1

Descartes' Discovery

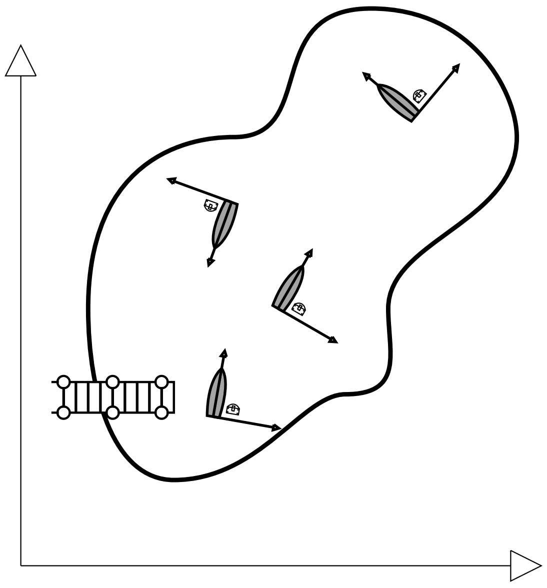


Figure 1.1.
Local and global coordinate systems: the treasure's local coordinates with respect to the boat do not change as the boat moves. However, the treasure's global coordinates, defined relative to the lake, do change as the boat moves.

There is a collection of old German tales that take place sometime in the 17th century, and they are about an alleged town called Schilda, whose inhabitants were not known for their intelligence. Here is one story [12]:

> An army was approaching Schilda and would likely conquer it. The town council, in charge of the town treasure, had to hide it from the invaders. What better way than to sink it in the nearby town lake? So the town council members board the town boat, head for the middle of the lake, and sink the treasure. The town treasurer gets out his pocket knife and cuts a deep notch in the boat's rim, right where the treasure went down. Why would he do that, the other council members wonder? "So that we will remember where we sunk the treasure, otherwise we'll never find it later!" replies the treasurer. Everyone is duly impressed at such planning genius!
>
> Eventually, the war is over and the town council embarks on the town boat again, this time to reclaim the treasure from the lake. Once out on the lake, the treasurer's plan suddenly does not seem so smart anymore. No matter where they went, the notch in the boat's rim told them they had found the treasure!

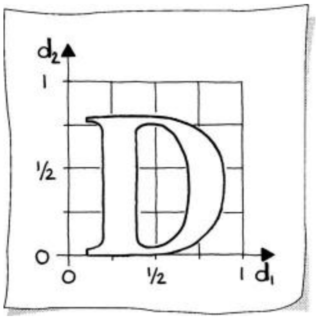
The French philosopher René Descartes (1596–1650) would have known better: he invented the theory of *coordinate systems*. The treasurer recorded the sinking of the treasure accurately by marking it on the boat. That is, he recorded the treasure's position relative to a *local* coordinate system. But by neglecting the boat's position relative to the lake, the *global* coordinate system, he lost it all! (See Figure 1.1.) The remainder of this chapter is about the interplay of local and global coordinate systems.

1.1 Local and Global Coordinates: 2D

This book is written using the LaTeX typesetting system (see [9] or [13]), which converts every page to be output to a page description language called PostScript (see [1]). It tells a laser printer where to position all the characters and symbols that go on a particular page. For the first page of this chapter, there is a PostScript command that positions the letter **D** in the chapter heading.

In order to do this, one needs a two-dimensional, or 2D, coordinate system. Its origin is simply the lower left corner of the page, and the x- and y-axes are formed by the horizontal and vertical paper edges meeting there. Once we are given this coordinate system, we can position objects in it, such as our letter **D**.

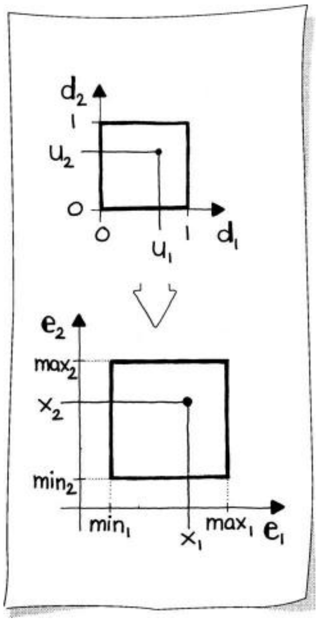
The **D**, on the other hand, was designed by font designers who obviously did not know about its position on this page or of its actual size. They used their own coordinate system, and in it, the letter **D** is described by a set of points, each having coordinates relative to **D**'s coordinate system, as shown in Sketch 1.1.



