2 What is homography?

In the field of computer vision, any two images of the same planar surface in space are related by a homography (assuming a pinhole camera model). This has many practical applications, such as image rectification, image registration, or computation of camera motion—rotation and translation—between two images. Once camera rotation and translation have been extracted from an estimated homography matrix, this information may be used for navigation, or to insert 3D object model into an image or video, so that they are rendered with the correct perspective and appear to have been part of the original scene.

In the inverse projection, the homography $\mathbf{H}$ depends on the rotation matrix $\mathbf{R}$ and the translation vector $\mathbf{t}$ which correspond to a rigid transformation between the actual position of the camera providing the image and the virtual image in which other image is generated. The homography matrix can be calculated using the following equation.

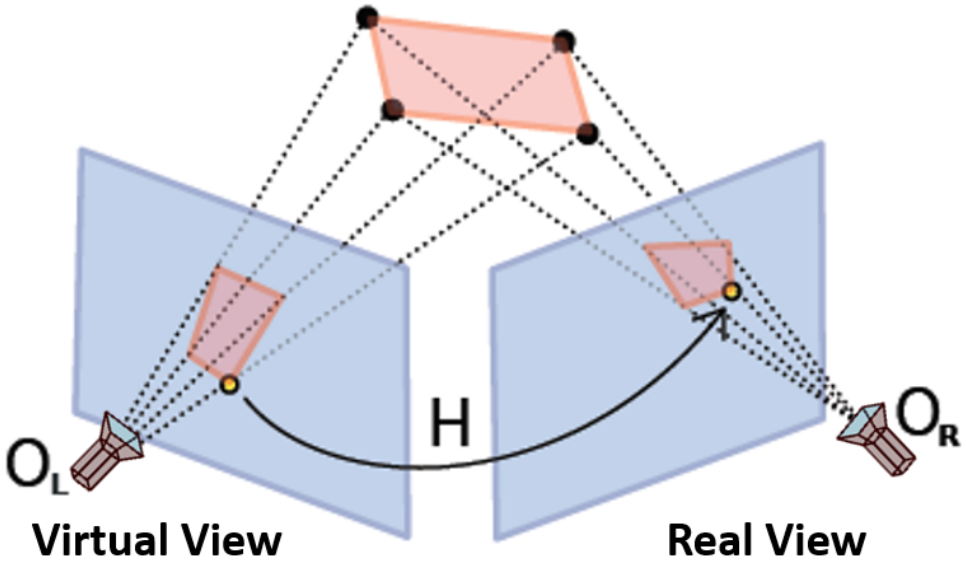


Figure 2 – Homogarphy concept

$$\mathbf{H} = \mathbf{R} + \frac{1}{d}\,\mathbf{t}\,\mathbf{n}^{\top} \tag{1}$$

H = rotationMatrix + t.* Normal

where $\mathbf{n}$ is the normal vector to the planer object.whereas d is the distance between the plane and the virtual camera image frame center as we discussed before. In case of the IPM transformation, $\mathbf{n} = (0\ 0\ 1)^T$ and d is fixed with respect to the defined Region of Interest **(ROI)** of the bird eye view image.

1.3 Rotation Matrices

Rotation matrices are used to rotate a vector into a new direction. In transforming vectors in three-dimensional space, rotation matrices are often encountered. Rotation matrices are used in two senses: they can be used to rotate a vector into a new position or they can be used to rotate a coordinate basis (or coordinate system) into a new one. In this case, the vector is left alone but its components in the new basis will be different from those in the original basis. In Euclidean space, there are three basic rotations: one each around the x, y and z axes. Each rotation is specified by an angle of rotation. The rotation angle is defined to be positive for a rotation that is counter clockwise when viewed by an observer looking along the rotation axis towards the origin.

Right hand rule

- x-axis forward, from the index figure
- y-axis left, from the middle figure
- z-axis up, from the thumb
- Roll ϕ about x-axis
- Pitch θ about y-axis
- Yaw ψ about z-axis

