A Tutorial On Camera Models

Bryan Poling

University of Minnesota

1 Introduction

A *camera model* is a function which maps our 3-dimensional world onto a 2-dimensional plane, called the *image plane*. Generally, this function is designed to closely model a real-world, physical camera. There are many camera models of varying complexity, and a natural dividing line which helps categorize them is whether or not they are able to capture *perspective*. Perspective, or the *perspective effect* is simply the property that objects far away from us appear smaller than objects up close. This is obviously the case with human vision, and with most cameras in the real world. However, there are instances where this is not the case, and there are many more instances where, even though the perspective effect is present, it is acceptable (and sometimes advantageous) to ignore it.

We will start in §2 by developing *homogeneous coordinates*, which are pervasive in the subject of computer vision, and needed to work with all but the simplest camera models. Then, in §3 we explore the most general models in common use, all of which are capable of capturing perspective. Finally, in §4 we study the simpler models which are not capable of capturing perspective.

2 Homogeneous Coordinates

Homogeneous Coordinates provide another way of representing points in space. To represent a point in $\mathbb{R}^n$ using standard Euclidean coordinates, we use a vector of n elements, encoding what combination of the standard basis vectors is needed to construct the point. With homogeneous coordinates we use a vector of $n + 1$ elements instead. The conversion between homogeneous and Euclidean coordinates is given by:

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \\ w \end{bmatrix} \leftrightarrow \begin{pmatrix} x_1/w \\ x_2/w \\ \vdots \\ x_n/w \end{pmatrix} \tag{1}$$

We use square brackets on the left to specifically indicate that the vector should be interpreted as a set of homogeneous coordinates. We will use parenthesis for euclidean coordinates, and when no specific interpretation is intended (ie. for formal manipulation). The $\leftrightarrow$ symbol indicates that these vector represent the same point in space. These are conventions we will continue to use for vectors.

The first thing to notice is that homogeneous coordinates over-parametrize space. In particular, if we multiply a homogeneous vector by a non-zero constant, this constant is multiplied into every x_i, as well as w. Thus, the corresponding euclidean vector is unaffected. In summary, homogeneous representation is scale-invariant. The proper way to "scale" a point by a constant $\alpha \neq 0$ is to multiply all but the last coordinate of the homogeneous vector by α, or equivalently to divide the last coordinate by α.

Homogeneous coordinates are an interesting subject on their own, but our interest in them here comes from a few specific properties. First, one may recall that when using euclidean coordinates, we can effect an arbitrary linear transformation on a point by left-multiplying by an appropriate square matrix. With homogeneous coordinates, we can effect an arbitrary *affine transformation* in this way. That is, we can apply any linear transformation, accompanied by any translation by left-multiplying by the correct matrix. To see this, let $\mathbf{B}$ be any 3-by-3 matrix (which one would apply to euclidean vectors in $\mathbb{R}^3$). Let $\mathbf{T}$ be a euclidean vector in $\mathbb{R}^3$, representing a translation amount. Consider a point with the following homogeneous and euclidean representations:

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ w \end{bmatrix} \leftrightarrow \begin{pmatrix} x_1/w \\ x_2/w \\ x_3/w \end{pmatrix} \tag{2}$$

Then observe:

$$\left(\begin{array}{ccc|c} & \mathbf{B} & & \mathbf{T} \\ 0 & 0 & 0 & 1 \end{array}\right) \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ w \end{bmatrix} = \begin{bmatrix} \mathbf{B} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + w\mathbf{T} \\ w \end{bmatrix} \leftrightarrow \mathbf{B} \begin{pmatrix} x_1/w \\ x_2/w \\ x_3/w \end{pmatrix} + \mathbf{T} \tag{3}$$

Thus, the homogeneous operation:

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ w \end{bmatrix} \longrightarrow \left(\begin{array}{ccc|c} & \mathbf{B} & & \mathbf{T} \\ 0 & 0 & 0 & 1 \end{array}\right) \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ w \end{bmatrix} \tag{4}$$

is equivalent to the following operation in euclidean coordinates:

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} \longrightarrow \mathbf{B} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} + \mathbf{T} \tag{5}$$

The other important property of homogeneous coordinates is that "perspective projection" can also be implemented through matrix multiplication. The relevance of this will be apparent in the next section, but for now, imagine that the following operation is important to us:

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} \longrightarrow \begin{pmatrix} y_1/y_3 \\ y_2/y_3 \end{pmatrix} \tag{6}$$

This is a non-linear operation, and so cannot (using standard euclidean coordinates) be expressed using matrix multiplication. However, this is equivalent to the following:

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ w \end{bmatrix} \longrightarrow \begin{pmatrix} 1\,0\,0\,0 \\ 0\,1\,0\,0 \\ 0\,0\,1\,0 \end{pmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ w \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \tag{7}$$

The homogeneous vector on the left represents the point with euclidean coordinates:

$$\begin{pmatrix} x_1/w \\ x_2/w \\ x_3/w \end{pmatrix} \tag{8}$$

and the vector on the right represents the point with euclidean coordinates:

$$\begin{pmatrix} x_1/x_3 \\ x_2/x_3 \end{pmatrix} = \begin{pmatrix} \frac{x_1/w}{x_3/w} \\ \frac{x_2/w}{x_3/w} \end{pmatrix} \tag{9}$$

Thus, this simple matrix multiply, when using homogeneous coordinates, has the same effect as (6) has when using euclidean coordinates.

3 Cameras Which Capture The Perspective Effect

3.1 The Pinhole, or Central-Projection Camera

The simplest camera in this category is the pinhole camera. In this model, we place a plane (this will be the image plane) some distance from a point which we will call the camera center. We map a point into the image plane by translating the point on a straight line towards the camera center, until it intersects the image plane.

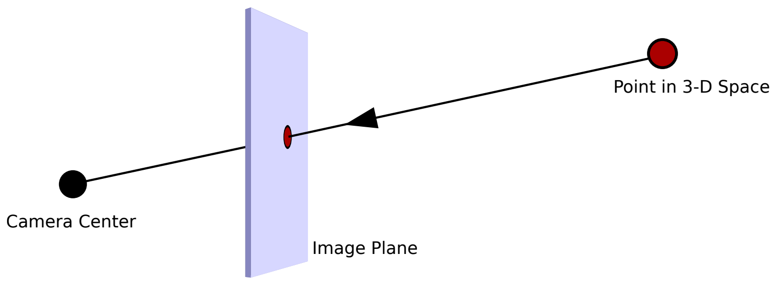

Fig. 1: Operation of projecting a point into the image plane for a pinhole camera

For simplicity, we will first assume that the camera center is at the origin of a 3-D coordinate frame. We will also assume that the image plane is positioned parallel to the xy-plane, at position $z = f$. We will define a 2-D coordinate system in the image plane with origin at position $(0, 0, f)^T$ (in 3-D euclidean coordinates). The x and y axes of this new frame will be parallel to the x and y axes of the 3-D frame. We will often use the term “image coordinates” when we are referring to this new frame. Imagine a point in 3-D space with y-coordinate 0, and lets see how it gets projected into the image plane.

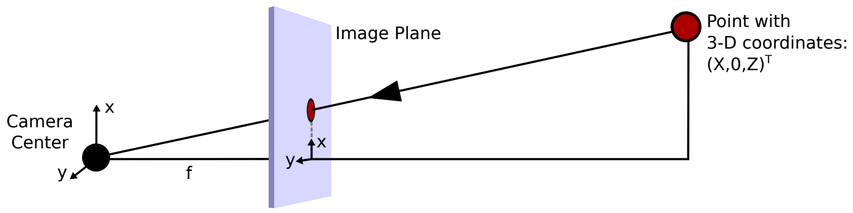

Fig. 2: Similar triangles in pinhole camera projection

By looking at similar triangles, we can see that the euclidean image coordinates of the projection of the point in question are $f(X/Z, 0)^T$. Similarly, if a point in space has general euclidean coordinates $(X, Y, Z)^T$, then the projection of this point is defined by:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \to f \begin{pmatrix} X/Z \\ Y/Z \end{pmatrix} \tag{10}$$