Sketch 1.1.
A local coordinate system.

We call this system a *local coordinate system*, as opposed to the *global coordinate system*, which is used for the whole page. Positioning letters on a page thus requires mastering the interplay of the global and local systems.

Following Sketch 1.2, let's make things more formal: Let (x_1, x_2) be coordinates in a global coordinate system, called the $[\mathbf{e}_1, \mathbf{e}_2]$-system. The boldface notation will be explained in the next chapter. You may be used to calling coordinates (x, y); however, the (x_1, x_2) notation will streamline the material in this book, and it also makes writing programs easier. Let (u_1, u_2) be coordinates in a local system called the $[\mathbf{d}_1, \mathbf{d}_2]$-system. Let an object in the local system be enclosed by a box with lower left corner $(0, 0)$ and upper right corner $(1, 1)$. This means that the object "lives" in the *unit square* of the local system, i.e., a square of edge length one, and with its lower left corner at the origin. Restricting ourselves to the unit square for the local system makes this first chapter easy—we will later relax this restriction.



Sketch 1.2.
Global and local systems.

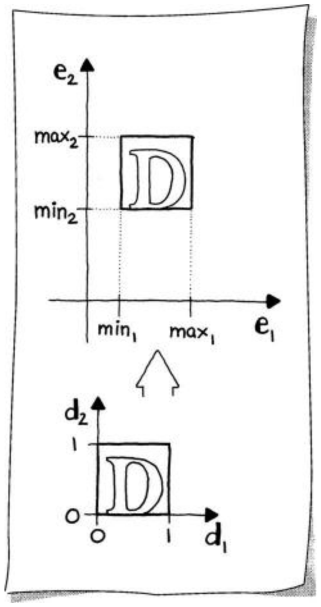
We wish to position our object into the global system so that it fits into a box with lower left corner $(\text{min}_1, \text{min}_2)$ and upper right corner $(\text{max}_1, \text{max}_2)$ called the *target box* (drawn with heavy lines in Sketch 1.2). This is accomplished by assigning to coordinates (u_1, u_2) in the local system the corresponding target coordinates (x_1, x_2) in the global system. This correspondence is characterized by preserving each coordinate value with respect to their extents. The local coordinates are also known as *parameters*. In terms of formulas, these parameters are written as quotients,

$$\frac{u_1 - 0}{1 - 0} = \frac{x_1 - \text{min}_1}{\text{max}_1 - \text{min}_1}$$
$$\frac{u_2 - 0}{1 - 0} = \frac{x_2 - \text{min}_2}{\text{max}_2 - \text{min}_2}.$$

Thus, the corresponding formulas for x_1 and x_2 are quite simple:

$$x_1 = (1 - u_1)\text{min}_1 + u_1\text{max}_1, \tag{1.1}$$
$$x_2 = (1 - u_2)\text{min}_2 + u_2\text{max}_2. \tag{1.2}$$

We say that the coordinates (u_1, u_2) are *mapped* to the coordinates (x_1, x_2). Sketch 1.3 illustrates how the letter **D** is mapped. This concept of a parameter is reintroduced in Section 2.5.



Sketch 1.3.
Local and global $\mathbf{D}$.