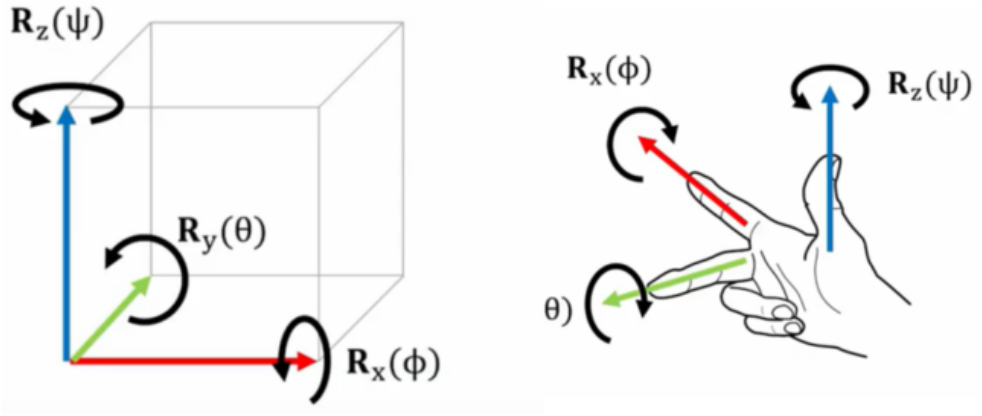


Figure 3 – Right hand rule

1.3.1 Why order of rotations is important ?

Algorithm 1: Matrices are not commutative, so just simple an example to proof that:

Result: AB $\neq$ BA

$(AB)^{-1} = B^{-1}A^{-1}$

Such that:

$$\begin{aligned} AB(AB)^{-1} &= ABB^{-1}A^{-1} \\ &= AA^{-1} \\ &= \mathrm{I} \end{aligned}$$

But

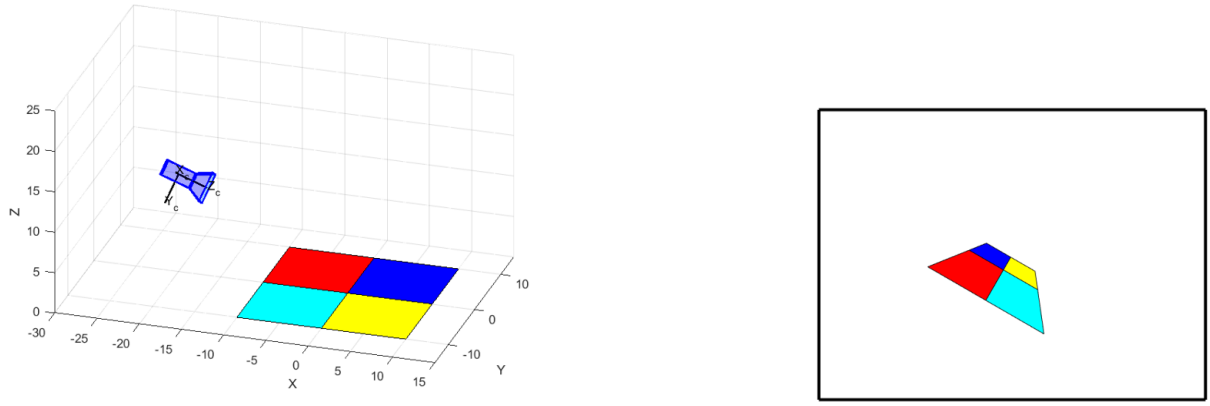
$$\begin{aligned} BA(AB)^{-1} &= BAB^{-1}A^{-1} \\ &\neq \mathrm{I} \end{aligned}$$

$\therefore$ AB $\neq$ BA Proved

Mathematical approach for pin hole camera

1.4 Modeling of the conventional camera

This section aims to recall some theoretical notions of computer vision. We will be interested in modeling the camera and more specifically the pinhole model which is nothing other than a perspective projection. It will be admitted that the camera is modeled by a projective model. The projects a point of the 3D space in a 2D image with pixel coordinates (see Figure 4). As with any model, it is necessary to characterize the parameters of the camera. We distinguish two types of parameters, those called intrinsic (focal lengths, coordinates of themain point, etc.) in opposition to the extrinsic parameters that describe the pose of thecamera with respect to the scene through a rotation matrix and a translation vector. The Intrinsic parameters can be obtained after a pre-calibration step. Note that many calibration tools are already available to designers for a lot of software: Matlab / Simulink, OpenCV, etc.



(a) 3D View (b) 2D View

Figure 4 – The capture of an image by a conventional camera

Now look at the camera marked with a pinhole model. Or a point of reference F_c (O_c, X_c, Y_c, Z_c) orthonormal attached to this camera whose origin O_c is coincided with its optical center. The marker is oriented in such a way that the Z_c axis coincided with the optical axis of the camera (figure). We will then note F_w (O_w, X_w, Y_w, Z_w) the terrestrial reference in which coordinates of the object to be captured are expressed. Finally, let P_w be a 3D point of homogeneous coordinatesP_w = (X Y Z 1) in F_w. The perspective projection of the camera transforms the point P_w into a point that will be noted p. The latter has for homogeneous coordinates p = (u v 1) in the 2D plane of the image denoted F_i. The projection equation is given by:

$$\mathbf{p} \approx \mathbf{K} \ (\mathbf{R} \qquad \mathbf{t}) \ \mathbf{P}_w \tag{2}$$

where $\approx$ denotes the equality up to scale.

