13 Frontiers

13.1 Haptics ........................................... 217
13.2 Taste and Smell .................................. 217
13.3 Robotic Interfaces ............................... 217
13.4 Brain-Machine Interfaces ......................... 217
Preface

The Rebirth of Virtual Reality

Virtual reality (VR) is a powerful technology that promises to change our lives unlike any other. By artificially stimulating our senses, our bodies become tricked into accepting another version of reality. VR is like a waking dream that could take place in a magical cartoon-like world, or could transport us to another part of the Earth or universe. It is the next step along a path that includes many familiar media, from paintings to movies to video games. We can even socialize with people inside of new worlds, either of which could be real or artificial.

At the same time, VR bears the stigma of unkept promises. The hype and excitement has often far exceeded the delivery of VR experiences to match it, especially for people without access to expensive laboratory equipment. This was particularly painful in the early 1990s when VR seemed poised to enter mainstream use but failed to catch on (outside of some niche markets). Decades later, we are witnessing an exciting rebirth. The latest technological components, mainly arising from the smartphone industry, have enabled high-resolution, low-cost, portable VR headsets to provide compelling VR experiences. This has mobilized leading technology companies to invest billions of US dollars into growing a VR ecosystem that includes art, entertainment, enhanced productivity, and social networks. At the same time, a new generation of technologists is entering the field with fresh ideas. Online communities of hackers and makers, along with college students around the world, are excitedly following the rapid advances in VR and are starting to shape it by starting new companies, working to improve the technology, and making new kinds of experiences.

The Intended Audience

The book is growing out of material for an overwhelmingly popular undergraduate course on VR that I introduced at the University of Illinois in 2015 (with hardware support from Oculus/Facebook). I have never in my life seen students so excited to take a course. We cannot offer enough slots to come even close to meeting the demand. Therefore, the primary target of this book is undergraduate students around the world. This book would be an ideal source for starting similar VR courses at other universities. Although most of the interested students have been computer scientists, the course at Illinois has attracted students from many disciplines, such as psychology, music, kinesiology, engineering, medicine, and economics. Students in these other fields come with the most exciting project ideas because they can see how VR has the potential to radically alter their discipline. To make the course accessible to students with such diverse backgrounds, I have made the material as self-contained as possible. There is no assumed background in software development or advanced mathematics. If prospective readers have at least written some scripts before and can remember how to multiply matrices together, they should be ready to go.

In addition to use by students who are studying VR in university courses, it also targeted at developers in industry, hobbyists on the forums, and researchers in academia. The book appears online so that it may serve as a convenient reference for all of these groups. To provide further assistance, there are also accompanying materials online, including lecture slides (prepared by Anna Yershova) and recorded lectures (provided online for free by NPTEL of India).

Why Am I Writing This Book?

I enjoy teaching and research, especially when I can tie the two together. I have been a professor and have taught university courses for two decades. Robotics has been my main field of expertise; however, in 2012, I started working at Oculus VR a few days after its Kickstarter campaign. I left the university and became their head scientist, working on head tracking methods, perceptual psychology, health and safety, and numerous other problems. I was struck at how many new challenges arose during that time because engineers and computer scientists (myself included) did not recognize human perception problems that were disrupting our progress. I became convinced that for VR to succeed, perceptual psychology must permeate the design of VR systems. As we tackled some of these challenges, the company rapidly grew in visibility and influence, eventually being acquired by Facebook for $2 billion in 2014. Oculus VR is largely credited with stimulating the rebirth of VR in the consumer marketplace.

I quickly returned to the University of Illinois with a new educational mission: Teach a new generation of students, developers, and researchers the fundamentals of VR in a way that fuses perceptual psychology with engineering. Furthermore, this book focuses on principles do not depend heavily on the particular technology of today. The goal is to improve the reader’s understanding of how VR systems work, what limitations they have, and what can be done to improve them. One important component is that even though technology rapidly evolves, humans who use it do not. It is therefore crucial to understand how our sensors systems function, especially with matched with artificial stimulation. This intent is to provide a useful foundation as the technology evolves. In many cases, open challenges remain. The book does not provide the solutions to them, but instead provides the background to begin researching them.
Online Materials

The entire book is posted online at:

http://msl.cs.uiuc.edu/vr/

along with pointers to additional materials, such as lecture videos and slides.

Suggested Use

This text may be used for a one-semester course by spending roughly one week per chapter, with the exception of Chapter 3 which may require two weeks. The book can also be used to augment other courses such as computer graphics, interfaces, and game development. Selected topics may also be drawn for a short course.

Depending on the technical level of the students, the mathematical concepts in Chapter 3 might seem too oppressive. If that is the case, students may be advised to skim over it and jump to subsequent chapters. They can understand most of the later concepts without the full mathematical details of Chapter 3. Nevertheless, understanding these concepts will enhance their comprehension throughout the book and will also make them more comfortable with programming exercises.

Lab Component

We currently use Oculus Rift DK2s on gaming PCs with expensive graphics cards (nVidia Titan Black with 6GB RAM). Development on many more platforms will soon become feasible for this course, including Samsung Gear VR, HTC Vive, and even Google Cardboard, but the quality is generally unacceptable. For software, almost all students develop VR projects using Unity 3D. Alternatives may be Unreal Engine and CryENGINE, depending on their level of technical coding skills. Unity 3D is the easiest because knowledge of C++ and associated low-level concepts is unnecessary.

Acknowledgments

I am very grateful to many students and colleagues who have given me extensive feedback and advice in developing this text. It evolved over many years through the development and teaching at the University of Illinois. The biggest thanks go to Anna Yershova, who has also taught the virtual reality course at the University of Illinois and collaborated on the course development. We also worked side-by-side at Oculus VR since the earliest days.

Among many helpful colleagues, I especially thank Ian Bailey, Paul MacNeilage, Betty Mohler, Aaron Nichols, Yury Petrov, Dan Simons, and Richard Yao for their helpful insights, explanations, suggestions, feedback, and pointers to materials.
Chapter 1

Introduction

<table>
<thead>
<tr>
<th>Chapter Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Taken from Virtual Reality, S. M. LaValle</td>
</tr>
</tbody>
</table>

This online chapter is not the final version! Check http://msl.cs.uiuc.edu/vr/ for information on the latest draft version.

This draft was compiled on March 2, 2016.

1.1 What Is Virtual Reality?

Virtual reality (VR) technology is evolving rapidly, making it precarious to define VR in terms of specific devices that may fall out of favor in a year or two. In this book, we are concerned with fundamental principles that are less sensitive to particular technologies and therefore survive the test of time. Our first challenge is to consider what VR actually means, in a way that captures the most crucial aspects in spite of rapidly changing technology. The concept must also be general enough to encompass what VR is considered today and what we envision for its future.

We start with two representative examples that employ current technologies: 1) A human having an experience of flying over virtual San Francisco by flapping his own wings (Figure 1.1); 2) a mouse running on a freely rotating ball while exploring a virtual maze that appears on a projection screen around the mouse (Figure 1.2). We want our definition of VR to be broad enough to include these examples and many more, which are coming in Section 1.2. This motivates the following.

**Definition of VR:** Inducing targeted behavior in an organism by using artificial sensory stimulation, while the organism has little or no awareness of the interference.

Figure 1.1: In the Birdly experience from the Zurich University of the Arts, the user, wearing a VR headset, flaps his wings while flying over virtual San Francisco, while a motion platform and fan provide additional sensory stimulation. The figure on the right shows the stimulus presented to each eye.

Figure 1.2: (a) An experimental setup used by neurobiologists at LMU Munich to present visual stimuli to rodents while they run on a spherical ball that acts as a treadmill (Figure by Kay Thurley). (b) A picture of a similar experiment, performed at Princeton University.
1.1. WHAT IS VIRTUAL REALITY?

Terminology regarding various “worlds” Several terms related to VR are in common use at present. The term virtual environments predates widespread usage of VR, and is commonly considered to be synonymous; however, we emphasize in this book that VR can be an interface to a captured “real” world just as well as experiencing a completely artificial world. Augmented reality (AR) refers to systems in which most of the visual stimuli are propagated directly through glass or cameras to the eyes, and some additional structures appear to be superimposed onto the user’s world. The term mixed reality is sometimes used to refer to an entire spectrum that encompasses VR, AR, and normal reality. Telepresence refers to systems that enable users to feel like they are somewhere else in the real world; if they are able to control anything, such as a flying drone, then teleoperation is an appropriate term. For our purposes, virtual environments, AR, mixed reality, telepresence, and teleoperation will all be considered as perfect examples of VR. The most important idea of VR is that the user’s perception of reality has been altered through engineering, rather than whether the environment they believe they are in seems more “real” or “virtual”. We will instead use these terms to distinguish whether VR is employed: The real world refers to the physical world that contains the user, and the virtual world refers to the perceived world as part of the targeted VR experience.

