1 The Basics of Matchmoving

Anytime a computer-generated (CG) element needs to be placed into a live-action sequence or vice versa, a matchmove is required. But what exactly is matchmoving?

Matchmoving is the process of matching CG elements into live-action footage. As a result, it's a crucial part of many visual effects shots. Despite its importance, it is completely invisible in the final shot—that is, if it's done right.

In this chapter, I explain the key steps involved in a matchmove and how matchmovers work with the rest of the visual effects team. I've also included a tutorial that is designed to help you get comfortable working with cameras and perspective.

Chapter Contents

A Typical Matchmove
From 3D to 2D and Back Again
The Matchmove Process
Matchmoving in the Production Pipeline
Perspective Matching Tutorial

A Typical Matchmove

In order to better understand what matchmovers do, let's consider a typical visual effects shot. The director has called for a CG creature to crash out of window of a building and run across the street and into an alley. Because the monster will need to interact with the window, the visual effects supervisor decides that in addition to the monster, the window-shattering effect should also be done in the computer.

On the day of shooting, the director makes artistic decisions as to how he wants to shoot the scene and eventually decides on a camera position that he likes. There is an opening on the building where the window should be, although the panes of glass are missing. The director and the cameraman practice the camera move a few times and watch the video playback to see how it looks. When they are filming, they move the camera as though it were following the monster crashing through the window and running across the street, even though the monster isn't there. Extras react to the imaginary beast, and props around the window are rigged with monofilament string (fishing line) to be pulled down on cue as though they were knocked over. Once the director is happy with the shot, the film is sent off to be digitized and then given to the visual effects artists to add the monster.

When the visual effects studio receives the digitized sequence (known as a *plate*), they decide that they will need an animator to animate the creature and a technical director (TD) to do the glass-shattering effect. And, of course, they'll need a matchmover to matchmove the plate.

The visual effects artists' goals are to make their 3D elements look as realistic as the scene that was filmed. The animator will need to make the creature move as though it were really crashing through a window, and the TD will need to make the window shatter like a real window. The matchmover needs to figure out where the camera was and how it was moving when the scene was filmed.

Matchmovers play an important role in this case, because in order for the creature and window to appear matched realistically with the scene, they need to make sure that the CG objects are "filmed" the same way with their CG camera as the real set was filmed with the real camera. Consider the window that needs to shatter—if the perspective of the window doesn't match the perspective in the plate, it will look out of place. Furthermore, if the real camera moves to the left and the CG window stays put, everyone will know it's a fake.

In our example effects shot, the visual effects supervisor measures key items on the set. For example, she measures the size of the opening of the window as well as its height off the ground. She measures the distance across the street and the size of the opening to the alleyway. She draws a rough picture of the set and makes notes about positions of certain props and lights that might be useful to know. She also

measures how high the camera is off the ground, what lens is used, and how far it is from the window.

Typically the animator, TD, and matchmover all start at the same time. Since there are some measurements of the set, the animator knows how high the creature needs to jump to get through the window and how far to make it run across the street. The TD knows the size of the window that needs to shatter and how high it is off the ground. This is enough information to allow them to start setting up their scenes.

While they're doing that, the matchmover starts by first examining the footage to get an idea of how the camera was moving during the shot. He brings the footage into his matchmoving software and begins to track the 2D features in the scene as they move around the screen. 2D tracking usually involves identifying things in the scene that don't move (such as the corner of a building) and then letting the software follow that feature as the footage plays.

Once the matchmover has tracked a number of 2D tracks, the software analyzes these tracks and computes the position of the camera in relation to the items in the scene. At the end of this process, the matchmover exports a scene to his 3D-animation package that includes an animated camera and the 3D positions for all of the features he tracked.

Once the matchmover is happy with the camera he has generated, he goes about fitting that camera into a CG scene that is the same size as the one the animator and TD are using. When he's finished, he is able to look through his CG camera at the CG window and creature, and they appear in the right perspective and scale when compared to the original live-action footage.

Finally, he saves the scene with the matchmove camera in it. The animator and TD both use this camera to render their animations. If they've placed their window and creature in the right place in the environment, then they don't need to worry about whether the perspective matches or whether movement of the camera is the same. As long as they've rendered it using the matchmove camera, it will appear matched into the footage (Figure 1.1).