The matrix **K** is the matrix of intrinsic parameters, also called calibration matrix. It will be admitted that there is no distortion phenomenon in this part. Where applicable, the parameters inherent to the distortion can easily be identified during the calibration. In the absence of distortion, the intrinsic matrix **K** can be expressed by:

$$\mathbf{K} = \begin{pmatrix} f_u & 0 & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{pmatrix}$$

where u_0 and v_0 are the pixel coordinates of the principle point. f_u and f_v are the horizontal and vertical focal length of the camera. Note that when you calibrate the camera using tools in matlab, opencv etc, then you will get the intrinsic parameters which include focal length and principle points.

It will be remembered that **R** is the matrix that characterizes the rotation between the F_c and F_w bands expressed in F_c. It can be divided into three basic rotations (Eular angles) axis of the F_c.

- ϕ around X_c denoting the roll angle
- θ around Y_c denoting the pitch angle
- ψ around Z_c denoting the yaw angle

To distribute the matrix **R** as the product of the matrix associated with these three angles. In the rest of this part, we will consider the order of the following rotations, as discussed before in algorithm why order of rotation is important.

The roll $\mathbf{R}_\phi$, then pitch $\mathbf{R}_\theta$ and finally yaw $\mathbf{R}_\psi$. so we have:

$$\mathbf{R} = \mathbf{R}_\phi \mathbf{R}_\theta \mathbf{R}_\psi \tag{3}$$

$$\mathbf{R}_\phi = \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos(\phi) & -sin(\phi) \\ 0 & sin(\phi) & cos(\phi) \end{pmatrix}$$

$$\mathbf{R}_\theta = \begin{pmatrix} cos(\theta) & 0 & sin(\theta) \\ 0 & 1 & 0 \\ -sin(\theta) & 0 & cos(\theta) \end{pmatrix}$$

$$\mathbf{R}_\psi = \begin{pmatrix} cos(\psi) & -sin(\psi) & 0 \\ sin(\psi) & cos(\psi) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

1.5 Inverse Perspective Mapping

The Inverse Perspective Mapping **(IPM)** is widespread technique in the field of computer vision, It allows that to reconstruct the top view of the plane object in an image.To use this technique uses a transformation inverse based on the projective model. This technique is already used in many application, best example The scanning of documents to using a smartphone with CamScanner.

It is also applied in agriculture field that attracts more our attention, but widely used for transport sector for such an applications, the IPM turn out to be an fascinating tool since it allows to create the top view or we can say Bird Eye's View **(BEV)** of the road. It is found behind some recent parking assistants which allows to create top view for safety parking. It is also used in algorithms to detect the obstacle.

In general terms, The IPM creates a virtual camera whose optical axis is perpendicular on the planer object. The equation of transition from the real camera to the virtual camera allows to remove the perspective effect which is present in the image for the planer object.

Consider a point p with homogeneous coordinates $\mathbf{p} = (u\ v\ 1)^\top$ belonging to an real image **I**. The **IPM** transforms **p** into **ṕ** of new homogeneous coordinates $\mathbf{\acute{p}} = (\acute{u}\ \acute{v}\ 1)^\top$ in the top view image that will be **Í**. The equation ensuring such as transformation is given by:

$$\mathbf{\acute{p}} \approx \mathbf{G}\ \ \mathbf{p} \tag{4}$$

with **G** the collinear matrix ensuring the IPM transformation form **I** to **Í**. The matrix **G** can be expressed according to the intrinsic matrix **K** and the homography matrix **H** relative to plane of the observed object. Note that **H** depends on the rotation matrix **R** and translation vector **t**. The homography matrix **H** can be calculated using follwing equation:

$$\mathbf{H} = \mathbf{R} + \frac{1}{d}\,\mathbf{t}\,\mathbf{n}^\top \tag{5}$$

where **n** denoting the vector normal to the plane object expressed in the reference of the virtual camera. while **d** is the distance between the plane and object and the center of the reference of the virtual camera.

In case of **IPM** where $\mathbf{n} = (0\ 0\ 1)^\top$ and d is set relative to the desired **R**egion **O**f **I**nterest **(ROI)** on the top view.