Let's check that this actually works: the coordinates $(u_1, u_2) = (0, 0)$ in the local system must go to the coordinates $(x_1, x_2) = (\mathrm{min}_1, \mathrm{min}_2)$ in the global system. We obtain

$$x_1 = (1 - 0) \cdot \mathrm{min}_1 + 0 \cdot \mathrm{max}_1 = \mathrm{min}_1,$$
$$x_2 = (1 - 0) \cdot \mathrm{min}_2 + 0 \cdot \mathrm{max}_2 = \mathrm{min}_2.$$

Similarly, the coordinates $(u_1, u_2) = (1, 0)$ in the local system must go to the coordinates $(x_1, x_2) = (\mathrm{max}_1, \mathrm{min}_2)$ in the global system. We obtain

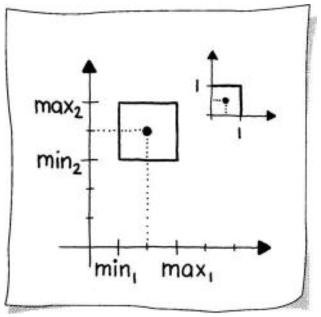
$$x_1 = (1 - 1) \cdot \mathrm{min}_1 + 1 \cdot \mathrm{max}_1 = \mathrm{max}_1,$$
$$x_2 = (1 - 0) \cdot \mathrm{min}_2 + 0 \cdot \mathrm{max}_2 = \mathrm{min}_2.$$

Example 1.1

Let the target box be given by

$$(\mathrm{min}_1, \mathrm{min}_2) = (1, 3) \quad \text{and} \quad (\mathrm{max}_1, \mathrm{max}_2) = (3, 5),$$

see Sketch 1.4. The coordinates $(1/2, 1/2)$ can be thought of as the "midpoint" of the local unit square. Let's look at the result of the mapping:



Sketch 1.4.
Map local unit square to a target box.

$$x_1 = (1 - \frac{1}{2}) \cdot 1 + \frac{1}{2} \cdot 3 = 2,$$
$$x_2 = (1 - \frac{1}{2}) \cdot 3 + \frac{1}{2} \cdot 5 = 4.$$

This is the "midpoint" of the target box. You see here how the geometry in the unit square is replicated in the target box.

A different way of writing (1.1) and (1.2) is as follows: Define $\Delta_1 = \mathrm{max}_1 - \mathrm{min}_1$ and $\Delta_2 = \mathrm{max}_2 - \mathrm{min}_2$. Now we have

$$x_1 = \mathrm{min}_1 + u_1\Delta_1, \tag{1.3}$$
$$x_2 = \mathrm{min}_2 + u_2\Delta_2. \tag{1.4}$$

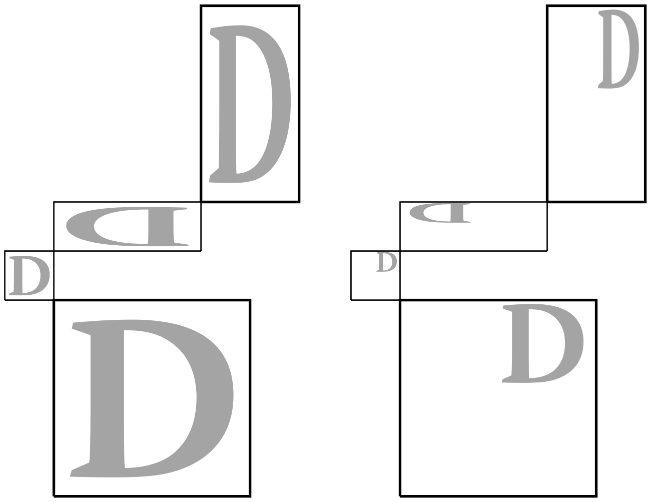


Figure 1.2.

Target boxes: the letter $\mathbf{D}$ is mapped several times. Left: centered in the unit square. Right: not centered.

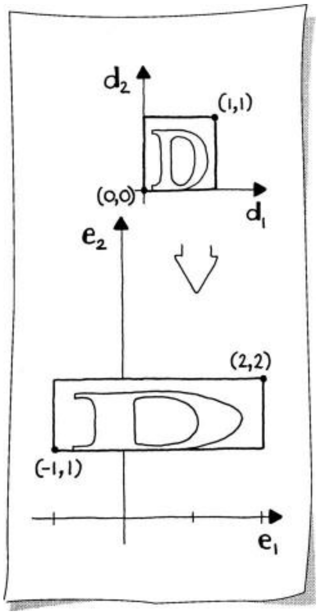
A note of caution: if the target box is not a square, then the object from the local system will be distorted. We see this in the following example, illustrated by Sketch 1.5. The target box is given by

$$(\min_1, \min_2) = (-1, 1) \quad \text{and} \quad (\max_1, \max_2) = (2, 2).$$

You can see how the local object is stretched in the $\mathbf{e}_1$-direction by being put into the global system. Check for yourself that the corners of the unit square (local) still get mapped to the corners of the target box (global).

