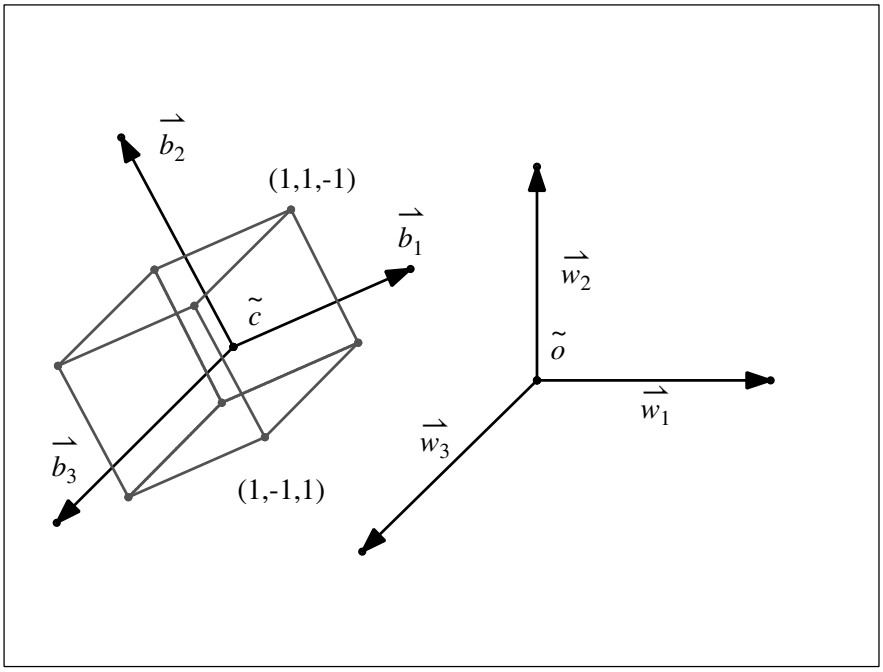
Changing Coordinate Frames

Often in computer graphics we will find it useful to describe locations with respect to a particular coordinate frame. The most common scenario is describing the vertices that make up an object with respect to that object's coordinate frame.



In the diagram shown here, one of the vertices that makes up the cube has coordinates (1,1-1) in the object coordinate system with basis $\vec{b}_1$, $\vec{b}_2$, and $\vec{b}_3$. Ultimately, we will want to know what that vertex location is in world coordinates. This can be determined by setting up a transformation matrix that transforms the world frame to the object's coordinate frame. In homogeneous coordinates, that transformation matrix is

$$A = \begin{bmatrix} \vec{b}_1 & \vec{b}_2 & \vec{b}_3 & \tilde{c} \end{bmatrix}$$

Using that transformation matrix we can compute the vertex's coordinates in world coordinates:

$$\begin{bmatrix} \vec{w}_1 & \vec{w}_2 & \vec{w}_3 & \tilde{o} \end{bmatrix} A \begin{bmatrix} 1 \\ 1 \\ -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} b_{1,1} & b_{1,2} & b_{1,3} & c_1 \\ b_{2,1} & b_{2,2} & b_{2,3} & c_2 \\ b_{3,1} & b_{3,2} & b_{3,3} & c_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ -1 \\ 1 \end{bmatrix}$$