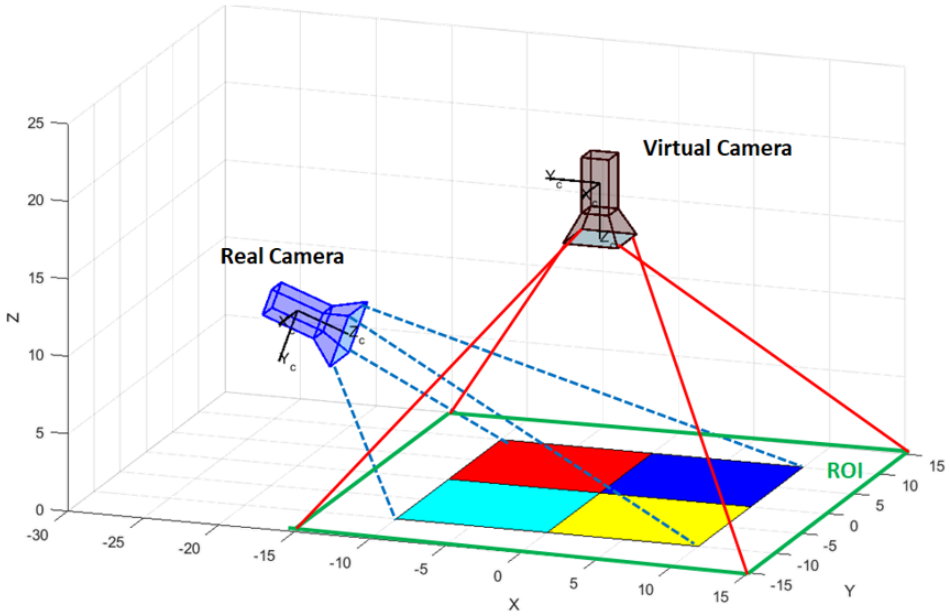


Figure 5 – Region of interest concept

In the figure 5 more illustrates about the **(ROI)**. As you can see two camera's real and virtual camera, we define the **ROI** which is **green** part which covered by virtual camera from top view such as $X_{min} = 0,\ X_{max} = 20, Y_{min} = -15, Y_{max} = 15$. Basically the general definition a region of interest **(ROI)** is a portion of an image that you want to filter or perform some other operation on.

When it comes to homography concept, in fact for each entire pixel of the top view image **Í**, its coordinates are calculated in the initial image **I** using the inverse of equation 4. Although the pixels chosen in **Í** have integer coordinates, this is rarely the case when we calculate their coordinates in **I**. It is therefore necessary to interpolate to the gray level information from the initial image **I**.

1.6 Reconstruction of the top view

The global rotation matrix $\mathbf{R}$ but we take into an account the matrix $\mathbf{R}_c$ which makes it possible to orientate mechanically the reference of the camera $\mathbf{F}_c$. so finally the rotation matrix $\mathbf{R}$ can be expressed by:

$$\mathbf{R} = \mathbf{R}_c\mathbf{R}_\phi\mathbf{R}_\theta\mathbf{R}_\psi \tag{6}$$

In simulation we also take into an account yaw angle to justify our result that why we have equation (6), so $\mathbf{R}_c$ can be expressed by:

$$\mathbf{R}_c = \begin{pmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \end{pmatrix}$$

The collineation matrix $\mathbf{G}$ from equation (4) can be easily identified in two steps. If we use the assumption $\mathbf{Z} = 0$ and $\mathbf{t} = (0\ 0\ -h_c)^\top$ in equation (1), we obtain

$$\mathbf{p} \approx \mathbf{K}\ \mathbf{M}\ \mathbf{P}_w \tag{7}$$

where $\mathbf{M} = (r_1\ \ r_2\ \ -h_c r_3)$ with r_i the column i of the rotation matrix $\mathbf{R}$. The matrix $\mathbf{M}$ can be expressed by:

$$\mathbf{M} = \begin{bmatrix} -sin_\phi sin_\theta & -cos_\phi & -h_c sin_\phi cos_\theta \\ cos_\phi sin_\theta & -sin_\phi & -h_c cos_\phi cos_\theta \\ cos_\theta & 0 & -h_c sin_\theta \end{bmatrix}$$

p_w is a point of homogeneous coordinates $\mathbf{p}_w = (X\ Y\ 1)^\top$ lying on the plane $Z = 0$ in the frame F_v. whereas $\mathbf{p} = (u\ v\ 1)^\top$ is its projection into the image $\mathbf{I}$. The relation between p_w and $\acute{p}$ is obtained by cropping the top-view regarding the desired **(ROI)** and output **BEV** resolution. If the output resolution is $(\acute{n} \times \acute{m})$, then;