This operation was what we called “perspective projection” in section 2 (although we scale by f here), and we showed that this operation has a nice, simple form if we use homogeneous coordinates. The ability to use matrices to represent our camera models, and also use them to chain together compositions of operations greatly simplifies the notation in this subject. It is for this reason that we use homogeneous coordinates. In homogeneous coordinates (10) becomes:

$$\begin{bmatrix} X \\ Y \\ Z \\ w \end{bmatrix} \to \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{bmatrix} X \\ Y \\ Z \\ w \end{bmatrix} \tag{11}$$

In the above, we assumed that the camera was centered at the origin of the 3-D coordinate frame, and was focused down the z-axis. In general, we may want to have multiple cameras, with different poses[1] at the same time. To facilitate this, we define two distinct 3-D coordinate frames. The first is exactly the frame we used above, with origin at the camera center, and oriented so that its z-axis is orthogonal to the image plane; this is called the *camera frame*. The other frame will be called the *world frame*. A point in this frame may be referred to as a world point, or we may say it is in world-coordinates. In general, there may be an arbitrary rotation and translation required to move between

[1] A cameras *pose* refers to the combination of its position and orientation in space.

the camera frame and the world frame. With this abstraction, we can have multiple cameras. There will always be a single world frame, and a different camera frame for each camera.

It is convenient to be able to build the cameras pose into the camera model, so that we can input a point in world coordinates, without having to worry about performing coordinate transformations manually. So, lets assume that $\mathbf{X}_{world}$ and $\mathbf{X}_{cam}$ are the homogeneous coordinates of a single point in the world and camera frames, respectively, and that they are related by:

$$\mathbf{X}_{cam} = \left(\begin{array}{ccc|c} & \mathbf{R} & & \mathbf{C} \\ 0 & 0 & 0 & 1 \end{array}\right) \mathbf{X}_{world} \tag{12}$$

Then, the function which maps a point in homogeneous world coordinates to image coordinates is given by:

$$\begin{bmatrix} X \\ Y \\ Z \\ w \end{bmatrix} \to \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \left(\begin{array}{ccc|c} & \mathbf{R} & & \mathbf{C} \\ 0 & 0 & 0 & 1 \end{array}\right) \begin{bmatrix} X \\ Y \\ Z \\ w \end{bmatrix} \tag{13}$$

We will use $\mathcal{P}$ to denote the camera projection operation. Then, we can write this more simply as:

$$\mathcal{P}\left(\mathbf{X}_{world}\right) = \mathbf{P}\mathbf{X}_{world} \tag{14}$$

$$\mathbf{P} = \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix} \left(\mathbf{R} \mid \mathbf{C}\right) \tag{15}$$

The matrix $\mathbf{P}$ is called the cameras *projection matrix*. Notice that since the last column of the first matrix in (13) is all 0's, we were able to delete that column, and the corresponding row in the matrix on the right.

3.2 Slight Generalizations of the Pinhole Camera

The camera model from section 3.1 captures the main idea behind perspective projection, but it made some simplifying assumptions which can be easily relaxed. The first is that the two axes in the image plane are assumed to be identical. This may be problematic. For instance, it is common for digital CCDs (the optical sensor in a digital camera) to have non-square pixel elements. This effect is well-modeled by a pinhole camera with two different focal lengths for the x and y camera axes. With this relaxation, the camera projection matrix is:

$$\mathbf{P} = \begin{pmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \left(\mathbf{R} \mid \mathbf{C}\right) \tag{16}$$