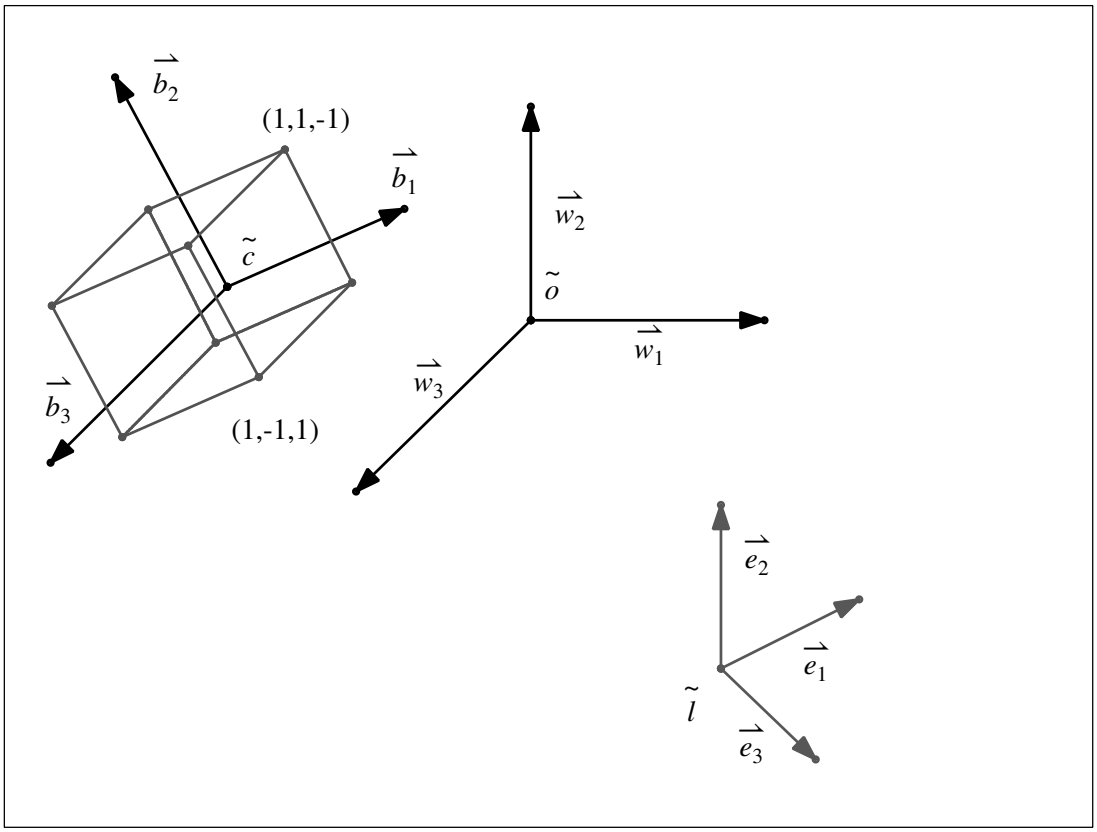
If we subsequently want to apply a transformation T to the object coordinate frame, that transformation will move the vertex to

$$\left(\begin{bmatrix} \vec{w}_1 & \vec{w}_2 & \vec{w}_3 & \tilde{o} \end{bmatrix} A \right) T \begin{bmatrix} 1 \\ 1 \\ -1 \\ 1 \end{bmatrix} = \left(\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} b_{1,1} & b_{1,2} & b_{1,3} & c_1 \\ b_{2,1} & b_{2,2} & b_{2,3} & c_2 \\ b_{3,1} & b_{3,2} & b_{3,3} & c_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \right) T \begin{bmatrix} 1 \\ 1 \\ -1 \\ 1 \end{bmatrix}$$

Here I have used grouping to help make it clear that the transformation is being applied relative to the object's coordinate frame, which is the world coordinate frame multiplied by A.

The eye coordinate frame

Up until this point we have only been concerned with placing objects in our world coordinate space. When it comes time to render the scene, we will have to insert ourselves into the imaginary scene as observers. To simulate this, we will have position ourselves at some point in the world coordinate system as observers and orient ourselves in space to look in a particular direction. As we do that, we will be establishing a coordinate frame for ourselves, called the *eye coordinate frame*.



In the diagram above, we are positioning ourselves at a location $\tilde{l}$ in the world coordinate system, and we are looking in the direction of the negative of the basis vector $\vec{e}_3$. The other two basis vectors in the eye coordinate system orient us in space: in particular, the basis vector $\vec{e}_2$ establishes our *up direction*.

As with any other coordinate frame, we can easily establish a transformation matrix that maps coordinates from our eye frame to the world frame:

$$M = \begin{bmatrix} \vec{e}_1 & \vec{e}_2 & \vec{e}_3 & \tilde{l} \end{bmatrix}$$