$$\acute{\mathbf{p}} = \mathbf{S}\ \mathbf{p}_w \tag{8}$$

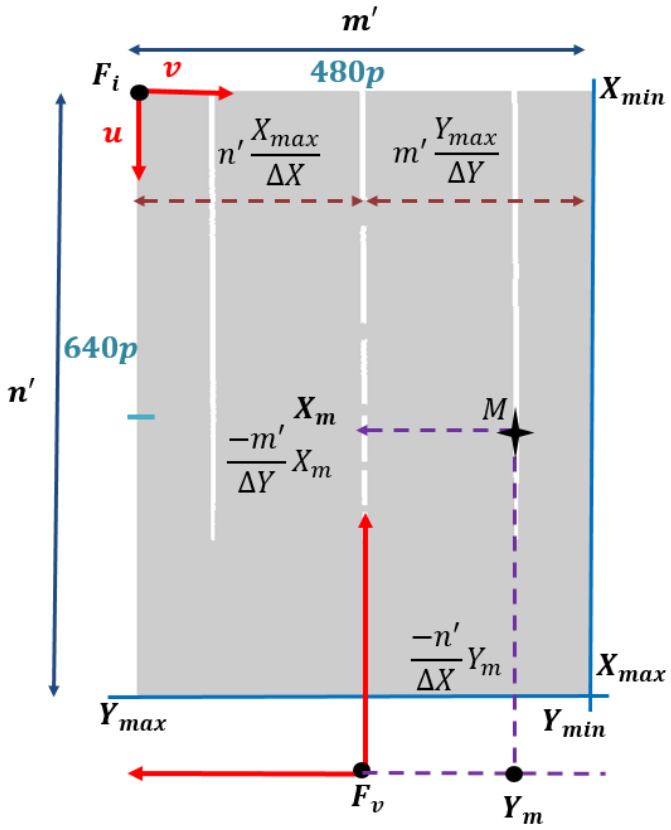


Figure 6 – Define **S** matrix with image

As you can see figure(6), we calculate the **S** with virtual camera image **Í** here the mathematical expressions

Pixel	**Matrix**	**1m**
$\acute{m}$	$X_{max} - X_{min} = \Delta X$	$1m = \frac{\acute{m}}{\Delta X}$
$\acute{n}$	$Y_{max} - Y_{min} = \Delta Y$	$1m = \frac{\acute{n}}{\Delta Y}$

$$u' = \frac{X_{max}\acute{m}}{\Delta X} - X_m\frac{\acute{m}}{\Delta X}$$

$$v' = \frac{Y_{max}\acute{n}}{\Delta Y} - Y_m\frac{\acute{n}}{\Delta Y}$$

$$\begin{bmatrix} u' \\ v' \end{bmatrix} = \begin{bmatrix} 0 & -\acute{m}\frac{1}{\Delta Y} \\ -\acute{n}\frac{1}{\Delta X} & 0 \end{bmatrix} \begin{bmatrix} X_m \\ Y_m \end{bmatrix} + \begin{bmatrix} \acute{m}\frac{Y_{max}}{\Delta Y} \\ \acute{n}\frac{X_{max}}{\Delta X} \end{bmatrix}$$

After solving the matrix operation we get;

$$\begin{bmatrix} u' \\ v' \end{bmatrix} = \begin{bmatrix} 0 & -\acute{m}\frac{1}{\Delta Y} & \acute{m}\frac{Y_{max}}{\Delta Y} \\ -\acute{n}\frac{1}{\Delta X} & 0 & \acute{n}\frac{X_{max}}{\Delta X} \end{bmatrix} \begin{bmatrix} X_m \\ Y_m \end{bmatrix}$$

Finally we get;

$$\begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & -\acute{m}\frac{1}{\Delta Y} & \acute{m}\frac{Y_{max}}{\Delta Y} \\ -\acute{n}\frac{1}{\Delta X} & 0 & \acute{n}\frac{X_{max}}{\Delta X} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_m \\ Y_m \\ 1 \end{bmatrix}$$

After calculating the matrix formation, we are able to get the **S** matrix,

$$\mathbf{S} = \begin{pmatrix} 0 & -\acute{m}\frac{1}{\Delta Y} & \acute{m}\frac{Y_{max}}{\Delta Y} \\ -\acute{n}\frac{1}{\Delta X} & 0 & \acute{n}\frac{X_{max}}{\Delta X} \\ 0 & 0 & 1 \end{pmatrix}$$

$\therefore \quad \Delta X = X_{max} - X_{min}$ and $\Delta Y = Y_{max} - Y_{min}$
So finally form the equations (4),(7),(8), it expressed by;

$$\mathbf{G} = (\mathbf{K}\ \mathbf{M}\ \mathbf{S}^{-1})^{-1} \tag{9}$$

Form this equation (9), it is easy to develop algebraic expressions for calculating the coordinates of a pixel $(u'\ v')$ belonging to the image $\acute{I}$ in function pixel coordinates $(u\ \ v)$ in **I**. These expressions are given by:

$$u' = \frac{n}{\Delta Y} \frac{Y_{max} f_u f_v s_\theta + \Delta u(Y_{max} f_u s_\phi c_\theta + h_c f_v c_\phi) + \Delta v(Y_{max} f_u c_\phi c_\theta - h_c f_u s_\phi)}{f_u f_v s_\theta + f_u s_\phi c_\theta \Delta u + f_u c_\phi c_\theta \Delta v}$$

$$v' = \frac{m}{\Delta X} \frac{X_{max} f_v f_u s_\theta - h_c f_u f_v c_\theta + \Delta u(X_{max} f_v s_\phi c_\theta + h_c f_v s_\phi c_\theta) + \Delta v(X_{max} f_u c_\phi c_\theta + h_c f_u c_\phi s_\theta}{f_u f_v s_\theta + f_v s_\phi c_\theta \Delta u + f_u c_\phi c_\theta \Delta v}$$

with $\Delta u = u - u_0$, $\Delta = v - v_0$. The terms ΔX and ΔY have been defined previously. For the sake of simplicity, the notations s_ϕ and c_θ denote respectively $\sin(\theta)$ and $\cos(\theta)$.