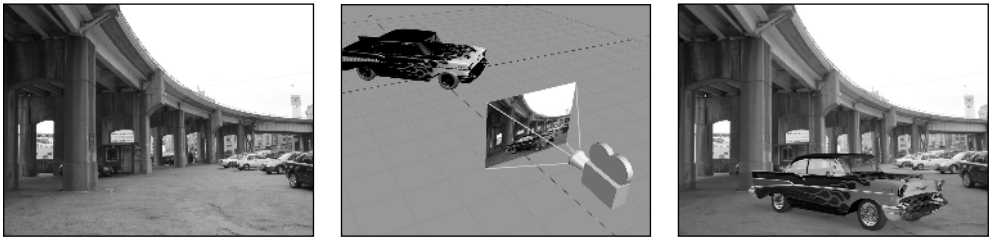

Figure 1.1 In the original "clean" plate (left) we see the image before the CG elements are placed in the shot. In the 3D-animation software, a CG camera is positioned in the environment (middle) in such a way that when the CG vehicle is viewed through it (right), the perspective matches that of the plate.

From 3D to 2D and Back Again

No discussion of matchmoving would be complete without discussing cameras. For matchmovers, it is important to understand how real-world cameras work and how they make the images we see on the screen. I'll discuss cameras more in-depth in Chapter 6, "Cameras," but for now there are a few basic concepts about real-world cameras that are important to know before getting started.

When a real camera films a scene, it is basically doing one thing: capturing the three-dimensional world as a two-dimensional image. That is, it gathers light from the 3D world around us and records it in a 2D form, such as a piece of film or a digital image. Let's consider for a moment exactly how this happens.

The light from the scene passes through the camera lens and is focused onto film that rests in the *film gate* on the back of the camera's inner chamber. The shutter closes, the film advances, the shutter opens again, and the process repeats. In the case of digital cameras, the film and film gate are replaced by a *CCD* (Charge-Coupled Device) that electronically captures the light information and records it to some sort of memory device.

The cameras in a 3D-animation program are based on real cameras, but they are represented in a slightly different manner. 3D-animation cameras represent a mathematically perfect model of the optics and construction of a real camera. Like real-world cameras, they have a focal length and a *film back* (the equivalent of a film gate). But rather than capturing light from the real world, they are simply capturing information of the synthetic, computer-generated environment in which they have been placed.

Whether you're dealing with exposed film or a 3D render, the resulting image is a *projection*. That is, the three-dimensional scene is flattened out into a two-dimensional representation of that scene. We have become so accustomed to these flattened images that we hardly notice them anymore, but every time we watch TV or a film, we are watching a flat recording of a three-dimensional scene (Figure 1.2).

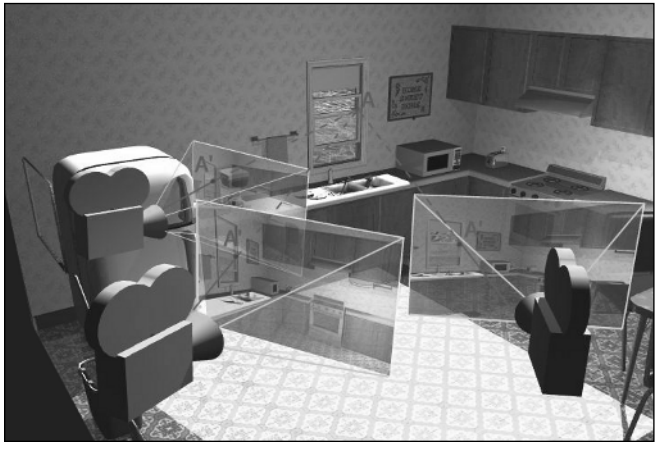

Figure 1.2 When a camera captures an image of the three-dimensional world, it is really capturing the light from the scene that is projected across its image plane.

The Matchmoving Process

So if a camera's purpose is to take the three-dimensional world and make a two-dimensional image, a matchmover's job is the exact opposite. A matchmover must take a two-dimensional image and create a three-dimensional world. The portal between these two halves is the camera. If information about the camera can be reconstructed, it will go a long way toward figuring out how the 3D environment must have been at the time of filming. This information—the 3D environment and the camera—is what a matchmover will ultimately deliver to the animators to work with.

A matchmover's workflow (Figure 1.3) generally follows the same pattern for each shot, although there are a variety of ways to complete each task. The figure shows a typical matchmove *pipeline*. In the following sections, we'll examine each of these steps more closely.