In general, if $\Delta_1 > 1$, then the object will be stretched in the $\mathbf{e}_1$-direction, and it will be shrunk if $0 < \Delta_1 < 1$. The case of $\max_1$ smaller than $\min_1$ is not often encountered: it would result in a reversal of the object in the $\mathbf{e}_1$-direction. The same applies, of course, to the $\mathbf{e}_2$-direction if $\max_2$ is smaller than $\min_2$. An example of several boxes containing the letter $\mathbf{D}$ is shown in Figure 1.2. Just for fun, we have included one target box with $\max_1$ smaller than $\min_1$!

Another characterization of the change of shape of the object may be made by looking at the change in *aspect ratio*, which is the ratio of the width to the height, Δ_1/Δ_2, for the target box. This is also written as $\Delta_1 : \Delta_2$.The aspect ratio in the local system is one. Revisiting



Sketch 1.5.
A distortion.

Example 1.1, the aspect ratio of the target box is one, therefore there is no distortion of the letter **D**, although it is stretched uniformly in both coordinates. In Sketch 1.5, a target box is given that has aspect ratio 3, therefore the letter **D** is distorted.

Aspect ratios are encountered many times in everyday life. Televisions and computer screens have recently changed from nearly square 4 : 3 to 16 : 9. Sketch 1.5 illustrates the kind of distortion that occurs when an old format program is stretched to fill a new format screen. (Normally a better solution is to not stretch the image and allow for vertical black bars on either side of the image.) All international (ISO A series) paper, regardless of size, has an aspect ratio of $1 : \sqrt{2}$. Golden rectangles, formed based on the golden ratio $\phi = (1 + \sqrt{5})/2$ with an aspect ratio of $1 : \phi$, provide a pleasing and functional shape, and found their way into art and architecture. Credit cards have an aspect ratio of 8 : 5, but to fit into your wallet and card readers the size is important as well.

This principle, by the way, acts strictly on a "don't need to know" basis: we do not need to know the relationship between the local and global systems. In many cases (as in the typesetting example), there actually isn't a known correspondence at the time the object in the local system is created. Of course, one must know where the actual object is located in the local unit square. If it is not nicely centered, we might have the situation shown in Figure 1.2 (right).

You experience this "unit square to target box" mapping whenever you use a computer. When you open a window, you might want to view a particular image in it. The image is stored in a local coordinate system; if it is stored with extents $(0, 0)$ and $(1, 1)$, then it utilizes *normalized coordinates*. The target box is now given by the extents of your window, which are given in terms of *screen coordinates* and the image is mapped to it using (1.1) and (1.2). Screen coordinates are typically given in terms of *pixels*;[1] a typical computer screen would have about 1440×900 pixels, which has an aspect ratio of 8 : 5 or 1.6.

1.2 Going from Global to Local

When discussing global and local systems in 2D, we used a target box to position (and possibly distort) the unit square in a local $[\mathbf{d}_1, \mathbf{d}_2]$-system. For given coordinates (u_1, u_2), we could find coordinates (x_1, x_2) in the global system using (1.1) and (1.2), or (1.3) and (1.4).

[1]The term is short for "picture element."

How about the inverse problem: given coordinates (x_1, x_2) in the global system, what are its local (u_1, u_2) coordinates? The answer is relatively easy: compute u_1 from (1.3), and u_2 from (1.4), resulting in

$$u_1 = \frac{x_1 - \text{min}_1}{\Delta_1}, \tag{1.5}$$

$$u_2 = \frac{x_2 - \text{min}_2}{\Delta_2}. \tag{1.6}$$

Applications for this process arise any time you use a mouse to communicate with a computer. Suppose several icons are displayed in a window. When you click on one of them, how does your computer actually know which one? The answer: it uses Equations (1.5) and (1.6) to determine its position.