where f_x and f_y are the effective focal lengths for the x and y camera axes, respectively. Additionally, we may wish to re-position the origin in the image plane (rather than require it to be the intersection of the z-axis of the camera frame with the image plane). To accomplish this, we simply translate image, following the perspective projection. Our camera matrix becomes:

$$\mathbf{P} = \begin{pmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \left(\mathbf{R} \mid \mathbf{C} \right) = \begin{pmatrix} f_x & 0 & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{pmatrix} \left(\mathbf{R} \mid \mathbf{C} \right) \tag{17}$$

where $(x_0, y_0)^T$ is the translation applied to the image after perspective projection.

Both of the generalizations applied to the pinhole camera model in this section can be realized by using a standard pinhole camera model, and applying an appropriate affine transformation to the resulting image.

3.3 The General Projective Camera

The camera models from sections 3.1 and 3.2 are both expressed as:

$$\mathcal{P}\left(\mathbf{X}_{world}\right) = \mathbf{P}\mathbf{X}_{world} \tag{18}$$

for an appropriate 3-by-4 matrix $\mathbf{P}$. A general projective camera simply removes all restrictions on the elements of this matrix. That is, the *projective camera* is defined as follows [1]:

$$\mathcal{P}(\mathbf{X}_{world}) = \begin{pmatrix} T_{11} & T_{12} & T_{13} & T_{14} \\ T_{21} & T_{22} & T_{23} & T_{24} \\ T_{31} & T_{32} & T_{33} & T_{34} \end{pmatrix} \mathbf{X}_{world} \tag{19}$$

where each T_{ij} is arbitrary. It should be noted that some still require that $\mathbf{P}$ be of full rank (to ensure that the image of $\mathbb{R}^3$ fills the image plane) [2, p. 157].

4 Cameras Which Do Not Capture The Perspective Effect

The models seen so far are rather general, and all of them have the ability to capture perspective. However, when imaging objects far away from a camera, small differences in depth become less apparent. In this case perspective can often be ignored in a camera model. We will explore a few models which ignore perspective; we will call these *non-perspective cameras*. Of the non-perspective camera models in frequent use, the most general model that the author is aware of is called the *affine camera*. Two others are the *orthographic camera* and the *weak perspective camera*. The orthographic camera is the simplest to understand, and we will see that the other two are just slight generalizations of this model. We will therefor start with the orthographic camera and then proceed to generalize it.

4.1 The Orthographic Camera

In the *orthographic camera* model, the image of a world point is found by simply translating the world point parallel to the optical axis[2] until it lands in the image plane. An example of where this model is appropriate would be if one holds an object above the ground at noon on a sunny day (so the sun is directly overhead) and views the shadow of the object on the ground as the image of the object. Since the sun is so far away from us, all of the light rays hitting the object are effectively parallel, resulting in the described effect. In principal, if one were to move the object closer to, or farther away from the ground, the shadow would not change in size. This demonstrates that no perspective information survives the projection.

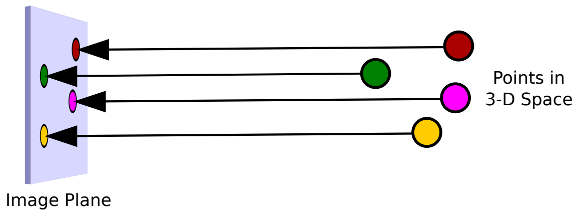

Fig. 3: Operation of projecting points into the image plane for an orthographic camera

To derive this cameras projection matrix, let $\mathbf{R}$ represent a 3-by-3 rotation matrix and $\mathbf{C}$ represent a 3-by-1 euclidean vector s.t. if $\mathbf{X}$ is a euclidean vector in the world coordinate frame, then $\mathbf{X}' = \mathbf{R}\mathbf{X} + \mathbf{C}$ gives the same vector, with euclidean representation, coordinatized in the camera frame.