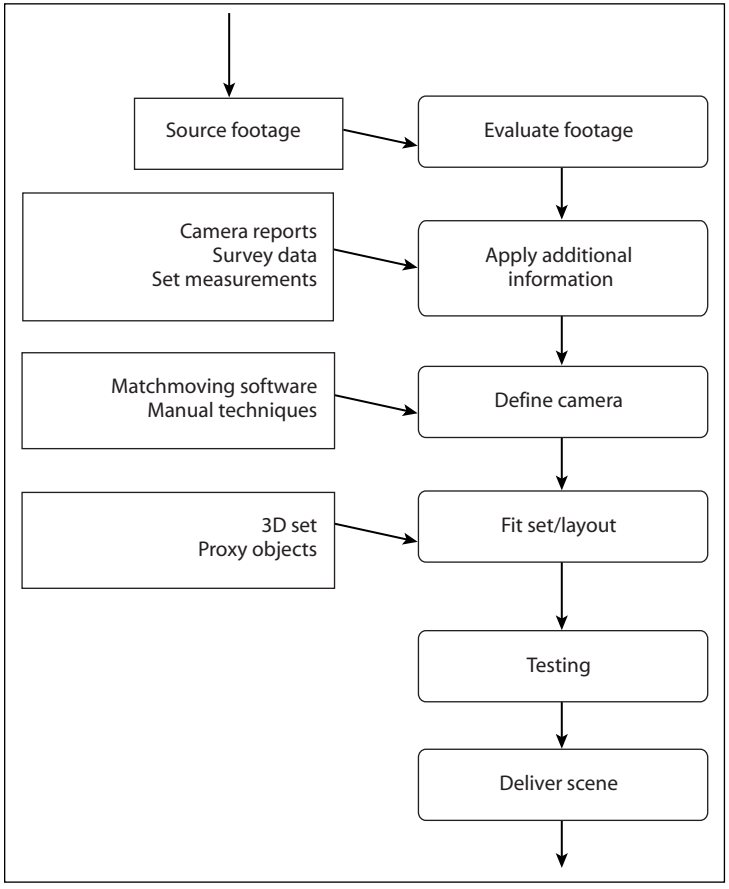

Figure 1.3 Matchmovers receive data from various sources, use that information to solve and set up the scene, and then deliver a scene with a matchmoved camera.

Evaluate the Footage

This is perhaps the most important step in the matchmoving process, and unfortunately it is often overlooked. There are many ways to solve a matchmove, and careful scrutiny of the plate can help decide the appropriate tool to use, what pitfalls to watch out for, and how long it might take. This last item is particularly important, since clients and supervisors often put it at the top of their list of questions.

One reason the evaluation process is so often glossed over is that many of the things that determine the difficulty of matchmove require some experience to judge. Some typical questions asked during the evaluation of a shot could include:

- What does the camera seem to be doing? Is it moving, and if so, how? Is it locked off or panning? How fast is it moving?
- What is visible in the shot? Are there tracking markers? Is anything blocking the markers?
- What format is the plate? Was it shot on film? DV? HD? Is there excessive compression, grain, or noise on the images?
- What needs to be placed in the shot? How accurate does it have to be?
- Who will be using the matchmove, and how will they be using it?

Of course, these only scratch the surface, but the more questions you ask, the more you will know what you need to do. The tutorials in this book are designed to help you learn what these key questions are and how to deal with their implications. A more thorough list of questions are included in Appendix A. The Evaluation Checklist there can be used as a guideline to help determine the difficulty of a matchmove.

Applying Information

As I've said, solving a matchmove can be like solving a puzzle (it's no coincidence that it's referred to as "solving" a matchmove). And as such, the more information there is, the easier it should be to achieve good results.

The amount of information given to matchmovers can run the gamut. There might only be an image sequence and nothing else, or perhaps someone was allowed on set to record all the camera information and take measurements. Usually it's somewhere in between. But the good news is that a surprisingly small amount of data can go a long way toward solving the matchmove.

The following are typical data a matchmover might include:

Camera information Such as focal length, aperture, and film type.

Set measurements Including camera height, focus distance, and measurements of various items in the shot.

Survey data This is very detailed measuring of the set, usually done by a professional surveyor.

Define the Camera

As stated before, the matchmover's job is to define all of the internal and external parameters of the camera, and there are many ways to do that. Knowing which method to use and under what circumstances comes with experience, but sometimes the only way to solve the matchmove is to experiment and see what works best. In broad terms, there are two major ways of solving for the camera: manual and automatic.

The manual methods harken back to the days before software existed to help matchmovers. This category would include perspective matching (matching the perspective of a single background image, rather than an image sequence, which is covered later in this chapter) and old-fashioned hand-tracking. This method of tracking a sequence involves making a speculation as to the camera's position and then refining it over many iterations until a match is achieved. Tracking a camera by hand is no small feat. It can often take weeks to truly figure out what is happening, because the process is nothing more than making educated guesses and tweaking them until they work.