Interactivity Most VR experiences involve another crucial component: interactivity. Does the sensory stimulation depend on actions taken by the organism? If the answer is “no”, then the VR system is called open-loop; otherwise, it is closed-loop. In the case of closed-loop VR, the organism has partial control over the stimulation, which could vary as a result of body motions, including eyes, head, hands, or legs. Other possibilities include voice commands, heart rate, body temperature, and skin conductance (are you sweating?).

First- vs. Third-person If you are reading this book, then you most likely want to develop VR systems or experiences. Pay close attention to this next point! When a scientist designs an experiment for an organism, as shown in Figure 1.2, then the separation is clear: The laboratory subject (organism) has a first-person experience, while the scientist is a third-person observer. The scientist carefully designs the VR system as part of an experiment that will help to resolve a scientific hypothesis. For example, how does turning off a few neurons in a rat’s brain affect its navigation ability? On the other hand, when engineers or developers construct a VR system or experience, they are usually targeting themselves and people like them. They feel perfectly comfortable moving back and forth between being...
1.1. WHAT IS VIRTUAL REALITY?

the "scientist" and the "lab subject" while evaluating and refining their work. As you will learn throughout this book, this is a bad idea! The creators of the experience are heavily biased by their desire for it to succeed without having to redo their work. They also know what the experience is supposed to mean or accomplish, which provides a strong bias in comparison to a fresh subject. To complicate matters further, the creator's body will physically and mentally adapt to whatever flaws are present so that they may soon become invisible. We have seen these kinds of things before. For example, it is hard to predict how others will react to your own writing. Also, it is usually harder to proofread your own writing in comparison to that of others. In the case of VR, these effects are much stronger and yet elusive to the point that you must force yourself to pay attention to them. Take great care when hijacking the senses that you have trusted all of your life. This will most likely be uncharted territory for you.

More real than reality? How “real” should the VR experience be? It is tempting to try to make it match our physical world as closely as possible. Our brains are most familiar with this setting, thereby making it seem most appropriate. This philosophy has dominated the video game industry at times, for example, in the development of highly realistic first-person-shooter (FPS) games that are beautifully rendered on increasingly advanced graphics cards. In spite of this, understand that extremely simple, cartoon-like environments can also be effective and even preferable. Examples appear throughout history, as discussed in Section 1.3.

As a VR experience creator, think carefully about the task, goals, or desired effect you want to have on the user. You have the opportunity to make the experience “better than real”. What will they be doing? Taking a math course? Experiencing a live theatrical performance? Writing software? Designing a house? Maintaining a long-distance relationship? Playing a game? Having a meditation and relaxation session? Traveling to another place on Earth, or in the universe? For each of these, think about how the realism requirements might vary. For example, consider writing software in VR. We currently write software by typing into windows that appear on a large screen. Note that even though this is a familiar experience for many people, it was not even possible in the physical world of the 1950s. In VR, we could simulate the modern software development environment by convincing the programmer that she is sitting in front of a screen; however, this misses the point that we can create almost anything in VR. Perhaps a completely new interface will emerge that does not appear to be a screen sitting on a desk in an office. For example, the windows could be floating above a secluded beach or forest. Furthermore, imagine how a debugger could show the program execution trace.

Synthetic vs. captured Two extremes exist when constructing a virtual world as part of a VR experience. At one end, we may program a synthetic world, which is completely invented from geometric primitives and simulated physics. This is common in video games and such virtual environments were assumed to be the main way to experience VR in earlier decades. At the other end, the world may be captured using modern imaging techniques. For viewing on a screen, the video camera has served this purpose for over a century. Capturing panoramic images and videos and then seeing them from any viewpoint in a VR system is a natural extension. In many settings, however, too much information is lost when projecting the real world onto the camera sensor. What happens when the user changes her head position and viewpoint? More information should be captured in this case. Using depth sensors and SLAM (Simultaneous Localization And Mapping) techniques, a 3D representation of the surrounding world can be captured and maintained over time as it changes. It is extremely difficult, however, to construct an accurate and reliable representation, unless the environment is explicitly engineered for such capture (for example, a motion capture studio).

As humans interact, it becomes important to track their motions, which is an important form of capture. What are their facial expressions while wearing a VR headset? Do we need to know their hand gestures? What can we infer about their emotional state? Are their eyes focused on me? Synthetic representations of ourselves called avatars enable us to interact and provide a level of anonymity, if desired in some contexts. The attentiveness or emotional state can be generated synthetically. We can also enhance our avatars by tracking the motions and other attributes of our actual bodies. A well-known problem is the uncanny valley, in which a high degree of realism has been achieved in a avatar, but its appearance makes people feel uneasy. It seems almost right, but the small differences are disturbing. There is currently no easy way to make ourselves appear to others in a VR experience exactly as we do in the real world, and in most cases, we might not want to.

Health and safety Although the degree of required realism may vary based on the tasks, one requirement remains invariant: The health and safety of the users. Unlike simpler media such as radio or television, VR has the power to overwhelm the senses and the brain, leading to fatigue or sickness. This phenomenon has been studied under the heading simulator sickness for decades; in this book we will refer to adverse symptoms from VR usage as VR sickness. Sometimes the discomfort is due to problems in the VR hardware and low-level software; however, in most cases, it is caused by a careless VR developer who misunderstands or disregards the side effects of the experience on the user. This is one reason why human physiology and perceptual psychology are large components of this book. To develop comfortable VR experiences, you must understand how these factor in. In many cases, fatigue arises because the brain appears to work harder to integrate the unusual stimuli being presented to the senses. In some cases, inconsistencies with prior expectations, and outputs from other senses, even lead to dizziness and nausea.

Another factor that leads to fatigue is an interface that requires large amounts of muscular effort. For example, it might be tempting move objects around in a sandbox game by moving your arms around in space. This quickly leads to
fatigue and an avoidable phenomenon called gorilla arms, in which people feel that the weight of their extended arms is unbearable. For example, by following the principle of the computer mouse, it may be possible to execute large, effective motions in the virtual space by small, comfortable motions of a controller. Over long periods of time, the brain will associate the motions well enough for it to seem realistic while also greatly reducing fatigue.

### 1.2 Modern VR Experiences

This section gives you a quick overview of what people are doing with VR today, and provides a starting point for searching for similar experiences on the Internet. Here, we can only describe the experiences in words and pictures, which is a long way from the appreciation gained by experiencing them yourself. This printed medium (a book) is woefully inadequate for fully conveying the medium of VR. Perhaps this is how it was in the 1890s to explain in a newspaper what a movie theater was like! If possible, it is strongly recommended that you try many VR experiences yourself to form first-hand opinions and spark your imagination to do something better.

**Video games**  People have dreamed of entering their video game worlds for decades. By 1982, this concept was already popularized by the Disney movie Tron. Figure 1.4(a) shows several video game experiences in VR. Most gamers currently want to explore large, realistic worlds through an avatar. Figure 1.4(a) shows Valve’s Portal 2, which is a puzzle-solving adventure game developed for the HTC Vive VR headset. Figure 1.4(b) shows an omnidirectional treadmill peripheral that gives users the sense of walking while they slide their feet in a dish on the floor. These two examples give the user a first-person perspective of their character. By contrast, Figure 1.4(c) shows Lucky’s Tale, which instead yields a comfortable third-person perspective as the player floats above his character.

**Immersive cinema**  Hollywood movies continue to offer increasing degrees of realism. Why not make the viewers feel like they are part of the scene? Figure 1.5 shows an immersive short story. Movie directors are entering a fascinating new era of film. The tricks of the trade that were learned across the 20th century need to be reinvestigated because they are based on the assumption that the cinematographer controls the camera viewpoint. In VR, viewers can look in any direction, and perhaps even walk through the scene. What should they be allowed to do? How do you make sure they do not miss part of the story? Should the story be linear, or should it adapt to the viewer’s actions? Should the viewer be a first-person character in the film, or a third-person observer who in invisible to the other characters? How can a group of friends experience a VR film together? When are animations more appropriate versus the capture of real scenes?