Then, to map a point $\mathbf{X}$ (now with homogeneous representation in the world frame) to its image point $\mathbf{x}$ (with homogeneous representation in the image plane), we first express $\mathbf{X}$ in the camera frame, and then eliminate the z-component of the vector. This is accomplished as follows:

$$\mathbf{x} = \begin{pmatrix} 1&0&0&0 \\ 0&1&0&0 \\ 0&0&0&1 \end{pmatrix} \left(\begin{array}{ccc|c} & \mathbf{R} & & \mathbf{C} \\ 0 & 0 & 0 & 1 \end{array} \right) \mathbf{X} \tag{20}$$

This reduces to $\mathbf{x} = \mathbf{P}\mathbf{X}$ where:

$$\mathbf{P} = \left(\begin{array}{ccc|c} \leftarrow & \mathbf{R}_{(1,:)} & \rightarrow & \mathbf{C}_1 \\ \leftarrow & \mathbf{R}_{(2,:)} & \rightarrow & \mathbf{C}_2 \\ 0 & 0 & 0 & 1 \end{array} \right) \tag{21}$$

4.2 The Weak Perspective Camera

One may notice that the orthographic camera does not allow any sort of “zooming”. That is, we flatten out space through orthogonal projection, but we don’t dilate, or scale

[2] The optical axis is in the direction orthogonal to the image plane

the result. The *weak perspective* camera is nothing more than an orthographic camera, followed by a scaling of the resulting image. Notice that the scaling is applied after orthogonal projection, which destroys all perspective information. Thus, the weak perspective camera is equally incapable of capturing perspective (its name may therefor seem misleading - indeed if I were first naming this, I would probably have called it a "scaled orthographic camera"; a name which is generally reserved for the case where the scaling factor is the same in both the x and y directions [2, p. 171]).

Recall that if we want to effect an affine transformation in homogeneous coordinates, this can be accomplished by left-multiplying by the appropriate matrix. In our case, our desired transformation is simply scaling the x and y axes (lets say by $\alpha \neq 0$ and $\beta \neq 0$, respectively). This can be applied by left multiplying by the following matrix:

$$\begin{pmatrix} \alpha & 0 & 0 \\ 0 & \beta & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{22}$$

Then the camera projection matrix for a weak perspective camera becomes:

$$\mathbf{P} = \left(\begin{array}{ccc|c} \leftarrow & \alpha\mathbf{R}_{(1,:)} & \rightarrow & \alpha\mathbf{C}_1 \\ \leftarrow & \beta\mathbf{R}_{(2,:)} & \rightarrow & \beta\mathbf{C}_2 \\ 0 & 0 & 0 & 1 \end{array}\right) \tag{23}$$

Frequently, it is convenient to use a weak perspective camera to approximate a pinhole perspective camera. As was mentioned before, for such an approximation to be reasonable, we need to be imaging objects that are reasonably far away from the camera. Specifically, we need the differences in depth of the imaged objects to be small, compared to the average depth of all of the objects, Z_{ave}. When this assumption is satisfied, we choose α so that an object at depth Z_{ave} appears to have the same size in the image plane with both camera models.

If f is the focal length of the pinhole perspective camera we are trying to approximate, an object of unit size, at depth Z_{ave}, will appear to have size f/Z_{ave} in the image plane. If we use orthographic projection instead, an object of unit size (regardless of how far away it is) will have an image of unit size in the image plane. Thus, to make our weak-perspective camera approximate the pinhole camera, we need to scale both axes of the orthographic image by $\alpha = \beta = f/Z_{ave}$. In this case, the camera projection matrix is:

$$\mathbf{P} = \left(\begin{array}{ccc|c} \leftarrow & f/Z_{ave}\mathbf{R}_{(1,:)} & \rightarrow & f/Z_{ave}\mathbf{C}_1 \\ \leftarrow & f/Z_{ave}\mathbf{R}_{(2,:)} & \rightarrow & f/Z_{ave}\mathbf{C}_2 \\ 0 & 0 & 0 & 1 \end{array}\right) \tag{24}$$