Example 1.2

Let a window on a computer screen have screen coordinates

$$(\text{min}_1, \text{min}_2) = (120, 300) \quad \text{and} \quad (\text{max}_1, \text{max}_2) = (600, 820).$$

The window is filled with 21 icons, arranged in a 7×3 pattern (see Figure 1.3). A mouse click returns screen coordinates (200, 709). Which icon was clicked? The computations that take place are as follows:

$$u_1 = \frac{200 - 120}{480} \approx 0.17,$$
$$u_2 = \frac{709 - 300}{520} \approx 0.79,$$

according to (1.5) and (1.6).

The u_1-partition of normalized coordinates is

$$0, \quad 0.33, \quad 0.67, \quad 1.$$

The value 0.17 for u_1 is between 0.0 and 0.33, so an icon in the first column was picked. The u_2-partition of normalized coordinates is

$$0, \quad 0.14, \quad 0.29, \quad 0.43, \quad 0.57, \quad 0.71, \quad 0.86, \quad 1.$$

The value 0.79 for u_2 is between 0.71 and 0.86, so the "Display" icon in the second row of the first column was picked.

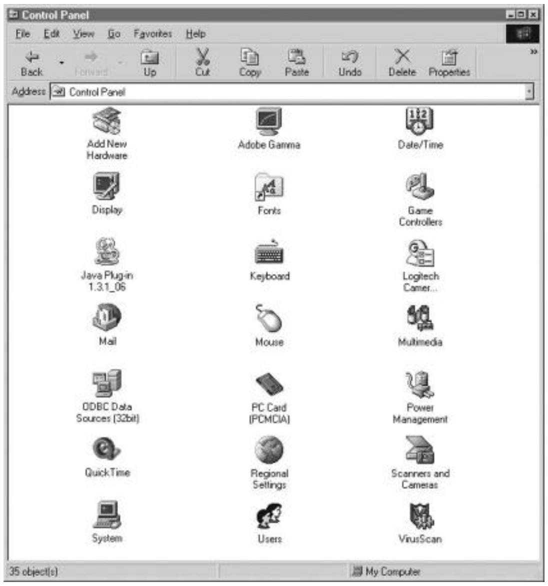
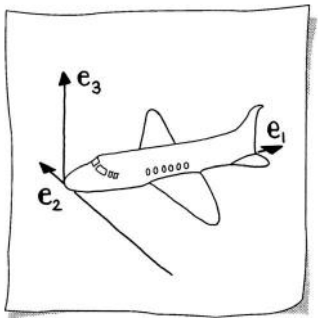


Figure 1.3.
Selecting an icon: global to local coordinates.

1.3 Local and Global Coordinates: 3D



Sketch 1.6.
Airplane coordinates.

These days, almost all engineering objects are designed using a *Computer-Aided Design* (*CAD*) system. Every object is defined in a coordinate system, and usually many individual objects need to be integrated into one coordinate system. Take designing a large commercial airplane, for example. It is defined in a three-dimensional (or 3D) coordinate system with its origin at the frontmost part of the plane, the $\mathbf{e}_1$-axis pointing toward the rear, the $\mathbf{e}_2$-axis pointing to the right (that is, if you're sitting in the plane), and the $\mathbf{e}_3$-axis is pointing upward. See Sketch 1.6.

Before the plane is built, it undergoes intense computer simulation in order to find its optimal shape. As an example, consider the engines: these may vary in size, and their exact locations under the wings need to be specified. An engine is defined in a local coordinate system, and it is then moved to its proper location. This process will have to be repeated for all engines. Another example would be the seats in the plane: the manufacturer would design just one—then multiple copies of it are put at the right locations in the plane's design.

Following Sketch 1.7, and making things more formal again, we are given a local 3D coordinate system, called the $[\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3]$-system, with coordinates (u_1, u_2, u_3). We assume that the object under consideration is located inside the *unit cube*, i.e., all of its defining points satisfy

$$0 \le u_1, u_2, u_3 \le 1.$$

This cube is to be mapped onto a *3D target box* in the global $[\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3]$-system. Let the target box be given by its lower corner $(\text{min}_1, \text{min}_2, \text{min}_3)$ and its upper corner $(\text{max}_1, \text{max}_2, \text{max}_3)$. How do we map coordinates (u_1, u_2, u_3) from the local unit cube into the corresponding target coordinates (x_1, x_2, x_3) in the target box? Exactly as in the 2D case, with just one more equation:

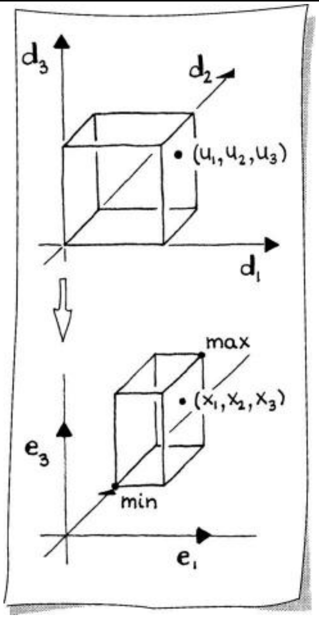
$$x_1 = (1 - u_1)\text{min}_1 + u_1\text{max}_1, \tag{1.7}$$
$$x_2 = (1 - u_2)\text{min}_2 + u_2\text{max}_2, \tag{1.8}$$
$$x_3 = (1 - u_3)\text{min}_3 + u_3\text{max}_3. \tag{1.9}$$

As an easy exercise, check that the corners of the unit cube are mapped to the corners of the target box.

The analog to (1.3) and (1.4) is given by the rather obvious

$$x_1 = \text{min}_1 + u_1\Delta_1, \tag{1.10}$$
$$x_2 = \text{min}_2 + u_2\Delta_2, \tag{1.11}$$
$$x_3 = \text{min}_3 + u_3\Delta_3. \tag{1.12}$$

As in the 2D case, if the target box is not a cube, object distortions will result—this may be desired or not.



Sketch 1.7.
Global and local 3D systems.

1.4 Stepping Outside the Box

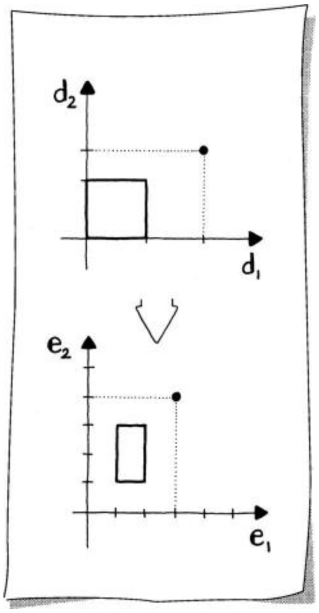
We have restricted all objects to be within the unit square or cube; as a consequence, their images were inside the respective target boxes. This notion helps with an initial understanding, but it is not at all essential. Let's look at a 2D example, with the target box given by

$$(\text{min}_1, \text{min}_2) = (1, 1) \quad \text{and} \quad (\text{max}_1, \text{max}_2) = (2, 3),$$

and illustrated in Sketch 1.8.

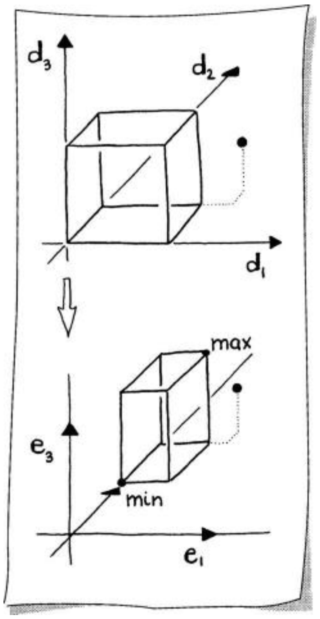
The coordinates $(u_1, u_2) = (2, 3/2)$ are not inside the $[\mathbf{d}_1, \mathbf{d}_2]$-system unit square. Yet we can map it using (1.1) and (1.2):

$$x_1 = -\text{min}_1 + 2\text{max}_1 = 3,$$
$$x_2 = -\frac{1}{2}\text{min}_2 + \frac{3}{2}\text{max}_2 = 4.$$



Sketch 1.8.
A 2D coordinate outside the target box.