It will take many years to resolve these questions and countless more that will arise. In the meantime, VR can also be used as a kind of “wrapper” around existing movies. Figure 1.6 shows the VR Cinema application, which allows the user to choose any seat in a virtual movie theater. Whatever standard movies or videos that are on the user’s hard drive can be streamed to the screen in the theater. These could be 2D or 3D movies. A projector in the back emits flickering lights and the audio is adjusted to mimic the acoustics of a real theater. This provides an immediate way to leverage all content that was developed for viewing on a screen, and bring it into VR. Many simple extensions can be made without modifying the films. For example, in a movie about zombies, a few virtual zombies could enter the theater and start to chase you. In a movie about tornados, perhaps the theater rips apart. You can also have a social experience. Imagine having “movie night” with your friends from around the world, while you sit together in the virtual movie theater. You can even have the thrill of misbehaving in the theater without getting thrown out.

Figure 1.4: (a) Valve’s Portal 2 demo for the HTC Vive headset is a puzzle-solving experience in a virtual world. (b) The Virtuix Omni treadmill for walking through first-person-shooter games. (c) Lucky’s Tale for the Oculus Rift maintains a third-person perspective as the player floats above his character. (d) In the Dumpy game from DePaul University, the player appears to have a large elephant trunk. The purpose of the game is to enjoy this phenomenon while knocking things down.
1.2. MODERN VR EXPERIENCES

Figure 1.5: Oculus Story Studio produced 
*Henry*, an immersive short story about an unloved
hedgehog who hopes to make a new friend, the viewer.

Figure 1.6: VR Cinema, developed by Joo-Hyung Ahn for the
Oculus Rift. You can choose your seat and watch any movie you like.

Figure 1.7: An important component for achieving telepresence is to capture a panoramic view: (a) A car with cameras and
depth sensors on top, used by Google to make Street View. (b) Bublcam is a cheap, portable way to capture and stream
omnidirectional videos.

**Telepresence** The first step toward feeling like we are somewhere else is capturing a panoramic view of the remote environment (Figure 1.7). Google’s Street View and Earth apps already rely on the captured panoramic images from millions of locations around the world. Simple VR apps that query the Street View server directly enable user to feel like he is standing in each of these locations, while easily being able to transition between nearby locations (Figure 1.8). Panoramic video capture is even more compelling. Figure 1.9 shows a frame from an immersive rock concert experience. Even better is to provide live panoramic video interfaces, through which people can attend sporting events and concerts. Through a live interface, interaction is possible. People can take video conferencing to the next level by feeling present at the remote location. By connecting panoramic cameras to robots, the user is even allowed to move around in the remote environment (Figure 1.10). Current VR technology allows us to virtually visit far away places and interact in most of the ways that were previously possible only while physically present. This leads to improved opportunities for telecommuting to work. This could ultimately help reverse the urbanization trend sparked by the 19th-century industrial revolution, leading to *deurbanization* as we distribute more uniformly around the Earth.

**Virtual societies** Whereas telepresence makes us feel like we are in another part of the physical world, VR also allows us to form entire societies that remind us of the physical world, but are synthetic worlds that contain avatars connected to real people. Figure 1.11 shows a Second Life scene in which people interact in a fantasy world through avatars; such experiences were originally designed to view on a screen but can now be experienced through VR. Groups of people could spend time together in these spaces for a variety of reasons, including common
1.2. MODERN VR EXPERIENCES

Figure 1.8: A simple VR experience that presents Google Street View images through a VR headset: (a) A familiar scene in Paris. (b) Left and right eye views are created inside the headset, while also taking into account the user’s looking direction.

Figure 1.9: Jaunt captured a panoramic video of Paul McCartney performing Live and Let Die, which provides a VR experience where users felt like they were on stage with the rock star.

Figure 1.10: Examples of robotic avatars: (a) The DORA robot from the University of Pennsylvania mimics the users head motions, allowing him to look around in a remote world while maintaining a stereo view (panoramas are monoscopic). (b) The Plexidrone, a low-cost flying robot that is designed for streaming panoramic video.

Figure 1.11: Virtual societies develop through interacting avatars that meet in virtual worlds that are maintained on a common server. A snapshot from Second Life is shown here.
special interests, educational goals, or simple an escape from ordinary life.

Empathy  The first-person perspective provided by VR is a powerful tool for causing people to feel empathy for someone else’s situation. The world continues to struggle with acceptance and equality for others of different race, religion, age, gender, sexuality, social status, and education, while the greatest barrier to progress is that most people cannot fathom what it is like to have a different identity. Figure 1.12 shows a VR project sponsored by the United Nations to yield feelings of empathy for those caught up in the Syrian crisis of 2015. Some of us may have compassion for the plight of others, but it is a much stronger feeling to understand their struggle because you have been there before. Figure 1.13 shows a VR system that allows men and women to swap bodies. Through virtual societies, many more possibilities can be explored. What if you were 10cm shorter than everyone else? What if you teach your course with a different gender? What if you were the victim of racial discrimination by the police? Using VR, we can imagine many “games of life” where you might not get as far without being in the “proper” group.

Education  In addition to teaching empathy, the first-person perspective could revolutionize many areas of education. In engineering, mathematics, and the sciences, VR offers the chance to visualize geometric relationships in difficult concepts or data that is hard to interpret. Furthermore, VR is naturally suited for practical training because skills developed in a realistic virtual environment may transfer naturally to the real environment. The motivation is particularly high if the real environment is costly to provide or poses health risks. One of the earliest and most common examples of training in VR is flight simulation (Figure 1.14). Other examples include firefighting, nuclear power plant safety, search-and-rescue, military operations, and medical procedures.

Beyond these common uses of VR, perhaps the greatest opportunities for VR education lie in the humanities, including history, anthropology, and foreign language acquisition. Consider the difference between reading a book on the Victorian era in England and being able to roam the streets of 19th-century London, in a simulation that has been painstakingly constructed by historians. We could even visit an ancient city that has been reconstructed from ruins (Figure 1.15). Fascinating possibilities exist for either touring physical museums through a VR interface or scanning and exhibiting artifacts directly in virtual museums.

Virtual prototyping  In the real world, we build prototypes to understand how a proposed design feels or functions. Thanks to 3D printing and related technologies, this is easier than ever. At the same time, virtual prototyping enables designers to inhabit a virtual world that contains their prototype (Figure 1.16). They can quickly interact with it and make modifications. They also have the opportunities to bring clients into their virtual world so that they can communicate
1.2. MODERN VR EXPERIENCES

Figure 1.14: A flight simulator used by the US Air Force (photo by Javier Garcia). The user sits in a physical cockpit while being surrounded by displays that show the environment.

Figure 1.15: A tour of the Nimrud palace of Assyrian King Ashurnasirpal II, a VR experience developed by Learning Sites Inc. and the University of Illinois.

Figure 1.16: Architecture is a prime example of where a virtual prototype is invaluable. This demo, called Ty Hedfan, was created by designer Olivier Demangel. The real kitchen is above and the virtual kitchen is below.
1.2. MODERN VR EXPERIENCES

Figure 1.17: A heart visualization system based on images of a real human heart. This was developed by the Jump Trading Simulation and Education Center and the University of Illinois.

their ideas. Imagine you want to remodel your kitchen. You could construct a model in VR and then explain to a contractor exactly how it should look. Virtual prototyping in VR has important uses in many businesses, including real estate, architecture, and the design of aircraft, spacecraft, cars, furniture, clothing, and medical instruments.

Health care Although health and safety are challenging VR issues, the technology can also help to improve our health. There is an increasing trend toward distributed medicine, in which doctors train people to perform routine medical procedures in remote communities around the world. Doctors can provide guidance through telepresence, and also use VR technology for training. In another use of VR, doctors can immerse themselves in 3D organ models that were generated from medical scan data (Figure 1.17). This enables them to better plan and prepare for a medical procedure by studying the patient’s body shortly before an operation. They can also explain medical options to the patient or his family so that they may make more informed decisions. In yet another use, VR can directly provide therapy to help patients. Examples include overcoming phobias and stress disorders through repeated exposure, improving or maintaining cognitive skills in spite of aging, and improving motor skills to overcome balance, muscular, or nervous system disorders.

1.3. History Repeats

Figure 1.18: (a) Epic Games created a wild roller coaster ride through virtual living room. (b) A guillotine simulator was made by Andre Berlemont, Morten Brnbrjerg, and Erkki Trummal. Participants were hit on the neck by friends as the blade dropped, and they could see the proper perspective as their heads rolled.