In the past five or six years, software has emerged that allows a matchmover to track cameras somewhat automatically using a sophisticated technology known as *photogrammetry*. These software packages (which are covered in the next chapter) usually have a similar workflow to them. First, features in the image such as props in the scene or tape markers (commonly used on blue screen footage to mark points on the wall) are "tracked" as they move around the image in 2D. Then the software performs a *calibration* (or *solve*) for the camera by mathematically analyzing the movements of the 2D tracking markers. These packages usually generate an animated camera and 3D markers or nulls that represent the three-dimensional location of features that have been tracked in 2D. Matchmovers use this method most often since it is the easiest way to achieve a solution.

Some methods borrow from both manual and automatic techniques. Oftentimes, these are customized solutions that revolve around both types of workflows. For example, the 2D tracking information from matchmoving software could be used with a custom script that allows the matchmover to solve for pan shots.

Set Fitting

While cameras are the primary concern, they are only half of the process of matchmoving. Matchmovers must not only uncover all the facts about the camera, but they must also reconstruct the spatial layout of the environment on the live-action plate.

Figure 1.4 shows why this information is important. The first image shows an incorrect camera and building placement—what a mess. The second image has the correct camera, but the building is too close to the camera; therefore it doesn't match.

But notice the third image. In this case, the building is in the correct position and distance from the camera, but since the camera isn't correct, the building still doesn't line up. The last image shows how it should be matched up with the correct camera and building placement. These images illustrate how matchmoving is not just solving cameras, and not just solving environments, but also figuring out the relationship between the two.

Figure 1.4 These images show the relationship between solving correctly for the camera and correctly placing the set.

How much of the environment does the matchmover need to reproduce? That depends on what is being placed into the footage. If it is simply a character walking by, the animators and TDs might only need a simple ground plane. Other scenes might need rough geometry in order to cast 3D shadows. In some cases, such as digital set extensions, the matchmove might require an extremely accurate camera, detailed geometry, and spot-on positioning. Before beginning a matchmove, it is important to find out what type of 3D object is going into the scene and exactly where it will be placed.

3D environments might come from a variety of sources. Oftentimes, matchmovers create the rough geometry themselves or are provided a set to "fit" into the plate. And in some situations, the matchmover may provide a rough set from which a more detailed set is later constructed by a modeler. Many times, matchmovers use 3D markers they've calculated during the matchmove to provide information about the spatial relationships of the scene. But regardless of where the information comes from, it is often the matchmover's responsibility to establish the environment and set up the scene so that other artists further down the production pipeline don't need to worry about it.

Testing the Matchmove

Once the matchmove is solved, it needs to be tested for accuracy. A bad matchmove usually shows up as an obvious disconnect between the live-action plate and the CG elements. For example, the CG element seems to follow the motion and rotation of the live-action scene, but then suddenly the CG element pops to another location or gradually drifts away from the feature it's supposed to be resting on. Testing the matchmove consists of compositing the 3D objects over the image sequence and watching as the sequence moves to see if there are any unusual pops, drifts, or jitter.

Most often, matchmovers will have low-resolution objects, or *proxy objects*, in the scene to help them determine the quality of their matchmove. One of the best ways to see if an object is slipping is to place a checkerboard texture on it and render it out. This will help highlight slippage in 3D space.

A thorough testing is crucial, because a bad matchmove conceivably could work its way all the way through the pipeline and not be noticed until the end. And by this time there is usually a lot less time to deal with the problem and a lot more pressure.

Delivering the Scene

Last but not least, the final scene is delivered to other artists down the line. This step will vary greatly depending on whether the scene goes to a single artist working alone, or the scene is being turned over in a large production pipeline.

Certain items will need to be considered, such as:

- Orientation and scale of scene
- Objects that need to be included such as sets, characters, etc.
- Naming conventions, formats, etc.

A well-organized and clearly laid-out scene will make other artists' jobs easier and also make it less likely that you will have to explain the scene to someone after the fact.

Note: I've included a sample Scene Delivery Checklist in Appendix A. This checklist represents items I've found helpful in making my final matchmove scenes as foolproof as possible. I also cover this in more detail in Chapter 7, "Integrating Matchmoves with the Scene."

Matchmoving in the Production Pipeline

So how does matchmoving fit into the production pipeline? It really depends on the size of the production and the people who are being asked to provide the matchmoves. On larger-scale productions, there is often a department that deals specifically with matchmoving the shots. Of course, matchmoving is only necessary if you are trying to fit CG elements into live-action plates.

Matchmoving usually happens early in a production. This is because the animators and TDs need to know exactly where to place their characters, explosions, etc., and they need a camera from which to look at and render their objects. For tight deadlines, matchmovers might also be asked to set up temporary cameras that approximate the move, so that the TDs and animators can start their preliminary work and testing while the matchmove is finished.