Since the initial coordinates (u_1, u_2) were not inside the unit square, the mapped coordinates (x_1, x_2) are not inside the target box. The notion of mapping a square to a target box is a useful concept for mentally visualizing what is happening—but it is not actually a restriction to the coordinates that we can map!



Sketch 1.9.

A 3D coordinate outside the 3D target box.

Example 1.3

Without much belaboring, it is clear the same holds for 3D. An example should suffice: the target box is given by

$$(\mathrm{min}_1, \mathrm{min}_2, \mathrm{min}_3) = (0, 1, 0) \quad \text{and}$$
$$(\mathrm{max}_1, \mathrm{max}_2, \mathrm{max}_3) = (0.5, 2, 1),$$

and we want to map the coordinates

$$(u_1, u_2, u_3) = (1.5, 1.5, 0.5).$$

The result, illustrated by Sketch 1.9, is computed using (1.7)–(1.9): it is

$$(x_1, x_2, x_3) = (0.75, 2.5, 0.5).$$

1.5 Application: Creating Coordinates

Suppose you have an interesting real object, like a model of a cat. A friend of yours in Hollywood would like to use this cat in her latest hi-tech animated movie. Such movies use only mathematical descriptions of objects—everything must have coordinates! You might recall the movie *Toy Story*. It is a computer-animated movie, meaning that the characters and objects in every scene have a mathematical representation.

So how do you give your cat model coordinates? This is done with a *CMM*, or *coordinate measuring machine;* see Figure 1.4. The CMM is essentially an arm that is able to record the position of its tip by keeping track of the angles of its joints.

Your cat model is placed on a table and somehow fixed so it does not move during digitizing. You let the CMM's arm touch three points on the table; they will be converted to the origin and the $\mathbf{e}_1$- and $\mathbf{e}_2$-coordinate axes of a 3D coordinate system. The $\mathbf{e}_3$-axis (vertical to the table) is computed automatically.[2] Now when you touch your

[2]Just how we convert the three points on the table to the three axes is covered in Section 11.8.

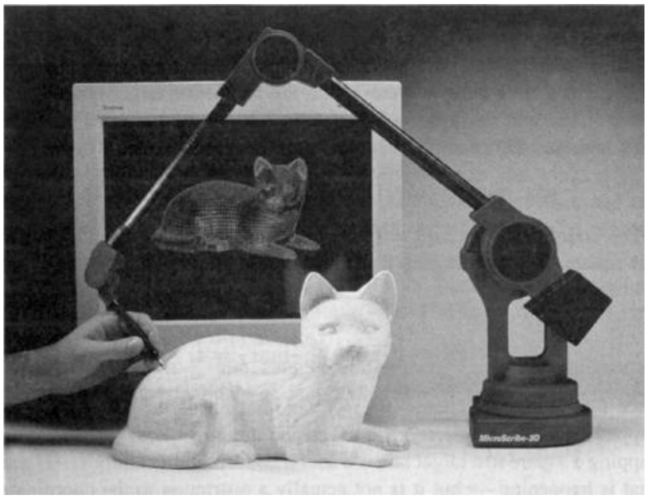


Figure 1.4.

Creating coordinates: a cat is turned into math. (Microscribe-3D from Immersion Corporation, http://www.immersion.com.)

cat model with the tip of the CMM's arm, it will associate three coordinates with that position and record them. You repeat this for several hundred points, and you have your cat in the box! This process is called *digitizing*. In the end, the cat has been "discretized," or turned into a finite number of coordinate triples. This set of points is called a *point cloud*.

Someone else will now have to build a mathematical model of your cat.[3] Next, the mathematical model will have to be put into scenes of the movie—but all that's needed for that are 3D coordinate transformations! (See Chapters 9 and 10.)

- unit square
- 2D and 3D local coordinates
- 2D and 3D global coordinates
- coordinate transformation
- parameter
- aspect ratio
- normalized coordinates
- digitizing
- point cloud

[3]This type of work is called *geometric modeling* or *computer-aided geometric design*, see [7] and Chapter 20.