New human experiences Finally, the point might be to simply provide a new human experience. Through telepresence, people can try experiences through the eyes of robots or other people. However, we can go further by giving people experiences that are impossible (or perhaps deadly) in the real world. Most often, artists are the ones leading this effort. The Birdly experience of human flying (Figure 1.1) was an excellent example. Figure 1.18 shows two more. What if we change our scale? Imagine being 2mm tall and looking ants right in the face. Compare that to being 50m tall and towering over a city while people scream and run from you. What if we simulate the effect of drugs in your system? What if you could become your favorite animal? What if you became a piece of food? The creative possibilities for artists seem to be endless. We are limited only by what our bodies can comfortably handle. Exciting adventures lie ahead!

Staring at rectangles How did we arrive to VR as it exists today? We start with a history that predates what most people would consider to be VR, but includes many aspects crucial to VR that have been among us for tens of thousands of years. Long ago, our ancestors were trained to look at the walls and imagine a 3D world that is part of a story. Figure 1.19 shows some examples of this. Cave paintings, such as the one shown in Figure 1.19(a) from 30,000 years ago. Figure 1.19(b) shows a painting from the European Middle Ages. Similar to the cave painting, it relates to military conflict, a fascination of humans regardless of the era or technology. There is much greater detail in the newer painting, leaving
Figure 1.19: (a) A 30,000-year-old painting from the Bhimbetka rock shelters in India (photo by Archaeological Survey of India). (b) An English painting from around 1470 that depicts John Ball encouraging Wat Tyler rebels (unknown artist). (c) A painting by Hans Vredeman de Vries in 1596. (d) An impressionist painting by Claude Monet in 1874.

Moving pictures Once humans were content with staring at rectangles on the wall, the next step was to put them into motion. The phenomenon of *stroboscopic apparent motion* is the basis for what we call *movies* or *motion pictures* today.
1.3. HISTORY REPEATS

Figure 1.21: A progression of special effects: (a) Arrival of a Train at La Ciotat Station, 1896. (b) A Trip to the Moon, 1902. (c) The movie 2001, from 1968. (d) Gravity, 2013.

Toward convenience and portability Another motivation for accepting lower levels of realism is cost and portability. As shown in Figure 1.22, families were willing to gather in front of a television to watch free broadcasts in their homes, even though they could go to theaters and watch high-resolution, color, panoramic, and 3D movies at the time. Such tiny, blurry, black-and-white television sets seem comically intolerable with respect to our current expectations. The next level of portability is to carry the system around with you. Thus, the progression is from: 1) having to go somewhere to watch it, to 2) being able to watch it in your home, to 3) being able to carry it anywhere. Whether pictures, movies, phones, computers, or video games, the same progression continues. We can therefore expect the same for VR systems. At the same time, note that the gap is closing between these levels: The quality we expect from a portable device is closer than ever before to the version that requires going somewhere to experience it.

Video games Motion pictures yield a passive, third-person experience, in contrast to video games which are closer to a first-person experience by allowing us to interact with him. Recall from Section 1.2 the differences between open-loop and closed-loop VR. Video games are an important step closer to closed-loop VR, whereas motion pictures are open-loop. As shown in Figure 1.23 we see the same trend from simplicity to improved realism and then back to simplicity. The earliest games, such as Pong and Donkey Kong, left much to the imagination. First-person shooter (FPS) games such as Doom gave the player a first-person perspective and launched a major campaign over the following decade toward higher quality graphics and realism. Assassin’s Creed shows a typical scene from a modern, realistic video game. At the same time, wildly popular games have emerged by focusing on simplicity. Angry Birds looks reminiscent of games from the 1980s, and Minecraft allows users to create and inhabit worlds composed of course blocks. Note that reduced realism often leads to simpler engineering requirements; in 2015, an advanced FPS game might require a powerful PC and graphics card, while simpler games would run on a basic smartphone. Repeated lesson: Don’t assume that more realistic is better!
Beyond staring at a rectangle The concepts so far are still closely centered on staring at a rectangle that is fixed on a wall. Two important steps come next: 1) Presenting a separate picture to each eye to induce a “3D” effect. 2) Increasing the field of view so that the user is not distracted by anything but the stimulus. One way our brains infer the distance of objects from our eyes is by stereopsis. Information is gained by observing and matching features in the world that are visible to both the left and right eyes. The differences between their images on the retina yield cues about distances; keep in mind that there are many more such cues, which we discuss in Section 6.1. The first experiment that showed this 3D effect of stereopsis was performed in 1838 by Charles Wheatstone in a system called the stereoscope (Figure 1.25(a)). By the 1930s, a portable version became a successful commercial product known to this day as the View-Master (Figure 1.25(b)). Pursuing this idea further led to Sensorama, which added motion pictures, sound, vibration, and even smells to the experience (Figure 1.25(c)). An unfortunate limitation of these designs is requiring that the viewpoint is fixed with respect to the picture. If the device is too large, then the user’s head also becomes fixed in the world. An alternative has been available in movie theaters since the 1950s. Stereopsis is achieved when participants wore special glasses that select
1.3. HISTORY REPEATS

(a) The first stereoscope, developed by Wheatstone in 1838, used mirrors to present a different image to each eye; the mirrors were replaced by lenses soon afterward. (b) The View-Master is a mass-produced stereoscope that has been available since the 1930s. (c) In 1957, Sensorama added motion pictures, sound, vibration, and even smells to the experience. (d) In competition to stereoscopic viewing, Cinerama offered a larger field of view. Larger movie screens caused the popularity of 3D movies to wane in the 1950s.

VR headsets Once again, the trend toward portability appears. An important step for VR was taken in 1968 with the introduction of Ivan Sutherland’s Sword of Damocles, which leveraged the power of modern displays and computers (Figure 1.26(b)). He constructed what is widely considered to be the first VR headset. As the user turns his head, the images presented on the screen are adjusted to compensate so that the virtual objects appear to be fixed in space. This yielded the first glimpse of an important concept in this book: The perception of stationarity. To make an object appear to be stationary while you move your sense organ, the device producing the stimulus must change its output to compensate for the motion. This requires sensors and tracking systems to become part of the VR system. Commercial VR headsets started appearing in the 1980s with Jaron Lanier’s company VPL, thereby popularizing the image of goggles and gloves; Figure 1.26(c). In the 1990s, VR-based video games appeared in arcades (Figure 1.26(d)) and at home units (Figure 1.26(e)). The experiences were not compelling or comfortable enough to attract mass interest. However, the current generation of VR headset leverages the widespread availability of high resolution screens and sensors, due to the smartphone industry, to offer lightweight, low-cost, high-field-of-view headsets, such as the Oculus Rift (Figure 1.26(f)). This has greatly improved the quality of VR experiences while significantly lowering the barrier of entry for developers and hobbyists. This also caused a recent flood of interest in VR technology and applications.

Bringing people together We have so far neglected an important aspect, which is human-to-human or social interaction. We use formats such as a live theater performance, a classroom, or a lecture hall for a few people to communicate with or entertain a large audience. We write and read novels to tell stories to each other. Prior to writing, skilled storytellers would propagate experiences to others, including future generations. We have communicated for centuries by writing letters to each other. More recent technologies have allowed us to interact directly without delay. The audio part has been transmitted through telephones for over a century, and now the video part is transmitted as well through videocon-
Figure 1.26: (a) CAVE VR, 1992. (b) Sword of Damocles, 1968. (c) VPL Eye-phones, 1980s. (d) Virtuality gaming, 1990s. (e) Nintendo Virtual Boy, 1995. (f) Oculus Rift DK2, 2014.

Figure 1.27: Second Life was introduced in 2003 as a way for people to socialize through avatars and essentially build a virtual world to live in. Shown here is the author giving a keynote address at the 2014 OpenSimulator Community Conference. The developers build open source software tools for constructing and hosting such communities of avatars with real people behind them.

S. M. LaValle: Virtual Reality

Returning to VR, we can create avatar representations of ourselves and “live” together in virtual environments, as is the case with Second Life and OpenSimulator. Without being limited to staring at rectangles, what kinds of societies will emerge with VR? Popular science fiction novels have painted a thrilling, yet dystopian future of a world where everyone prefers to interact through VR. It remains to be seen what the future will bring.

As the technologies evolve over the years, keep in mind the power of simplicity when making a VR experience. In some cases, maximum realism may be important; however, leaving much to the imagination of the users is also valuable. Although the technology changes, one important invariant is that humans are still
designed the same way. Understanding how our senses, brains, and bodies work is crucial to understanding the fundamentals of VR systems.