Figure 1.5 shows a fairly common production pipeline and where matchmoving fits in. The matchmover takes the original footage at full resolution, solves the matchmove, and then delivers a 3D scene to an animator or TD. In some studios, matchmove data might be useful to other departments such as roto, paint, or compositing. (See Chapter 12, "Multipurposing Matchmove Data," for more on this.)

Perspective Matching Tutorial

Now that you know how matchmoving fits into the big picture, we're ready to get down to actually working on a shot. In this tutorial, you'll create a simple matchmove scene that solves a problem many CG artists have faced at one time or another: perspective matching.

Perhaps the simplest form of matchmoving is matching the perspective of a single still image. Oftentimes, these are used as backdrops for 3D elements. Since the camera is not moving, all that needs to be done is to position the camera relative to the object so that the perspective of the model looks appropriate. Even so, it can be frustrating to try and line things up.

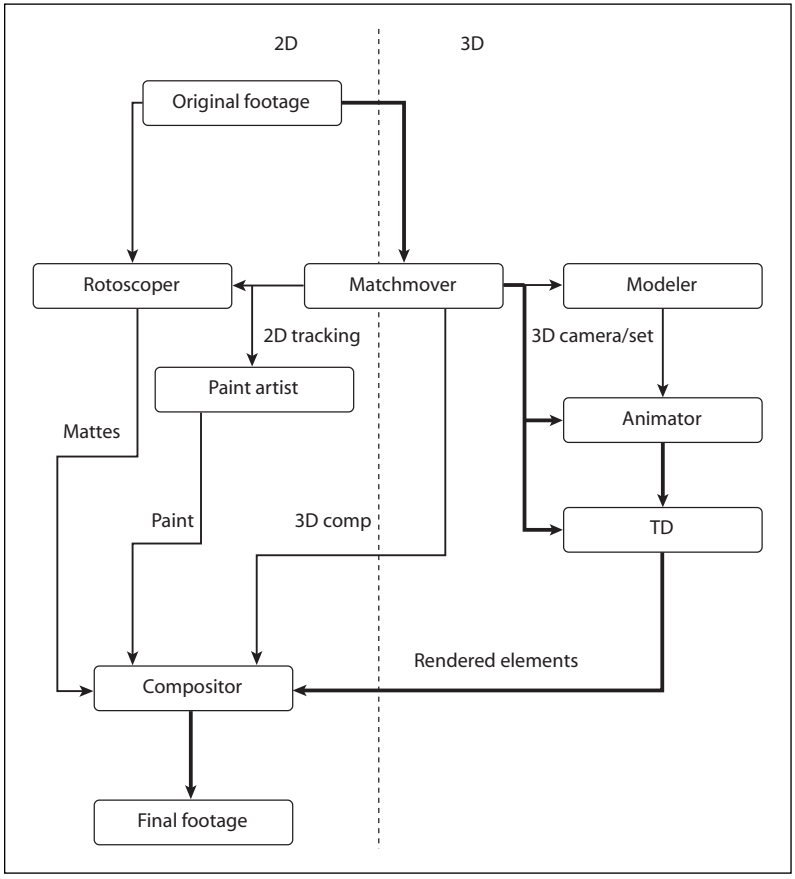

Figure 1.5 The normal pipeline has a matchmover delivering a scene to an animator or TD, but the matchmove data can also be used by other departments.

A few simple pieces of information can help make this process easier and less time-consuming:

- The focal length of the camera used to take the photo
- The height of the camera when the photo was shot
- The distance from the camera to an object in the real scene
- The scale of the object in the scene

Ideally, the person who took the image would provide all of the above information, but in most cases this is wishful thinking. For this tutorial, you'll need the image called f0106.tif (shown in Figure 1.6). For the purposes of this tutorial, let's assume a worst-case scenario: there isn't any information about the camera or environment for this image. The best way to deal with these situations is to approach the problem methodically. By making careful estimations and maintaining a 3D scene that is similar to the "real world" scene, we can zero in on a solution.

Figure 1.6 Our task is to create a camera so that 3D objects can be placed between the two buildings on the right.

Gathering the Data

Let's take a look at the image in Figure 1.6, in which we are asked to provide a camera and scene for the animators to work with. Consider what information we have and what we can do with it:

Focal length The focal length of the camera is a little bit tricky to figure out. For a still image, it's probably not necessary that it be exact, but simply in the ballpark. Until the building is in the right place, there's no way to play with this value, so it'll be necessary to make an initial guess and then try and zero in on a useable value later.

Measurements There aren't any measurements of the scene, but perhaps the strongest assumption that can be made about this image is that it was taken by a person standing

at street level. Therefore, it's reasonable to assume that the camera is a little less than 2 meters off the ground.

Note: This book uses metric measurements, because that is the standard practice in most studios.