1.6 Exercises

1. Let the coordinates of triangle vertices in the local $[\mathbf{d}_1, \mathbf{d}_2]$-system unit square be given by

$$(u_1, u_2) = (0.1, 0.1), \quad (v_1, v_2) = (0.9, 0.2), \quad (w_1, w_2) = (0.4, 0.7).$$

   (a) If the $[\mathbf{d}_1, \mathbf{d}_2]$-system unit square is mapped to the target box with

$$(\text{min}_1, \text{min}_2) = (1, 2) \quad \text{and} \quad (\text{max}_1, \text{max}_2) = (3, 3),$$

   where are the coordinates of the triangle vertices mapped?

   (b) What local coordinates correspond to $(x_1, x_2) = (2, 2)$ in the $[\mathbf{e}_1, \mathbf{e}_2]$-system?

2. Given local coordinates $(2, 2)$ and $(-1, -1)$, find the global coordinates with respect to the target box with

$$(\text{min}_1, \text{min}_2) = (1, 1) \quad \text{and} \quad (\text{max}_1, \text{max}_2) = (7, 3).$$

   Make a sketch of the local and global systems. Connect the coordinates in each system with a line and compare.

3. Let the $[\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3]$-system unit cube be mapped to the 3D target box with

$$(\text{min}_1, \text{min}_2, \text{min}_3) = (1, 1, 1) \quad \text{and} \quad (\Delta_1, \Delta_2, \Delta_3) = (1, 2, 4).$$

   Where will the coordinates $(u_1, u_2, u_3) = (0.5, 0, 0.7)$ be mapped?

4. Let the coordinates of triangle vertices in the local $[\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3]$-system unit square be given by

$$(u_1, u_2, u_3) = (0.1, 0.1, 0), \quad (v_1, v_2, v_3) = (0.9, 0.2, 1),$$
$$(w_1, w_2, w_3) = (0.4, 0.7, 0).$$

   If the $[\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3]$-system unit square is mapped to the target box with

$$(\text{min}_1, \text{min}_2) = (1, 2, 4) \quad \text{and} \quad (\text{max}_1, \text{max}_2) = (3, 3, 8),$$

   where are the coordinates of the triangle vertices mapped? *Hint: See Exercise 1a.*

5. Suppose we are given a global frame defined by $(\text{min}_1, \text{min}_2, \text{min}_3) = (0, 0, 3)$ and $(\text{max}_1, \text{max}_2, \text{max}_3) = (4, 4, 4)$. For the coordinates $(1, 1, 3)$ and $(0, 0, 0)$ in this frame, what are the corresponding coordinates in the $[\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3]$-system?

6. Assume you have an image in a local frame of 20 mm^2. If you enlarge the frame and the image such that the new frame covers 40 mm^2, by how much does the image size change?

7. Suppose that local frame coordinates $(v_1, v_2) = (1/2, 1)$ are mapped to global frame coordinates $(5, 2)$ and similarly, $(w_1, w_2) = (1, 1)$ are mapped to $(8, 8)$. In the local frame, $(u_1, u_2) = (3/4, 1/2)$ lies at the midpoint of two local coordinate sets. What are its coordinates in the global frame?

8. The size of a TV set is specified by its monitor's diagonal. In order to determine the width and height of this TV, we must know the relevant aspect ratio. What are the width and height dimensions of a $32''$ standard TV with aspect ratio $4 : 3$? What are the dimensions of a $32''$ HD TV with aspect ratio $16 : 9$? (This is an easy one to find on the web, but the point is to calculate it yourself!)

9. Suppose we are given coordinates $(1/2, 1/2)$ in the $[\mathbf{d}_1, \mathbf{d}_2]$-system. What are the corresponding coordinates in the global frame with aspect ratio $4 : 3$, $(\text{min}_1, \text{min}_2) = (0, 0)$, and $\Delta_2 = 2$?

10. In some implementations of the computer graphics viewing pipeline, *normalized device coordinates*, or NDC, are defined as the cube with extents $(-1, -1, -1)$ and $(1, 1, 1)$. The next step in the pipeline maps the (u_1, u_2) coordinates from NDC (u_3 is ignored) to the *viewport*, the area of the screen where the image will appear. Give equations for (x_1, x_2) in the viewport defined by extents $(\text{min}_1, \text{min}_2)$ and $(\text{max}_1, \text{max}_2)$ that correspond to (u_1, u_2) in NDC.

This page intentionally left blank