Further reading


Chapter 2

Bird’s-Eye View

This chapter presents an overview of VR systems from hardware (Section 2.1) to software (Section 2.2) to human perception (Section 2.3). The purpose is to quickly provide a complete perspective so that the detailed subjects in the remaining chapters are understood within the larger context.

2.1 Hardware

The first step to understanding how VR works is to consider what constitutes the entire VR system. It is tempting to think of it as being merely the hardware components, such as computers, headsets, and controllers. This would be woefully incomplete. As shown in Figure 2.1, it is equally important to account for the organism, which in this chapter will exclusively refer to a human user. The hardware produces stimuli that override the senses of the user. In the Sword of Damocles from Section 1.3 (Figure 1.26(b)), recall that tracking is needed to adjust the stimulus based on human motions. The VR hardware accomplishes this by using its own sensors, thereby tracking motions of the user. Head tracking is the most important, but tracking also may include button presses, controller movements, eye movements, or the movements of any other body parts. Finally, it is also important to consider the surrounding physical world as part of the VR system. In spite of stimulation provided by the VR hardware, the user will always have other senses that respond to stimuli from the real world. She also has the ability to change her environment through body motions. The VR hardware might also track objects other than the user, especially if interaction with them is part of the VR experience. Through a robotic interface, the VR hardware might also change the real world. One example is teleoperation of a robot through a VR interface.

Sensors and sense organs How is information extracted from the physical world? Clearly this is crucial to a VR system. In engineering, a transducer refers to a device that converts energy from one form to another. A sensor is a special transducer that converts the energy it receives into a signal for an electrical circuit. This may be an analog or digital signal, depending on the circuit type. A sensor typically has a receptor that collects the energy for conversion. Organisms work in a similar way. The “sensor” is called a sense organ, with common examples being eyes and ears. Because our “circuits” are formed from interconnected neurons, the sense organs convert energy into neural impulses. As you progress through this book, keep in mind the similarities between engineered sensors and natural sense organs. They are measuring the same things and sometimes even function in a similar manner. This should not be surprising because we and our engineered devices share the same physical world: The laws of physics and chemistry remain the same.

Configuration space of sense organs As the user moves through the physical world, his sense organs move along with him. Furthermore, some sense organs move relative to the body skeleton, such as our eyes rotating within their sockets. Each sense organ has a configuration space, which corresponds to all possible ways it can be transformed or configured. The most important aspect of this is the...
2.1. HARDWARE

Figure 2.2: Under normal conditions, the brain (and body parts) control the configuration of sense organs (eyes, ears, fingertips) as they receive natural stimulation from the surrounding, physical world.

Figure 2.3: In comparison to Figure 2.2, a VR system “hijacks” each sense by replacing the natural stimulation with artificial stimulation that is provided by hardware called a display. Using a computer, a virtual world generator maintains a coherent, virtual world. Appropriate “views” of this virtual world are rendered to the display.

number of degrees of freedom or DOFs of the sense organ. Chapter 3 will cover this thoroughly, but for now note that a rigid object that moves through ordinary space has six DOFs. Three DOFs correspond to its changing position in space: 1) side-to-side motion, 2) vertical motion, and 3) closer-further motion. The other three DOFs correspond to possible ways the object could be rotated; in other words, exactly three independent parameters are needed to specify how the object is oriented. These are called yaw, pitch, and roll, and are covered in Section 3.2.

As an example, consider your left ear. As you rotate your head or move your body through space, the position of the ear changes, as well as its orientation. This yields six DOFs. The same is true for your right eye. Keep in mind that our bodies have many more degrees of freedom, which affect the configuration of our sense organs. A tracking system may be necessary to determine the position and orientation of each sense organ that receives artificial stimuli, which will be explained shortly.

Figure 2.4: If done well, the brain is “fooled” into believing that the virtual world is in fact the surrounding physical world and natural stimulation is resulting from it.
2.1. HARDWARE

An abstract view Figure 2.2 illustrates the normal operation of one of our sense organs without interference from VR hardware. The brain controls its configuration, while the sense organ converts natural stimulation from the environment into neural impulses that are sent to the brain. Figure 2.2 shows how it appears in a VR system. The VR hardware contains several components that will be discussed shortly. A Virtual World Generator (VWG) runs on a computer and produces “another world”, which could be many possibilities, such as a pure simulation of a synthetic world, a recording of the real world, or a live connection to another part of the real world. The human perceives the virtual world through each targeted sense organ using a display, which emits energy that is specifically designed to mimic the type of stimulus that would appear without VR. The process of converting information from the VWG into output for the display is called rendering. In the case of human eyes, the display might be a smartphone screen or the screen of a video projector. In the case of ears, the display is referred to as a speaker. (A display need not be visual, even though this is the common usage in everyday life.) If the VR system is effective, then the brain is hopefully “fooled” in the sense shown in Figure 2.4. The user should believe that the stimulation of the senses is natural and comes from a plausible world, being consistent with at least some past experiences.

Aural: world-fixed vs. user-fixed Recall from Section 1.3 the trend of having to go somewhere for an experience, to having it in the home, and then finally to having it be completely portable. To understand these choices for VR systems and their implications on technology, it will be helpful to compare a simpler case: Audio or aural systems.

Figure 2.5 shows the speaker setup and listener location for a Dolby 7.1 Surround Sound theater system, which could be installed at a theater or a home family room. Seven speakers distributed around the room periphery generate most of the sound, while a subwoofer (the “1” of the “7.1") delivers the lowest frequency components. The aural displays are therefore world-fixed. Compare this to a listener wearing headphones, as shown in Figure 2.6. In this case, the aural displays are user-fixed. Hopefully, you have already experienced settings similar to these many times.

What are the key differences? In addition to the obvious portability of headphones the following quickly come to mind:

- In the surround-sound system, the generated sound (or stimulus) is far away from the ears, whereas it is quite close for the headphones.
- One implication of the difference in distance is that much less power is needed for the headphones to generate an equivalent perceived loudness level compared with distant speakers.
- Another implication based on distance is the degree of privacy allowed by the wearer of headphones. A surround-sound system at high volume levels could generate a visit by angry neighbors.

Figure 2.5: In a surround-sound system, the aural displays (speakers) are world-fixed while the user listens from the center.

Figure 2.6: Using headphones, the displays are user-fixed, unlike the case of a surround-sound system.
• Wearing electronics on your head could be uncomfortable over long periods of time, causing a preference for surround sound over headphones.

• Several people can enjoy the same experience in a surround-sound system (although they cannot all sit in the optimal location). Using headphones, they would need to split the audio source across their individual headphones simultaneously.

• They are likely to have different costs, depending on the manufacturing costs and available component technology. At present, headphones are favored by costing much less than a set of surround-sound speakers (although one can spend a large amount of money on either).

All of these differences carry over to VR systems. This should not be too surprising because we could easily consider a pure audio experience to be a special kind of VR experience based on our definition from Section 1.1.

While listening to music, close your eyes and imagine you are at a live performance with the artists surrounding you. Where do you perceive the artists and their instruments to be located? Are they surrounding you, or do they seem to be in the middle of your head? Using headphones, it is most likely that they seem to be inside your head. In a surround-sound system, if recorded and displayed properly, the sounds should seem to be coming from their original locations well outside of your head. They probably seem constrained, however, into the horizontal plane that you are sitting in.

This shortcoming of headphones is not widely recognized at present, but nevertheless represents a problem that becomes much larger for VR systems that include visual displays. If you want to preserve your perception of where sounds are coming from, then headphones would need to take into account the configuration of your ears in space so that audio is adjusted accordingly. For example, if you nod your head back and forth in a "no" gesture, then the sound being presented to each ear needs to be adjusted so that the simulated sound source is rotated in the opposite direction. In the surround-sound system, the speaker does not follow your head and therefore does not need to rotate. If the speaker rotates with your head, then a counter-rotation is needed to "undo" your head rotation so that the sound source location is perceived to be stationary.

Visual: world-fixed vs. user-fixed Now consider adding a visual display. You might not worry much about the perceived location of artists and instruments while listening to music, but you will quickly notice if their locations do not appear correct to your eyes. Our vision sense is much more powerful and complex than our sense of hearing. Figure 2.7(a) shows a CAVE system, which parallels the surround-sound system in many ways. The user again sits in the center while displays around the periphery present visual stimuli to your eyes. The speakers are replaced by video screens. Figure 2.7(b) shows a user wearing a VR headset, which parallels the headphones.