Since matrices for homogeneous vectors are scale invarient, this can be written equivalently as:

$$\mathbf{P} = \left(\begin{array}{ccc|c} \leftarrow & \mathbf{R}_{(1,:)} & \rightarrow & \mathbf{C}_1 \\ \leftarrow & \mathbf{R}_{(2,:)} & \rightarrow & \mathbf{C}_2 \\ 0 & 0 & 0 & Z_{ave}/f \end{array}\right) \tag{25}$$

4.3 The Affine Camera

We constructed the weak perspective camera by starting with an orthogonal projection and following it by a uniform scaling. The resulting camera model was non-perspective because the first operation in its construction (orthogonal projection) destroyed all perspective information. It would seem then that we could construct other non-perspective camera models by following orthogonal projection with more general transformations than just scaling. This is indeed the case, and our final model, the *affine camera*, is nothing more than orthogonal projection followed by an arbitrary affine transformation. This is the most general non-perspective camera model considered thus far. The weak perspective camera is a special case of an affine camera, by restricting our affine transformation to simple dilation. The orthogonal camera can similarly be viewed as a special case of the weak perspective camera, where our scaling amount is restricted to unity.

The affine camera is typically defined by the following map [1]:

$$\mathbf{x} = \begin{pmatrix} T_{11} & T_{12} & T_{13} & T_{14} \\ T_{21} & T_{22} & T_{23} & T_{24} \\ 0 & 0 & 0 & T_{34} \end{pmatrix} \mathbf{X} \quad \text{where each } T_{ij} \text{ is arbitrary.} \tag{26}$$

In this form it is not easy to see that this is an orthographic projection followed by some affine transformation. To show this, we will assume we have been given a matrix of the form in (26). We will consider an arbitrary composition of an orthogonal projection, followed by an affine transformation, and show that we can design such a composition to agree with the provided transformation. First, observe that since the matrix in (26) is applied to homogeneous vectors, the matrix can be scaled by any non-zero amount without changing the transformation. Hence, we loose nothing by assuming our given matrix has $T_{34} = 1$. Our arbitrary composition looks like:

$$F(\mathbf{X}) = \begin{pmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ 0 & 0 & 1 \end{pmatrix} \left(\begin{array}{ccc|c} \leftarrow & \mathbf{R}_{(1,:)} & \rightarrow & \mathbf{C}_1 \\ \leftarrow & \mathbf{R}_{(2,:)} & \rightarrow & \mathbf{C}_2 \\ 0 & 0 & 0 & 1 \end{array} \right) \mathbf{X} \tag{27}$$

Multiplying this out we get:

$$F(\mathbf{X}) = \mathbf{B}\mathbf{X} \tag{28}$$

where:

$$\mathbf{B} = \begin{pmatrix} A_{11}R_{11} + A_{12}R_{21} & A_{11}R_{12} + A_{12}R_{22} & A_{11}R_{13} + A_{12}R_{23} & A_{11}C_1 + A_{12}C_2 + A_{13} \\ A_{21}R_{11} + A_{22}R_{21} & A_{21}R_{12} + A_{22}R_{22} & A_{21}R_{13} + A_{22}R_{23} & A_{21}C_1 + A_{22}C_2 + A_{23} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Now we set:

$$\mathbf{B} = \begin{pmatrix} T_{11} & T_{12} & T_{13} & T_{14} \\ T_{21} & T_{22} & T_{23} & T_{24} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