Since we will ultimately want to see the world from our vantage point, the more relevant mapping is a mapping from the world coordinate frame to our eye coordinate frame. We can produce this mapping by using the inverse of the matrix that maps from eye coordinates to world coordinates. For example, if a point is located at coordinates (p_1, p_2, p_3) in world coordinates, its location in eye coordinates is

$$M^{-1} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ 1 \end{bmatrix}$$

A common operation in graphics is mapping points from object coordinates to world coordinates, and subsequently to eye coordinates. For example, in the diagram above there is a point whose coordinates in an object frame are (1,1,-1). We have already seen that this point maps to

$$\begin{bmatrix} \vec{w}_1 & \vec{w}_2 & \vec{w}_3 & \tilde{o} \end{bmatrix} A \begin{bmatrix} 1 \\ 1 \\ -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} b_{1,1} & b_{1,2} & b_{1,3} & c_1 \\ b_{2,1} & b_{2,2} & b_{2,3} & c_2 \\ b_{3,1} & b_{3,2} & b_{3,3} & c_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ -1 \\ 1 \end{bmatrix}$$

in world coordinates. Expressing this point in eye coordinates gives us

$$\begin{bmatrix} \vec{e}_1 & \vec{e}_2 & \vec{e}_3 & \tilde{l} \end{bmatrix} M^{-1} A \begin{bmatrix} 1 \\ 1 \\ -1 \\ 1 \end{bmatrix} = M^{-1} \begin{bmatrix} b_{1,1} & b_{1,2} & b_{1,3} & c_1 \\ b_{2,1} & b_{2,2} & b_{2,3} & c_2 \\ b_{3,1} & b_{3,2} & b_{3,3} & c_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ -1 \\ 1 \end{bmatrix}$$

How to construct an eye coordinate frame

To build an eye coordinate frame we will need an origin and a set of three basis vectors. These typically will not be given to us, so we will have to compute these things from other information we have available. These are the things we will typically have to work with.

1. Our location in space, $\tilde{p}$.
2. A point we are looking at, $\tilde{q}$.
3. An 'up vector' $\vec{u}$ that gives us a hint as to which direction is up for us. (This orients us in space.)

From these things we can systematically construct our coordinate frame.

1. $\tilde{l}$ is just $\tilde{p}$.
2. $\vec{e}_3$ is a unit vector running in the direction opposite of the direction from $\tilde{p}$ to $\tilde{q}$.

$$\vec{e}_3 = - \frac{\tilde{q} - \tilde{p}}{|\tilde{q} - \tilde{p}|}$$

3. $\vec{e}_2$ can be computed from $\vec{u}$ by subtracting away that part of $\vec{u}$ that is in the direction of $\vec{e}_3$ and then normalizing.

$$\vec{v} = \vec{u} - \left(\vec{u} \cdot \vec{e}_3\right) \vec{e}_3$$

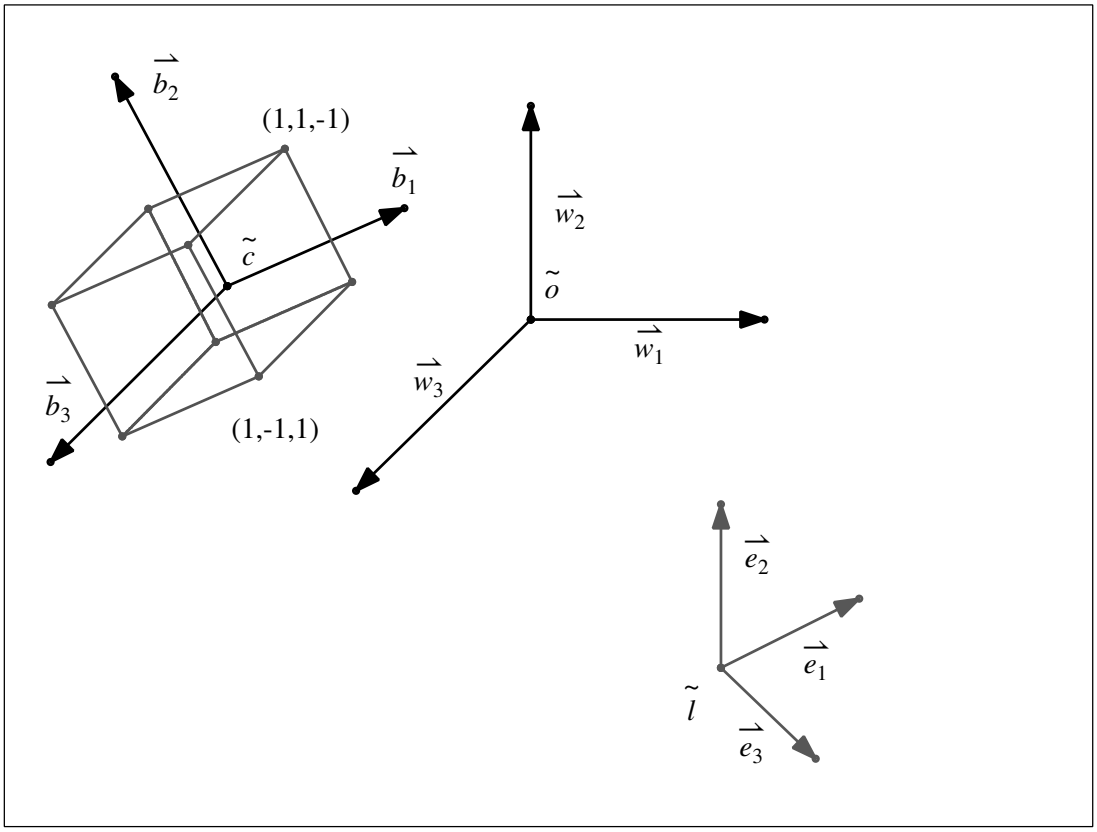
$$\vec{e}_2 = \frac{\vec{v}}{|\vec{v}|}$$