Figure 2.7: (a) A CAVE VR system developed at Teesside University, UK. (b) A woman wearing the first-generation Oculus Rift headset.

Suppose the screen in front of the user’s eyes shows a fixed image in the headset. If the user rotates his head, then the image will be perceived as being attached to the head. This would occur, for example, if you rotate your head while using the Viewmaster (recall Figure 1.25(b)). If you would like to instead perceive the image as part of a fixed world around you, then the image inside the headset must change to compensate as you rotate your head. The surrounding virtual world should be counter-rotated, the meaning of which should become more precise after reading Section 3.4. Once we agree that such transformations are necessary, it becomes a significant engineering challenge to estimate the amount of head and eye movement that has occurred and apply the appropriate transformation in a timely and accurate manner. If this is not handled well, then the user could have a poor or unconvincing experience. Worse yet, they could fall prey to VR sickness. This is one of the main reasons why the popularity of VR headsets waned in the 1990s. The component technology was not good enough yet. Fortunately, the situation is much improved at present. For audio, few seemed to bother with this transformation, but for the visual counterpart, it is absolutely critical. One final note is that tracking and applying transformations also becomes necessary in CAVE systems if we want the images on the screens to be altered according to changes in the eye positions inside of the room.

Now that you have a high-level understanding of the common hardware arrangements, we will take a closer look at hardware components that are widely available for constructing VR systems. These are expected to change quickly, with costs decreasing and performance improving. We also expect many new devices to appear in the marketplace in the coming years. In spite of this, the fundamentals in this book remain unchanged. Knowledge of the current technology provides concrete examples to make the fundamental VR concepts clearer.

The hardware components of VR systems are conveniently classified as:

• Displays (output): Devices that stimulate a sense organ.
2.1. HARDWARE

(a) (b)

Figure 2.8: Two examples of haptic feedback devices. (a) The Geomagic Phantom allows the user to feel strong resistance when poking into a virtual object with a real stylus. A robot arm provides the appropriate forces. (b) Some game controllers occasionally vibrate.

- **Sensors (input):** Devices that extract information from the real world.
- **Computers:** Devices that process inputs and outputs sequentially.

**Displays**

The purpose of a display is to generate a stimulus for a targeted sense organ. Vision is our dominant sense, and any display constructed for the eye must cause the desired image to be formed on the retina. Because of this importance, Chapters 4 and 5 will explain optical systems and the human vision system, respectively. For CAVE systems, some combination of digital projectors and mirrors are used. Due to the plummeting costs, an array of large-panel displays may alternatively be employed. For headsets, a smartphone display can be placed close to the eyes and brought into focus using one magnifying lens for each eye. Screen manufacturers are currently making custom displays for VR headsets by leveraging the latest LED display technology from the smartphone industry. Some are targeting one display per eye with frame rates above 90Hz and over two megapixels per eye. Reasons for this are explained in Chapter 5. More exotic displays, which are primarily in a research-and-development stage include pico projectors \[\text{?}\] \[\text{?}\], light-field displays \[33, 43\], and multi-focal-plane optical systems \[2\].

Now imagine displays for other sense organs. Sound is displayed to the ears using classic speaker technology. Bone conduction methods may also be used, which vibrate the skull and propagate the waves to the inner ear; this method appeared recently in Google Glass \[\text{[2]}\]. Chapter 12 covers the auditory part of VR in detail. For visual and auditory body-mounted displays, the position and orientation of the sense organ must be tracked by sensors to appropriately adapt the stimulus. The orientation part is usually accomplished by an inertial measurement unit or IMU. The main component is a gyroscope, which measures its own rate of rotation; the rate is referred to as angular velocity and has three components. Measurements from the gyroscope are integrated over time to obtain an estimate of the cumulative change in orientation. The resulting error, called drift error, would gradually grow unless other sensors are used. To reduce drift error, IMUs also contain an accelerometer and possibly a magnetometer. Over the years, IMUs have gone from existing only as large mechanical systems in aircraft and missiles to being tiny devices inside of our smartphones; see Figure 2.9. Due to their small size, weight, and cost, IMUs can be easily embedded in wearable devices. They are one of the most important enabling technologies for the current generation of VR headsets and are mainly used for tracking the user’s head orientation.

Digital cameras provide another critical source of information for tracking systems. Like IMUs, they have become increasingly cheap and portable due to the smartphone industry, while at the same time improving in image quality. Cameras enable tracking approaches that exploit line-of-sight visibility. The idea is to place markers on the object to be tracked and find them in the image. Such visibility constraints severely limit the possible object positions and orientations. Standard cameras passively form an image focusing the light through an optical system, much like the human eye. Once the camera calibration parameters are known, an observed marker is known to lie along a ray in space. Cameras are commonly...
2.1 HARDWARE

Figure 2.10: (a) The Microsoft Kinect sensor gathers both an ordinary RGB image and a depth map (the distance away from the sensor for each pixel). (b) The depth is determined by observing the locations of projected IR dots in an image obtained from an IR camera.

used to track eyes, heads, hands, entire human bodies, and any other objects in the physical world. One of the main difficulties at present is to obtain reliable and accurate performance without placing special markers on the user or objects around the scene.

As opposed to standard cameras, depth cameras work actively by projecting light into the scene and then observing its reflection in the image. This is typically done in the infrared (IR) spectrum so that humans do not notice; see Figure 2.10.

In addition to these sensors, we rely heavily on good-old mechanical switches and potentiometers to create keyboards and game controllers. An optical mouse is also commonly used. One advantage of these familiar devices is that users can rapidly input data or control their characters by leveraging their existing training. A disadvantage is that they might be hard to find or interact with if their faces are covered by a headset.

Computers A computer executes the virtual world generator (VWG). Where should this computer be? Although unimportant for world-fixed displays, the location is crucial for body-fixed displays. If a separate PC is needed to power the system, then fast, reliable communication must be provided between the headset and the PC. This connection is currently made by wires, leading to an awkward tether; current wireless speeds are not sufficient. As you have noticed, most of the needed sensors exist on a smartphone, as well as a moderately powerful computer. Therefore, a smartphone can be dropped into a case with lenses to provide a VR experience with little added costs (Figure 2.11). The limitation, though, is that the VWG must be simpler than in the case of a separate PC so that it runs on less-powerful computing hardware.

In addition to the main computing systems, specialized computing hardware may be utilized. Graphical processing units (GPUs) have been optimized for quickly rendering graphics to a screen and they are currently being adapted to handle the specific performance demands of VR. Also, a display interface chip converts an input video into display commands. Finally, microcontrollers are frequently used to gather information from sensing devices and send them to the main computer using standard protocols, such as USB.

To conclude with hardware, Figure 2.12 shows the hardware components for the Oculus Rift DK2, which became available in late 2014. In the lower left corner, you can see a smartphone screen that serves as the display. Above that is a circuit board that contains the IMU, display interface chip, a USB driver chip, a set of chips for driving LEDs on the headset for tracking, and a programmable microcontroller. The lenses, shown in the lower right, are placed so that the smartphone screen appears to be “infinitely far” away, but nevertheless fills most of the field of view of the user. The upper right shows flexible circuits that deliver power to IR LEDs embedded in the headset (they are hidden behind IR-transparent plastic). A camera is used for tracking, and its parts are shown in the center.

2.2 Software

From a developer’s standpoint, it would be ideal to program the VR system by providing high-level descriptions and having the software determine automatically all of the low-level details. In a perfect world, there would be a VR engine, which serves a purpose similar to the game engines available today for creating video games. If the developer follows patterns that many before him have implemented already, then many complicated details can be avoided by simply calling functions from a well-designed software library. However, if the developer wants to try something relatively original, then she would have to design the functions from scratch. This requires a deeper understanding of the VR fundamentals, while also
being familiar with lower-level system operations.

Unfortunately, we are currently a long way from having fully functional, general-purpose VR engines. As applications of VR broaden, specialized VR engines are also likely to emerge. For example, one might be targeted for immersive cinematography while another is geared toward engineering design. Which components will become more like part of a VR “operating system” and which will become higher level “engine” components? Given the current situation, developers will likely be implementing much of the functionality of their VR systems from scratch. This may involve utilizing a software development kit (SDK) for particular headsets that handles the lowest level operations, such as device drivers, head tracking, and display output. Alternatively, they might find themselves using a game engine that has been recently adapted for VR, even though it was fundamentally designed for video games on a screen. This can avoid substantial effort at first, but then may be cumbersome when someone wants to implement ideas that are not part of standard video games.