Distance We don't have any idea of the distance to the buildings or the scale of the buildings themselves. If we were given a model that was based on the blueprints of the building, we would have some solid data to work with, but since we don't, we'll have to make a guess at this as well.

Setting Up the Camera

As is often the case with a matchmove, the process is not a linear one. Usually, it is an iterative one, where we do one or two things to get a rough result and then make a few more passes to refine it and build on known facts, until at some point we have matched it to the required level of accuracy. This example is no exception. Below, I show the steps to get the building matched.

1. **Evaluate the scene.** Since we're not dealing with images in motion, the key thing to figure out when matching the perspective of an image is where your 3D models need to go. Let's say that we want a spaceship to fly in between these two buildings and cast shadows on them as it goes by. That means we need to know the camera's position as well as the position of each of the buildings.
2. **Make a camera and create an image plane or background image using the still image.** First we need a camera to work with. This should be a freely moving camera (as opposed to a targeted or aimed camera). Since the ultimate goal is to align the objects with the plate, we will need to set up the camera or environment so that the image is seen as a backdrop behind the 3D objects when we look through the camera.
3. **Estimate focal length and aperture on the camera.** The two main things we'll need to know for the camera setup are the focal length and the film back. As we've already covered, we don't know the exact value for this lens, so we'll have to guess. I took a guess of 35mm for this lens as a starting point.
4. **Estimate the film back value.** In this case, I don't know the specific value for the film back. If I knew the brand of camera, I could look up the specs for it on the Internet and plug in those values. If you don't know the film back, it's

best to go with a standard value of 0.980″ × 0.735″ or use the default settings in your animation program. The reasons for these values are a bit complicated, so I'll save the explanation for Chapter 6, but for now this will give us something to work with.

5. **Set the height of the camera.** Since this is the strongest assumption we can make about the camera, we'll enter it in first. We'll assume an average height for the camera of about 1.75 meters off the ground, so place the camera at 1.75m up on the Y-axis.

Not All Film Backs Are the Same

Don't worry if you find yourself a little bit confused by all of this film back business. It can bewilder even the most seasoned visual effects pros. Oftentimes, 3D artists just guess at the value or use preset values in the software.

To make matters even more confusing, every 3D animation and matchmove program uses different terminology and has different ways of entering the information. Matchmove programs use the term "film back," whereas most 3D animation programs call this the "aperture." For example, 3D Studio Max calls it the Horizontal Aperture and only allows you to enter the horizontal measurement in millimeters. Lightwave uses the Vertical Aperture only, and Maya allows you to enter both Horizontal and Vertical Aperture measurements, but only in inches! You should consult the documentation for your software to see where you need to enter this information and how.

Adding Rough Geometry and Refining the Camera

Now that we have a camera set up, we need to start fleshing out the geometry in the scene. When the animators get ready to put the UFO in between the buildings, they'll need to know where the buildings are. The task of identifying the location of objects in the scene often falls on the shoulders of matchmovers. This geometry needs to be accurate to the image so that any additional objects that are added to the scene will appear to be in the correct location. You usually don't need to build geometry for the entire scene, just the portions that will interact directly with the CG objects.

I've found that creating geometry for a scene can be done in conjunction with the initial camera setup. Often you'll need to make continual refinements to the camera and geometry until you've zeroed in on the correct placement for both. The good news is, once you've built one or two objects, the rest become much simpler to build. Also,

once you've built objects to work with, you'll find it much easier to find the camera's exact position.

1. **Create a building of an estimated height.** This is where it starts to get tricky, since we really have little to go on here. I estimated the size by looking at the windows. I figured that each floor of the building was about 4 meters tall. So I counted the number of floors and multiplied it by 4m. I guessed that the building was a little under 14 floors, so 55 meters seemed about right. Likewise, I tried to use visual clues to estimate the width and depth of the building. After my initial guess, I came up with values for my building of X = 16.5m, Y = 55m, and Z = 20m.
2. **Move the building to an estimated distance.** This is likely to be the most difficult value to estimate. There aren't too many clues as to how far the photographer was away from the building, so I needed to guess how far away the building should be. I chose 78 meters by comparing the scale of my building as I moved it away from the camera. This, of course, assumes that the size of my CG building is correct. If not, we can always adjust this later (Figure 1.7).

Figure 1.7 For this image, the camera is at its correct height, and the building is approximately the same scale as the building on the right.

3. **Adjust the camera angle until the corner of your object is lined up with the corner of the building in the image.** Now that our camera is roughly positioned and set up, we can start the business of lining up the object. Rather than try and randomly match the entire object at once, it's easier to try and line up just one point on the

model to "lock down" one feature. This creates a connection or common point between the 3D scene and the 2D scene and creates a foundation on which to build a better match (Figure 1.8).