It is possible to invert the previous system of equation $(u'\ \ v')$. which means that for one pixel $(u',\ v')$ in **Í**, we can calculate its coordinates (u, v) in **I** with the following equations:

$$u = \frac{(-f u \sin_\phi \sin_\theta + u_0 \cos_\theta)X - f_u \cos_\phi Y + (f_u \sin_\phi \cos_\theta + u_0 \sin_\theta)Z}{\cos_\theta X + \sin_\theta Z}$$

$$v = \frac{(-f_v \cos_\phi \sin_\theta + v_0 \cos_\theta)X + f_v \sin_\phi Y + (f_v \cos_\phi \cos_\theta + v_0 \sin_\theta)Z}{\cos_\theta X + \sin_\theta Z}$$

$$X = X_{\max} - u' \frac{X_{\max} - X_{\min}}{m}; \qquad Y = Y_{\max} - v' \frac{Y_{\max} - Y_{\min}}{n}; \qquad Z = hc \quad \therefore hc \textbf{ height of camera}$$

To summarize the top view of the virtual camera is obtained in two stages. First by calculating coordinates (u, v) in **I** of each integer pixel (u', v') belonging to **Í**. To do this, the (u, v) is used. Secondly, by interpolating the calculated (u, v) decimal coordinates in image **I**. An example of the top view of the virtual camera obtained with the preceding algorithm is presented by two scenario.

1.7 Bird Eye View Concept

It is transformation technique to generate a top view perspective of an image as shown in figure below. This technique can be classified under digital image precessing as geometrical image modification.

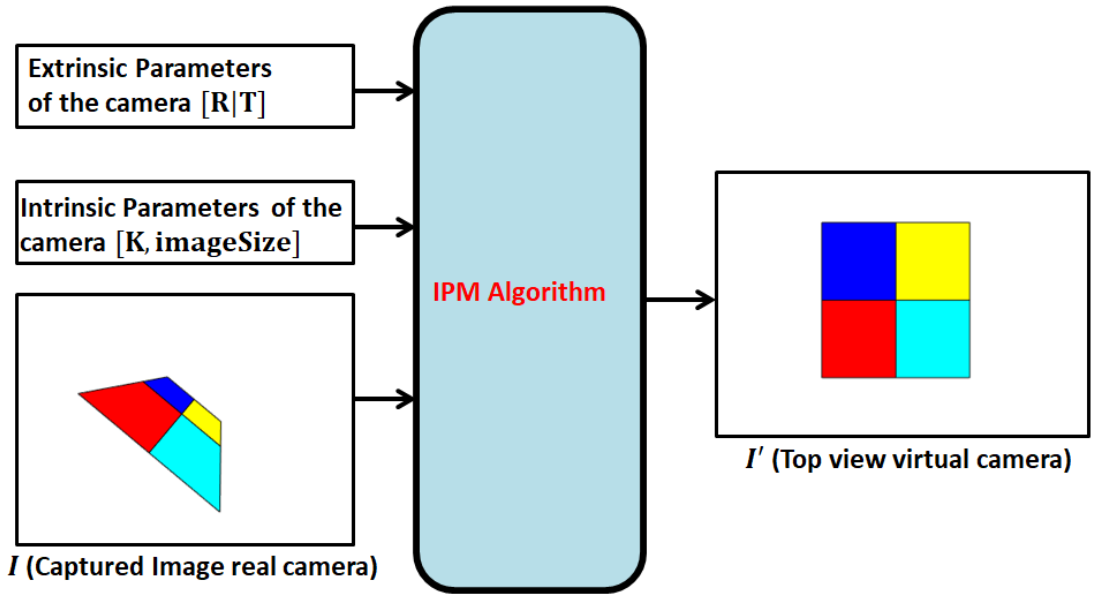


Figure 7 – Bird eye view technique using **IPM** Algorithm

As you can see the figure (7), it is clearly shows that if you want to use **BEV** technique (top view), the algorithm must known the camera parameters such as **E**xtrinsic and **I**ntrinsic parameter which allows to give you rotational, translation, imageSize, focal length and principle point. Note that when we have all information for camera, then do calibration process to get all parameters. To getting camera parameter there are many tools on internet such as **MatLab**, **OpenCV**, **NodeJS** etc.