What software components are needed to produce a VR experience? Figure 2.13 presents a high-level view that highlights the central role of the Virtual World Generator (VWG). The VWG receives inputs from low-level systems that indicate what the user is doing in the real world. A head tracker provides timely estimates of the user’s head position and orientation. Keyboard, mouse, and game controller events arrive in a queue that are ready to be processed. The key role of the VWG is to maintain enough of an internal “reality” so that renderers can extract the information they need to calculate outputs for their displays.

**Virtual world: real vs. synthetic** At one extreme, the virtual world could be completely synthetic. In this case, numerous triangles are defined in a 3D space, along with material properties that indicate how they interact with light, sound, forces, and so on. The field of **computer graphics** addresses computer-generated images from synthetic models [1], and it remains important for VR; see Chapter 7. At the other extreme, the virtual world might be a recorded physical world that was captured using modern cameras, computer vision, and Simultaneous Localization and Mapping (SLAM) techniques; Figure 2.14. Many possibilities exist between the extremes. For example, camera images may be taken of a real object, and then mapped onto a synthetic object in the virtual world. This is called **texture mapping**, a common operation in computer graphics; see Section 7.2.

**Matched motion** The most basic operation of the VWG is to maintain a correspondence between user motions in the real world and the virtual world; see Figure 2.15. In the real world, the user’s motions are confined to a safe region, which we will call the **matched zone**. Imagine the matched zone as a place where the real and virtual worlds perfectly align. One of the greatest challenges is the mismatch...
2.2 SOFTWARE

Figure 2.14: Using both color and depth information from cameras, a 3D model of the world can be extracted automatically using Simultaneous Localization and Mapping (SLAM) techniques. Figure from [27].

of obstacles: What if the user is blocked in the virtual world but not in the real world? The reverse is also possible. In a seated experience, the user sits in a chair while wearing a headset. The matched zone in this case is a small region, such as one cubic meter, in which users can move their heads. Head motions should be matched between the two worlds. If the user is not constrained to a seat, then the matched zone could be an entire room or an outdoor field. Note that safety becomes an issue because the user might spill a drink, hit walls, or fall into pits that exist only in the real world, but are not visible in the virtual world. Larger matched zones tend to lead to greater safety issues. Users must make sure that the matched zone is cleared of dangers in the real world, or the developer should make them visible in the virtual world.

Which motions from the real world should be reflected in the virtual world? This varies among VR experiences. In a VR headset that displays images to the eyes, head motions must be matched so that the visual renderer uses the correct viewpoint in the virtual world. Other parts of the body are less critical, but may become important if the user needs to perform hand-eye coordination or looks at other parts of her body and expects them to move naturally.

Figure 2.15: A matched zone is maintained between the user in their real world and his representation in the virtual world. The matched zone could be moved in the virtual world by using an interface, such as a game controller, while the user does not correspondingly move in the real world.

Locomotion In many VR experiences, users want to move well outside of the matched zone. This motivates locomotion, which means moving oneself in the virtual world, while this motion is not matched in the real world. Imagine you want to explore a virtual city while remaining seated in the real world. How should this be achieved? You could pull up a map and point to where you want to go, with a quick teleportation operation sending you to the destination. A popular option is to move oneself in the virtual world by operating a game controller, mouse, or keyboard. By pressing buttons or moving knobs, your self in the virtual world could be walking, running, jumping, swimming, flying, and so on. You could also climb aboard a vehicle in the virtual world and operate its controls to move yourself. These operations are certainly convenient, but often lead to sickness because of a mismatch between your vestibular and visual senses. See Sections 2.3 and 10.1.

Physics The VWG handles the geometric aspects of motion by applying the appropriate mathematical transformations. In addition, the VWG usually implements some physics so that as time progresses, the virtual world behaves like the real world. In most cases, the basic laws of mechanics should govern how objects move in the virtual world. For example, if you drop an object, it should accelerate to the ground due to gravitational force acting on it. One important component is a collision detection algorithm, which determines whether two or more bodies are intersecting in the virtual world. If a new collision occurs, then an appropriate response is needed. For example, suppose the user pokes his head through a wall in the virtual world. Should the head in the virtual world be stopped, even though it continues to move in the real world? To make it more complex, what should happen if you unload a dump truck full of basketballs into a busy street in the
virtual world? Simulated physics can become quite challenging, and is a discipline in itself. There is no limit to the complexity.

In addition to handling the motions of moving objects, the physics must also take into account how potential stimuli for the displays are created and propagate through the virtual world. How does light propagate through the environment? How does light interact with the surfaces in the virtual world? What are the sources of light? How do sound and smells propagate? These correspond to rendering problems, which are covered in Chapters 7 and 8 for visual and audio cases.

Networked experiences In the case of a networked VR experience, a shared virtual world is maintained by a server. Each user has a distinct matched zone. Their matched zones might overlap in a real world, but one must then be careful so that they avoid unwanted collisions. Most often, these zones are disjoint and distributed around the Earth. Within the virtual world, user interactions, including collisions, must be managed by the VWG. If multiple users are interacting in a social setting, then the burdens of matched motions may increase. As users see each other, they could expect to see eye motions, facial expressions, and body language; see Section 11.2.

Developer choices for VWGs To summarize, a developer could start with a basic Software Development Kit (SDK) from a VR headset vendor and then build her own VWG from scratch. The SDK should provide the basic drivers and an interface to access tracking data and make calls to the graphical rendering libraries. In this case, the developer must build the physics of the virtual world from scratch, handling problems such as avatar movement, collision detection, lighting models, and audio. This gives the developer the greatest amount of control and ability to optimize performance; however, it may come in exchange for a difficult implementation burden. In some special cases, it might not be too difficult. For example, in the case of the Google Street viewer (recall Figure 1.8), the “physics” is simple: The viewing location needs to jump between panoramic images in a comfortable way while maintaining a sense of location on the Earth. In the case of telepresence using a robot, the VWG would have to take into account movements in the physical world. Failure to handle collision detection could result in a broken robot (or human!).

At the other extreme, a developer may use a ready-made VWG that is customized to make a particular VR experience by choosing menu options and writing high-level scripts. Examples available today are OpenSimulator, Vizard by WorldViz, Unity 3D, and Unreal Engine by Epic Games. The latter two are game engines that were adapted to work for VR, and are by far the most popular among current VR developers. The first one, OpenSimulator, was designed as an open-source alternative to Second Life for building a virtual society of avatars. Using such higher-level engines make it easy for developers to make a VR experience in little time; however, the drawback is that it is harder to make unusual experiences that were not imagined by the engine builders.

2.3 Human Physiology and Perception

Our bodies were not designed for VR. By applying artificial stimulation to the senses, we are disrupting the operation of biological mechanisms that have taken hundreds of millions of years to evolve in a natural environment. We are also providing input to the brain that is not exactly consistent with all of our other life experiences. In some instances, our bodies may adapt to the new stimuli. This could cause us to become unaware of flaws in the VR system. In other cases, we might develop heightened awareness or the ability to interpret 3D scenes that were once difficult or ambiguous. Unfortunately, there are also many cases where our bodies react by increased fatigue or headaches, partly because the brain is working harder than usual to interpret the stimuli. Finally, the worst case is the onset of VR sickness, which typically involves symptoms of dizziness and nausea.

Perceptual psychology is the science of understanding how the brain converts sensory stimulation into perceived phenomena. Here are some typical questions that arise in VR and fall under this umbrella:

- How far away does that object appear to be?
- How much video resolution is needed to avoid seeing pixels?
- How many frames per second are enough to perceive motion as continuous?
- Is the user’s head appearing at the proper height in the virtual world?
- Where is that virtual sound coming from?
- Why am I feeling nauseated?
- Why is one experience more tiring than another?
- What is presence?

To answer these questions and more, we must understand several things: 1) basic physiology of the human body, including sense organs and neural pathways, 2) the key theories and insights of experimental perceptual psychology, and 3) the interference of the engineered VR system with our common perceptual processes and the resulting implications or side-effects.