4. $\vec{e}_1$ is a vector that is mutually perpendicular to $\vec{e}_2$ and $\vec{e}_3$. We can construct such a vector by using the cross product.

$$\vec{e}_1 = \vec{e}_2 \times \vec{e}_3$$

Transformations with respect to some special frame

Once we inject ourselves into the scene as observers, we will want to start moving objects around relative to our vantage point. For example, suppose we wanted the blue cube in the diagram below to appear to move to the right relative to our vantage point.



Rightward from our perspective is in the direction of the vector $\vec{e}_1$. Unfortunately, this does not line up with the object's interpretation of rightward, which runs in the direction of $\vec{b}_1$. If we wanted the object to move, say, one unit to the right relative to its own coordinate system, we could just apply the transformation

$$T = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

which is the matrix for an affine transformation that moves points one unit to the right. In our eye coordinate system that would map one of the corners of the blue cube to

$$M^{-1} \begin{bmatrix} b_{1,1} & b_{1,2} & b_{1,3} & c_1 \\ b_{2,1} & b_{2,2} & b_{2,3} & c_2 \\ b_{3,1} & b_{3,2} & b_{3,3} & c_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} T \begin{bmatrix} 1 \\ 1 \\ -1 \\ 1 \end{bmatrix}$$

Again, this is not the effect we are trying to achieve. This moves the cube one unit in the direction of $\vec{b}_1$. We want it to move one unit in the direction of $\vec{e}_1$, which looks to us like motion to the right.

The key to doing this right is to construct a special coordinate frame. We construct a frame whose origin

matches the origin of the object's coordinate system and whose axes are aligned with our eye axes:

$$A = \begin{bmatrix} e_{1,1} & e_{1,2} & e_{1,3} & c_1 \\ e_{2,1} & e_{2,2} & e_{2,3} & c_2 \\ e_{3,1} & e_{3,2} & e_{3,3} & c_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

We will want to do our transformation relative to that frame:

$$\begin{bmatrix} e_{1,1} & e_{1,2} & e_{1,3} & c_1 \\ e_{2,1} & e_{2,2} & e_{2,3} & c_2 \\ e_{3,1} & e_{3,2} & e_{3,3} & c_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = A\ T$$

One last thing we have to manage here is the coordinates we are applying this transformation to. Our problem here is that the point coordinates

$$\begin{bmatrix} 1 \\ 1 \\ -1 \\ 1 \end{bmatrix}$$

are coordinates relative to the object's frame, not our new frame. To correct this, we have to first transform these coordinates to world coordinates:

$$\begin{bmatrix} b_{1,1} & b_{1,2} & b_{1,3} & c_1 \\ b_{2,1} & b_{2,2} & b_{2,3} & c_2 \\ b_{3,1} & b_{3,2} & b_{3,3} & c_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ -1 \\ 1 \end{bmatrix}$$

and then subsequently transform those world coordinates into our special A coordinate system:

$$A^{-1} \begin{bmatrix} b_{1,1} & b_{1,2} & b_{1,3} & c_1 \\ b_{2,1} & b_{2,2} & b_{2,3} & c_2 \\ b_{3,1} & b_{3,2} & b_{3,3} & c_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ -1 \\ 1 \end{bmatrix}$$