The **IPM** algorithm performs two scenarios are as follows:

Scenario 1

Firstly, In this scenario we are using square box with four different colors to understand better way of **IPM**, after using **IPM** technique or we can say the pixel value equation of image (u, v), it's allows to create top view of image.

In other words, the BEV image I' is obtained by warping each point in I' onto I using the inverse mapping $\mathbf{G}^{-1}$ and then the intensity of the image is calculated by interpolation local pixel intensity in the origin image I.

As we addressed the behaviour of **IPM** to create a top view, The figure shown with camera roll angle is $40°$ and pitch angle is $20°$, but created bird eye is in gray scale property.

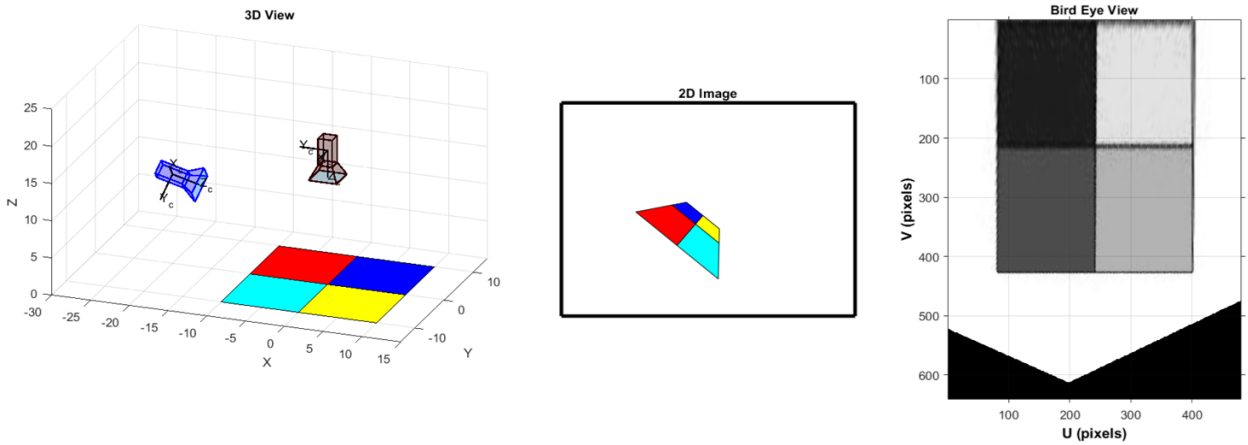


Figure 8 – S-1 **BEV** with four different color

Scenario 2

Secondly, The main of the concept using **IPM** to with road structure, so line equation in matlab allows to create a line looks like a road lane, The three lane left, right and middle one which is dotted lane.