We see that to solve this equation, we must be able to solve $\mathbf{B}_{ij} = T_{ij}$ for $i \in \{1, 2\}$, $j \in \{1, 2, 3\}$. Then, we can choose A_{13} and A_{23} to make $\mathbf{B}_{14} = T_{14}$ and $\mathbf{B}_{24} = T_{24}$ (A_{13} and A_{23} each appear in only one spot). Thus, we can solve this system of equations if the following system has a solution:

$$\begin{pmatrix} A_{11}R_{11} + A_{12}R_{21} \\ A_{11}R_{12} + A_{12}R_{22} \\ A_{11}R_{13} + A_{12}R_{23} \\ A_{21}R_{11} + A_{22}R_{21} \\ A_{21}R_{12} + A_{22}R_{22} \\ A_{21}R_{13} + A_{22}R_{23} \end{pmatrix} = \begin{pmatrix} T_{11} \\ T_{12} \\ T_{13} \\ T_{21} \\ T_{22} \\ T_{23} \end{pmatrix} \tag{29}$$

We define:

$$\mathbf{v}_1 = \begin{pmatrix} R_{11} \\ R_{12} \\ R_{13} \end{pmatrix} \quad \mathbf{v}_2 = \begin{pmatrix} R_{21} \\ R_{22} \\ R_{23} \end{pmatrix} \quad \mathbf{w}_1 = \begin{pmatrix} T_{11} \\ T_{12} \\ T_{13} \end{pmatrix} \quad \mathbf{w}_2 = \begin{pmatrix} T_{21} \\ T_{22} \\ T_{23} \end{pmatrix} \tag{30}$$

Here, $\mathbf{w}_1$ and $\mathbf{w}_2$ are made out of elements T_{ij}, and so are not in our control. We do, however, get to control the elements $\mathbf{R}$, $\mathbf{C}$, and A_{ij} to make the above system hold. $\mathbf{v}_1$ and $\mathbf{v}_2$ are made up of elements of $\mathbf{R}$ and so are values we get to control. We cannot, however, choose these vectors however we like. They form two rows of $\mathbf{R}$ (a rotation matrix) and so must be chosen of unit length, and orthogonal to each other. We now write (29) as:

$$\begin{cases} A_{11}\mathbf{v}_1 + A_{12}\mathbf{v}_2 = \mathbf{w}_1 \\ A_{21}\mathbf{v}_1 + A_{22}\mathbf{v}_2 = \mathbf{w}_2 \end{cases} \tag{31}$$

We can choose $\mathbf{v}_1$ and $\mathbf{v}_2$ by performing Gram-Schmidt[3] on the given vectors $\mathbf{w}_1$ and $\mathbf{w}_2$. Then, we choose A_{11}, A_{12}, A_{21}, and A_{22} so that (31) holds. We can always do this because the span of $\{\mathbf{v}_1, \mathbf{v}_2\}$ and $\{\mathbf{w}_1, \mathbf{w}_2\}$ will be the same.

Thus, the system (29) always has a solution, so we can always express an affine transformation, as defined in (26), as an orthogonal projection, followed by an appropriate affine transformation.

We have seen that the affine camera model generalizes both of the previous models in this section. One could ask "why not generalize more?" Certainly we could apply general non-affine transformations after orthogonal projection and derive even more general non-perspective cameras. One can do this, but if we go to any greater level of generality we loose an important property which holds for affine cameras. Namely, if we image two parallel lines in space using an affine camera, then the images of these lines will also be parallel. This can be seen by looking at our decomposition of the affine camera model above. Certainly, orthogonal projection preserves parallelism. Then, we follow that up with an affine transformation. The affine transformation can be decomposed into a translation and then a linear operation, both of which will preserve parallelism.

References

1. Subhashis Banerjee, *Camera models and affine multiple views geometry*. 6, 9
2. R. I. Hartley and A. Zisserman, *Multiple view geometry in computer vision*, Cambridge University Press, ISBN: 0521623049, 2000. 6, 8

[3] If one of $\mathbf{w}_1$ and $\mathbf{w}_2$ is a multiple of the other, choose $\mathbf{v}_1$ to be the normalized version of this vector, and set $\mathbf{v}_2$ to be any unit vector orthogonal to $\mathbf{v}_1$.