Figure 1.8 By rotating the camera, we can get the upper-right corner of our 3D building to match up with the image.

For now, don't concern yourself if the perspective or scale of the object is off, just focus on getting one corner of the 3D building to match with the corner of the building on the image. The idea here is to find one common point between the CG object and the real-world scene. If we can get one corner of each aligned properly, then we can use that as the basis for aligning the rest of the scene. For now we can assume that the camera height and distance are correct, so we don't want to translate (move) the model. In my scene, I simply rotated the camera until the upper-right corners lined up.

Creating a Camera Rig

Throughout this book, we'll be creating camera rigs in our 3D software. A *camera rig* is similar in concept to a real-life camera rig such as a boom. It simply is another way to move the camera around more conveniently in the scene. How these rigs are built will vary between the different software packages, but they generally consist of grouping or parenting the camera under one or more nulls (or similar nonrenderable objects) so that the camera can be moved around a different pivot point.

In this case, we have aligned the corners of the CG building and the building in the image, thereby establishing a link between the two. We want to be able to move the

camera while maintaining that alignment, so we'll create a rig that does exactly that. Here are the steps to create and position a camera rig:

1. **Make a parent object for the rig.** Create a null object in the scene and name it CameraRig. Snap the null to the upper-right corner of the 3D building that we lined up in the preceding section.

> **Note:** Null objects go by various names in different 3D packages. For example, in Maya they're referred to as "locators," and in 3D Studio Max they're called "Dummy Objects." Basically, whenever I use the term "null," I mean a 3D object that doesn't render and can be used in a parenting structure or to mark a 3D location.

2. **Parent the camera to the CameraRig null** (Figure 1.9). Take care that it stays in its position during the parenting (as opposed to snapping to position of the null).

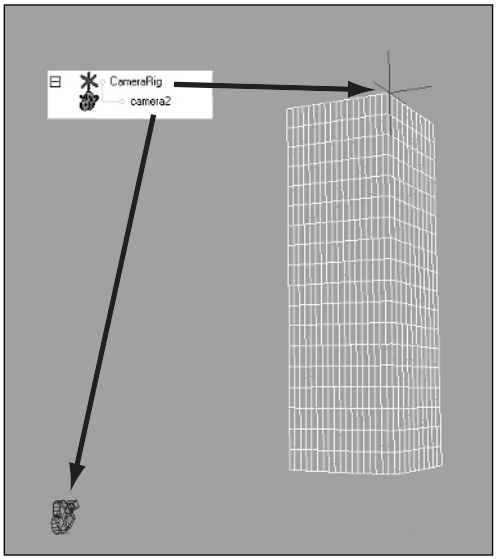

Figure 1.9 The camera rig is created by creating a null that is positioned on the upper-right corner of the image (the object that looks like a crosshair). Then the camera is parented to it.

3. **Use the camera rig to rotate the camera into position.** We can now rotate the CameraRig null to reposition the camera and try to match the perspective. We rotate it on the Y only because our strongest data suggests that the photo was shot at street level, and therefore we want to maintain that relationship. If the rig were rotated on the other axes, it would raise or lower the camera off the ground, and that would be one less piece of information we could take for granted.

If after rotating the camera rig on the Y the building still doesn't line up, you may have to adjust the rotation of the camera a bit more. This will cause the building's corner to come away from the building on the image, so you'll have to reposition it. You might have to go back and forth a few times between rotating the rig on the Y and rotating the camera to get the perspective working better (Figure 1.10).

Camera rigs give you more freedom and control over how you can position the camera. You'll use them quite often in the matchmoving process, and they can really help make it less frustrating.