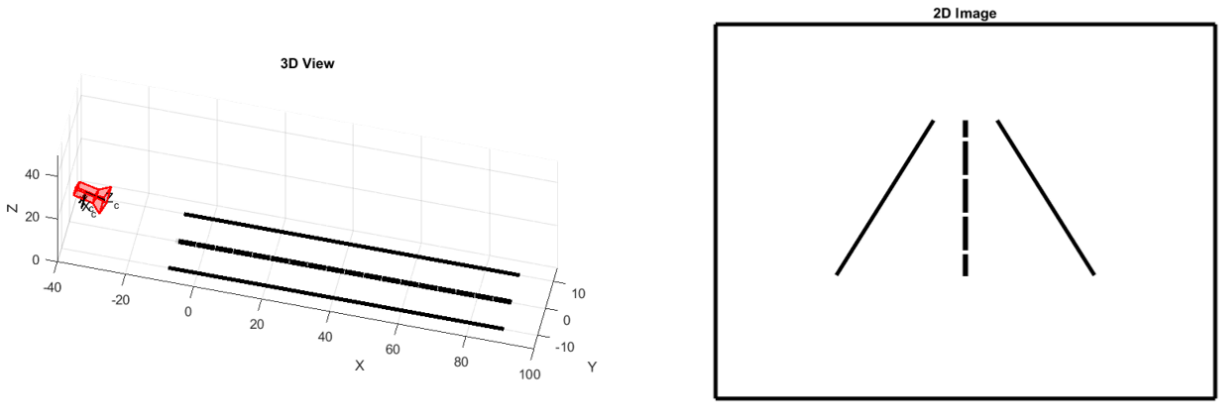


Figure 9 – S-2 Camera projection with three lane road marker

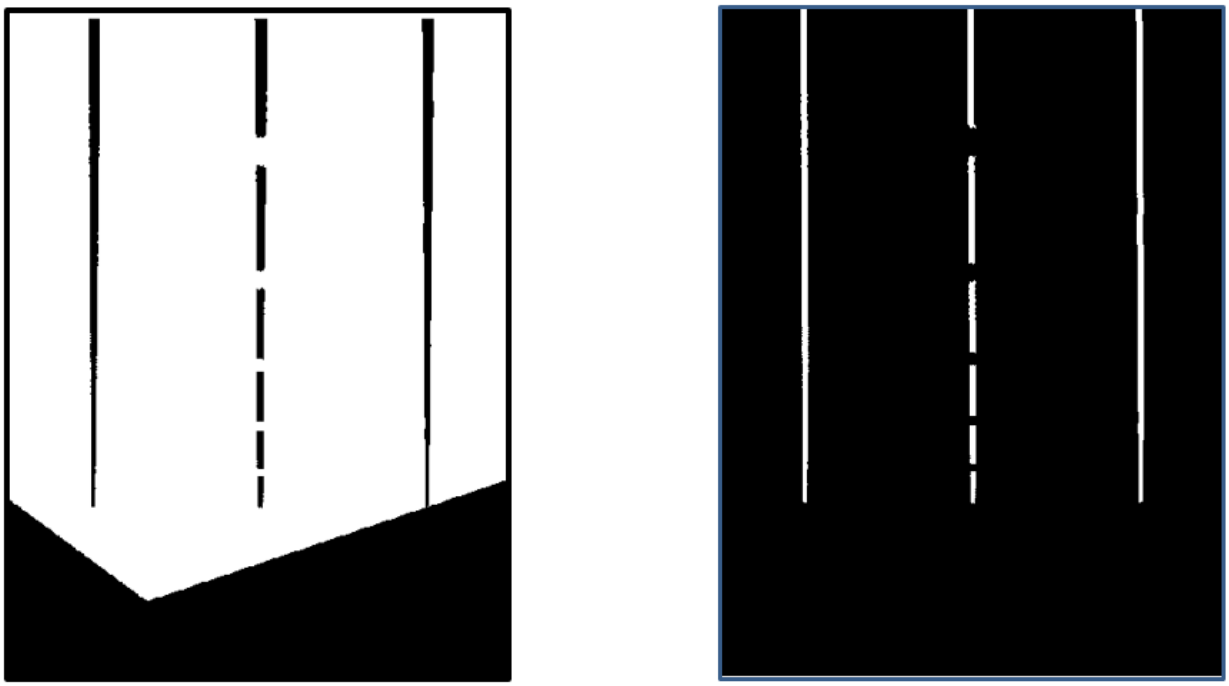
After getting 2D view from the camera, Now I am going to use **IPM** equation, from that equation we will get the top view (Virtual Camera)

When we have the top view from the virtual camera then we used the detection algorithm, Moreover matlab allows a lot algorithm to detect the line such as hough transform

1.8 Lane detection

Lane detection is a well-researched area of computer vision with applications to autonomous vehicles and driver assistance systems. This is partly because, despite the apparent simplicity of the white markings on a dark road, making it very difficult to identify the markings on different types of roads. These difficulties are of an occlusion in the shadow of other vehicles, changes in the roadway itself, and different types of road markings. A lane detection system must collect all types of markers roads confusion and filtered to give a reliable estimate of the path of the vehicle's position.

Lane detection plays an important role in driver assistance systems. In general, the steps of lane detection localize lane boundaries in the images of the specified path, and can help to estimate the geometry of the floor and lateral position ego vehicle on the road, Lane detection in intelligent cruise control environments for Lane Departure Warning, modeling the way, and so on.



(a) top view from virtual camera (b) line detection

Figure 10 – Bird Eye view for lane and lane detection

1.8.1 Canny Edge Detection

By applying the optimum global thresholding to selected part of image, ROI, we have binary image as the input for this step. In this step, to find lane boundaries in the image we use one of the edge detection methods called Canny Edge Detection.

Canny Edge detector most commonly used for step edges due to Optimal, then is corrupted by white noise. The objective is the edges detected must be as close as possible to the true edges the number of local maxima around the true edge should be minimum.

Canny edge detection basically uses gradient vector of an intensity image. Lane boundaries have high contrast in the image, and this feature yields high values of gradient vector by which we can find the edge direction, which is orthogonal to gradient vector. Many edge detection methods are based on this principle, but the efficiency levels are different. One of the best and efficient methods is canny edge detection. The most important characteristics of canny method are that the error rate of this method is low because this algorithm uses double thresholding, hysteresis thresholding. Hysteresis threshold, double thresholding, suppresses the pixels that are not related to edges. Therefore, the detected edge is really close to true place. We should also mention that canny edge detector is very sensitive to noise; therefore, we smooth the image by a low pass filter to reduce the effect of noise.