Now we know how to represent both the transformation and the coordinates correctly. Putting this all together we get

$$\begin{bmatrix} e_{1,1} & e_{1,2} & e_{1,3} & c_1 \\ e_{2,1} & e_{2,2} & e_{2,3} & c_2 \\ e_{3,1} & e_{3,2} & e_{3,3} & c_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \left(A^{-1} \begin{bmatrix} b_{1,1} & b_{1,2} & b_{1,3} & c_1 \\ b_{2,1} & b_{2,2} & b_{2,3} & c_2 \\ b_{3,1} & b_{3,2} & b_{3,3} & c_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ -1 \\ 1 \end{bmatrix} \right)$$

$$= A\ T\ A^{-1}\ O\ \mathbf{c}$$

This operation is called "*transforming O by T with respect to A*." You can both use a transformation like this to move an object and to move the object's coordinate frame without updating any of the object coordinates:

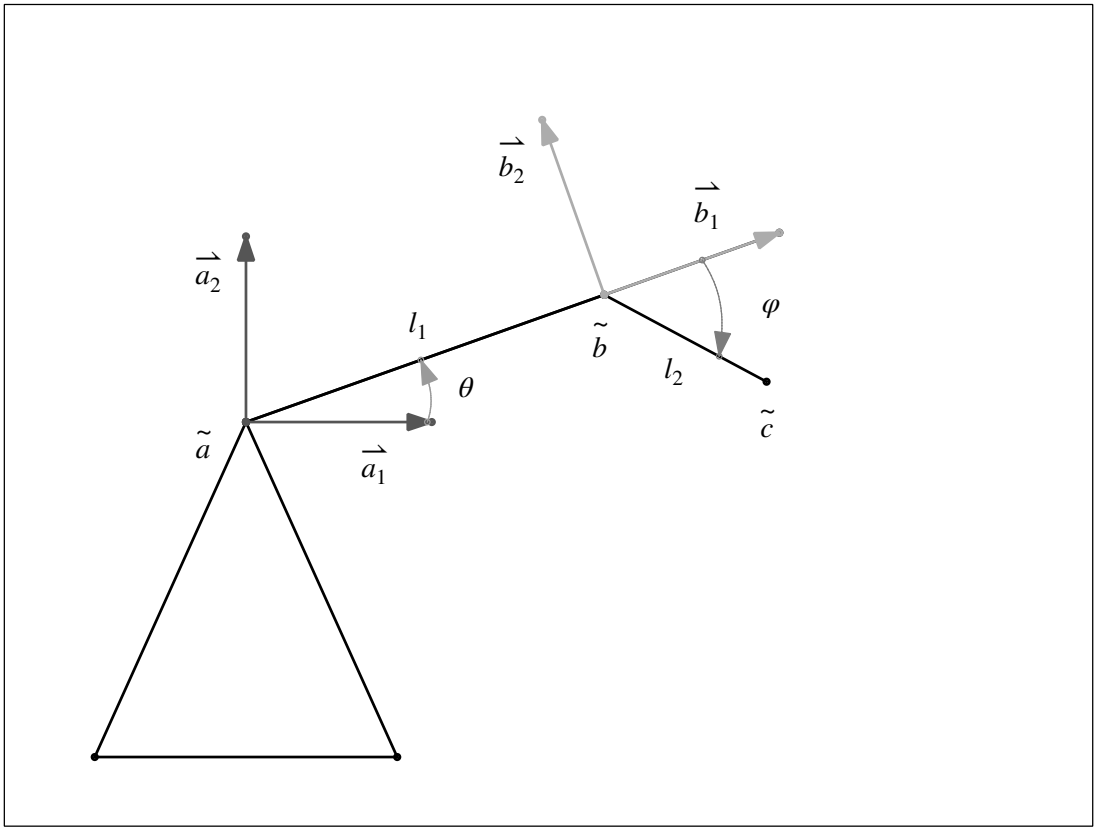
$$O \rightarrow A\ T\ A^{-1}\ O$$

One final note here. This computation carries out the desired transformation, but gives the results in world coordinates. We will then have to follow this up with one final transformation that maps world coordinates to eye coordinates. The standard way to handle this in OpenGL is to do everything in world coordinates up until the actual instant of viewing. At that instant we will apply one final transformation to everything that maps from world coordinates to eye coordinates.