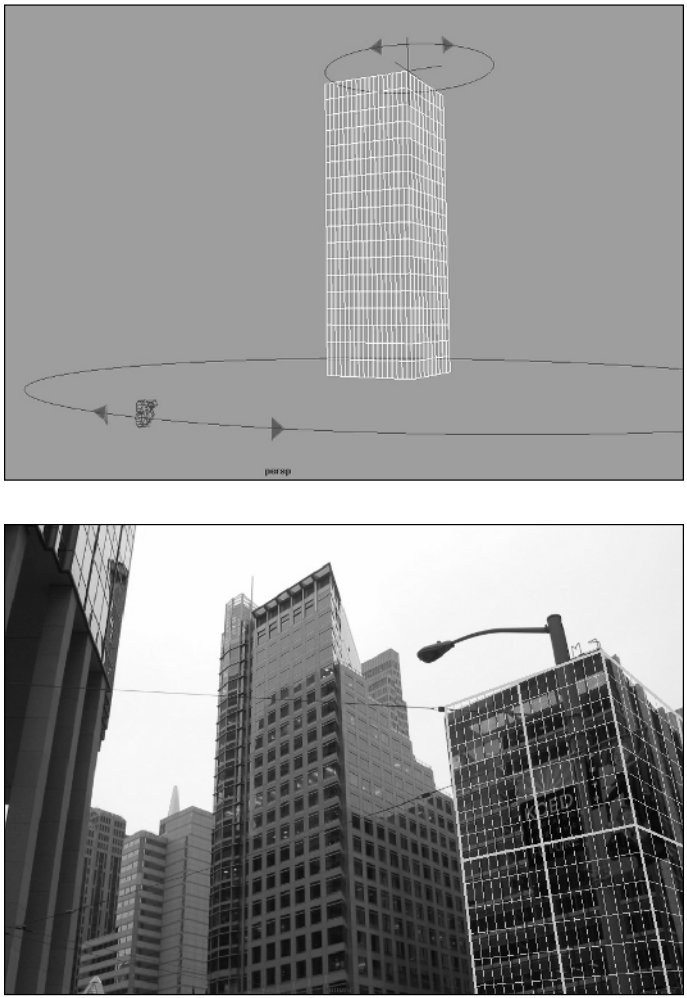

Figure 1.10 Note how the rig allows you to adjust the position of the camera simply by rotating the null. When you rotate the null on the Y-axis, the camera moves around the null on the X- and Z-axes. This keeps the corner of your 3D building on the corner of the building in the image while you adjust the perspective.

Evaluating and Adjusting the Camera

Up until this point, we've been working with approximations for our camera and building. Now we have everything we need to "tighten" up the matchmove and find the exact settings for the camera and the right size and shape for the building. In this phase, we are trying to bring the scene into perfect alignment.

1. **Is focal length right? Experiment.** Now that the camera is roughly positioned, we can do some experimenting to see if the focal length seems right. It looks pretty close, but it can't hurt to poke around some more. I tried different focal length values but ultimately decided that my original guess looked the best, so I stuck with that.

   The focal length can be a little tough to guess, but wider lenses (smaller focal lengths) tend to look more distorted, while longer lenses (higher focal lengths) look flatter. You may also need to do some adjustments to the building's position at this point as well. It's good to do this in tandem with adjusting the focal length, because both produce a similar result, making the building look larger or smaller in the frame (Figure 1.11).

Figure 1.11 Moving the building closer to the camera has made the building too tall and a little bit too narrow. A focal length of 35mm worked better.

2. **Make adjustments to building scale if necessary.** Now it's becoming obvious that the building's scale isn't too accurate, so we adjust it to fit better by scaling it to fit.

When you scale the building, you should scale it from one of the bottom corners. If you scale the building from the center, it will be more difficult to control. In most software packages, you can scale by moving the object's pivot point to the bottom corner of the object (Figure 1.12).

Figure 1.12 Rescaling the building from its bottom finishes off the matchmove.

3. **Continue adjusting the camera rotation, group rotation, and object placement until a fit is achieved.** From here, it's just a matter of tweaking the camera and/or building if necessary. One of the benefits of having created the camera rig is that it gives you several more degrees of freedom without losing accuracy. The CameraRig null allows you to spin the camera to different locations around the building (on the Y-axis only). If you need to adjust the distance of the camera, you can move the camera itself along the X- and Z-axes and rotate it on all three axes.
4. **Place a second building.** Once we've figured out the camera and the position of the first building, the second building's placement is easy. From the looks of it, they are both aligned at their fronts, so we can simply duplicate the existing building and slide it down the X-axis until the right edge of the 3D object lines up with the right edge of the building in the image. Next, we'll slide the model up until the corners are aligned and then scale the building to match (Figure 1.13).

With both buildings in place, you've provided a scene with the correct camera and enough information to show animators or TDs (or even yourself) where the CG object needs to go. If we were adding a UFO between the two buildings, as in Figure 1.14, our rough geometry would show us where to put it. These objects can also serve as geometry for reflections, shadows, or other effects.

Figure 1.13 The second building can be placed simply by duplicating the original and sliding it over on the X-axis.

Figure 1.14 Now we're ready to have our UFO do some fancy flying between the buildings.

Moving Toward Moving Pictures

So as you can see, matchmoving doesn't have to be a completely blind process of random guesses. By building on the information at hand, you can infer the information you don't have. Although we are working on a still image here, the same techniques can be applied to a moving sequence as well.

This method works well for these relatively simple still image situations, but for more complicated shots, it is useful to enlist the help of matchmoving software. Starting with the next chapter, I cover how these programs work, and how they can be used effectively to help solve even the toughest shots.