Heirarchical transformations

Once we start moving beyond simple models in computer graphics we will find it convenient to construct *heirarchical models*. In a heirarchical model we try to model a complex object made up of various parts that move independently. The most obvious example of this is a graphics model of a person with limbs that can move independently and be repositioned with respect to the person's torso.

The diagram below shows a simple example of a heirarchical model in two dimensions representing something like an industrial robot with a fixed base and an articulated arm that can move relative to the base by bending a pair of joints.



The coordinate frame positioned at $\tilde{a}$ acts as the object coordinate frame for the entire model. The robot has joints at $\tilde{a}$ and $\tilde{b}$ that can be rotated independently: the diagram shows the joint at $\tilde{a}$ rotated through a positive angle θ and the joint at $\tilde{b}$ rotated through a negative angle φ.

The diagram shows a secondary frame positioned at $\tilde{b}$. We can use that frame to describe the location of the point $\tilde{c}$:

$$\begin{bmatrix} b_{1,1} & b_{1,2} & b_1 \\ b_{2,1} & b_{2,2} & b_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} l_2 \cos\varphi \\ l_2 \sin\varphi \\ 1 \end{bmatrix} = \overrightarrow{\mathbf{b}}^t \mathbf{c}$$

If we change the joint angle θ, the point $\tilde{c}$ will stay at the same coordinates relative to the coordinate frame $\overrightarrow{\mathbf{b}}^t$.

What should we do with the coordinate frame $\overrightarrow{\mathbf{b}}^t$? One approach is to try to express that coordinate frame relative to the world coordinate frame. A second, somewhat more straightforward, approach is to express the $\overrightarrow{\mathbf{b}}^t$ coordinate frame relative to the object coordinate frame $\overrightarrow{\mathbf{a}}^t$:

$$\overrightarrow{b}_1 = \begin{bmatrix} a_{1,1} & a_{1,2} & a_1 \\ a_{2,1} & a_{2,2} & a_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta \\ \sin\theta \\ 0 \end{bmatrix}$$

$$\overrightarrow{b}_2 = \begin{bmatrix} a_{1,1} & a_{1,2} & a_1 \\ a_{2,1} & a_{2,2} & a_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -\sin\theta \\ \cos\theta \\ 0 \end{bmatrix}$$

$$\tilde{b} = \begin{bmatrix} a_{1,1} & a_{1,2} & a_1 \\ a_{2,1} & a_{2,2} & a_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} l_1 \cos\theta \\ l_1 \sin\theta \\ 1 \end{bmatrix}$$

Once we can express the coordinate frame $\overrightarrow{\mathbf{b}}^t$ relative to the frame $\overrightarrow{\mathbf{a}}^t$ we can express the location of the point $\tilde{c}$ relative to the frame $\overrightarrow{\mathbf{a}}^t$ as

$$\overrightarrow{\mathbf{b}}^t \mathbf{c} = \begin{bmatrix} a_{1,1} & a_{1,2} & a_1 \\ a_{2,1} & a_{2,2} & a_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & l_1 \cos\theta \\ \sin\theta & \cos\theta & l_1 \sin\theta \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} l_2 \cos\varphi \\ l_2 \sin\varphi \\ 1 \end{bmatrix} = \overrightarrow{\mathbf{a}}^t B \mathbf{c}$$

Here B is the transformation matrix that transforms coordinates relative to $\overrightarrow{\mathbf{b}}^t$ to coordinates relative to $\overrightarrow{\mathbf{a}}^t$. If we bend the joint at $\tilde{a}$ we will be moving the point at $\tilde{c}$, although that point's coordinates in the $\overrightarrow{\mathbf{b}}^t$ frame will not change. What will change instead the B matrix that relates the $\overrightarrow{\mathbf{b}}^t$ frame to the $\overrightarrow{\mathbf{a}}^t$ frame.

Finally, we will have to relate the $\overrightarrow{\mathbf{a}}^t$ frame to the world coordinate frame. As usual, this is done by setting up an appropriate transformation matrix.

$$\overrightarrow{\mathbf{a}}^t = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} A$$

In world coordinates our point $\tilde{c}$ is now expressed

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} A B \mathbf{c}$$

This allows us to move the entire robot around by updating only the A matrix, as well as bending the joint at $\tilde{a}$

by updating only the B matrix.