The perceptual side of VR often attracts far too little attention among developers. In the real world, perceptual processes are mostly invisible to us. Think about how much effort it requires to recognize a family member. When you see someone you know well, the process starts automatically, finishes immediately, and seems to require no effort. Scientists have conducted experiments that reveal how much work actually occurs in this and other perceptual processes. Through brain lesion studies, they are able to see the effects when a small part of the brain...
2.3. HUMAN PHYSIOLOGY AND PERCEPTION

S. M. LaValle: Virtual Reality

Figure 2.16: Optical illusions present an unusual stimulus that highlights limitations of our vision system. (a) The Ponzo illusion causes the upper line segment to appear larger than the lower one, even though they are the same length. (b) The checker shadow illusion causes the B tile to appear lighter than the A tile, even though they are the exactly the same shade of gray (figure by Adrian Pingstone).

is not functioning correctly. Some people suffer from prosopagnosia, which makes them unable to recognize the faces of familiar people, including themselves in a mirror, even though nearly everything else functions normally. Scientists are also able to perform single-unit recordings, mostly on animals, which reveal the firings of a single neuron in response to sensory stimuli. Imagine, for example, a single neuron that fires whenever you see a sphere.

Optical illusions One of the most popular ways to appreciate the complexity of our perceptual processing is to view optical illusions. These yield surprising results and are completely unobtrusive. Each one is designed to reveal some shortcoming of our visual system by providing a stimulus that is not quite consistent with ordinary stimuli in our everyday lives. Figure 2.16 shows two. These should motivate you to appreciate the amount of work that our sense organs and neural structures are doing to fill in missing details and make interpretations based on the context of our life experiences and existing biological structures. Interfering with these without understanding them is not wise!

Classification of senses Perception and illusions are not limited to our eyes. Figure 2.17 shows a classification of our basic senses. Recall that a sensor converts an energy source into signals in a circuit. In the case of our bodies, this means that a stimulus is converted into neural impulses. For each sense, Figure 2.17 indicates the type of energy for the stimulus and the receptor that converts the stimulus into neural impulses. Think of each receptor as a sensor that targets a particular kind of stimulus. This is referred to as sensory system selectivity. In each

eye, over 100 million photoreceptors target electromagnetic energy precisely in the frequency range of visible light. Different kinds even target various colors and light levels; see Section 5.1. The auditory, touch, and balance senses involve motion, vibration, or gravitational force; these are sensed by mechanoreceptors. The sense of touch additionally involves thermoreceptors to detect change in temperature. Our balance sense helps us to know which way our head is oriented, including sensing the direction of “up”. Finally, our sense of taste and smell is grouped into one category that relies on chemoreceptors, which provide signals based on chemical composition of matter appearing on our tongue or in our nasal passages.

Note that senses have engineering equivalents, most of which appear in VR systems. Imagine you are designing a humanoid telepresence robot, which you expect to interface with through a VR headset. You could then experience life through your surrogate robotic self. Digital cameras would serve as its eyes, and microphones would be the ears. Pressure sensors and thermometers could be installed to give a sense of touch. For balance, we can install an IMU. In fact, the human vestibular organs and modern IMUs bear a striking resemblance in terms of the signals they produce; see Section 8.3. We could even install chemical sensors, such as a pH meter, to measure aspects of chemical composition to provide taste and smell.

Big brains Perception happens after the sense organs convert the stimuli into neural impulses. According to latest estimates, human bodies contain around 86 billion neurons. Around 20 billion are devoted to the part of the brain called the cerebral cortex, which handles perception and many other high-level functions such as attention, memory, language, and consciousness. It is a large sheet of neurons around three millimeters thick and is heavily folded so that it fits into our skulls. In case you are wondering where we lie among other animals, a roundworm, fruit fly, and rat have 302, 100 thousand, and 200 million neurons, respectively. A gorilla has around 4 billion. A type of dolphin called the long-finned pilot whale has an estimated 37 billion neurons in its cerebral cortex, making it roughly twice as many as in the human cerebral cortex; however, scientists claim this does not
2.3. HUMAN PHYSIOLOGY AND PERCEPTION

Various perceptual phenomena occur, such as recognizing a face or identifying a song. Information or concepts that appear in the cerebral cortex tend to represent a global picture of the world around us. Surprisingly, topographic mapping methods reveal that spatial relationships among receptors are maintained in some cases among the distribution of neurons. Also, recall from Section 1.1 that place cells and grid cells encode spatial maps of familiar environments.

Fusion of senses

Signals from multiple senses and proprioception are being processed and combined with our experiences by our neural structures throughout our lives. In ordinary life, without VR or drugs, our brains interpret these combinations of inputs in coherent, consistent, and familiar ways. Any attempt to interfere with these operations is likely to cause a mismatch among the data from our senses. The brain may react in a variety of ways. It could be the case that we are not consciously aware of the conflict, but we may become fatigued or develop a headache. Even worse, we could develop symptoms of dizziness or nausea. In other cases, the brain might react by making us so consciously aware of the conflict that we immediately understand that the experience is artificial. This would correspond to a case in which the VR experience is failing to convince someone that they are present in a virtual world. To make an effective and comfortable VR experience, trials with human subjects are essential to understand how the brain reacts. It is practically impossible to predict what would happen in an unknown scenario, unless it is almost identical to other well-studied scenarios.

One of the most important examples of bad sensory conflict in the context of VR isvection, which is the illusion of self motion. The conflict arises when your vision sense reports to your brain that you are accelerating, but your balance sense reports that you are motionless. As you walk down the street, the balance and
vision senses are in harmony. You might have experienced vection before, even without VR. If you are stuck in traffic or stopped at a train station, you might have felt as if you are moving backwards while seeing a vehicle in your periphery that is moving forward. In the 1890s, Amariah Lake constructed an amusement park ride that consisted of a swing that remains at rest while the entire room surrounding the swing rocks back-and-forth (Figure 2.20). In VR, vection is caused by the locomotion operation described in Section 2.2. For example, if you accelerate yourself forward using a controller, rather than moving forward in the real world, then you perceive acceleration with your eyes, but not your vestibular organ. For strategies to alleviate this problem, see Section 10.1.

**Adaptation** A universal feature of our sensory systems is adaptation, which means that the perceived effect of stimuli changes over time. This may happen with any of our senses and over a wide spectrum of time intervals. For example, the perceived loudness of motor noise in an aircraft or car decreases within minutes. In the case of vision, the optical system of our eyes and the photoreceptor sensitivities adapt to change perceived brightness. Over long periods of time, perceptual training can lead to adaptation. In military training simulations, sickness experienced by soldiers is greatly reduced by regular exposure. Anecdotally, the same seems to be true of experienced video game players. Those who have spent many hours and days in front of large screens playing first-person shooter games apparently experience less vection when locomoting themselves in VR.

Adaptation therefore becomes a crucial factor for VR. Through repeated exposure, developers may become comfortable with an experience that is nauseating to a newcomer. This gives them a terrible bias while developing an experience; recall from Section 1.1 the problem of confusing the scientist with the lab subject in the VR experiment. On the other hand, through repeated, targeted training they may be able to improve their debugging skills by noticing flaws in the system that an “untrained eye” would easily miss. Common examples include:

- A large amount of tracking latency has appeared, which interferes with the perception of stationarity.
- The left and right eye views are swapped.
- Objects appear to one eye but not the other.
- One eye view has significantly more latency than the other.
- Straight lines are slightly curved due to uncorrected warping in the optical system.

This disconnect between the actual stimulus and your perception of the stimulus leads to the next topic.
2.3. HUMAN PHYSIOLOGY AND PERCEPTION

Figure 2.21: The most basic psychometric function. For this example, as the stimulus intensity is increased, the percentage of people detecting the phenomenon increases. The point along the curve that corresponds to 50 percent indicates a critical threshold or boundary in the stimulus intensity.

Psychophysics  
Psychophysics is the scientific study of perceptual phenomena that are produced by physical stimuli. For example, under what conditions would someone call an object “red”? The stimulus corresponds to light entering the eye, and the perceptual phenomenon is the concept of “red” forming in the brain. Other examples of perceptual phenomena are “straight”, “larger”, “louder”, “tickles”, and “sour”. Figure 2.21 shows a typical scenario in a psychophysical experiment. As one parameter is varied, such as the frequency of a light, there is usually a range of values for which subjects cannot reliably classify the phenomenon. For example, there may be a region where they are not sure whether the light is red. At one extreme, they may consistently classify it as “red” and at the other extreme, they consistently classify it as “not red”. For the region in between, the probability of detection is recorded, which corresponds to the frequency with which it is classified as “red”.

Stevens’ power law  
One of the most known results from psychophysics is Stevens’s power law, which characterizes the relationship between the magnitude of a physical stimulus and its perceived magnitude. The hypothesis is that an exponential relationship occurs over a wide range of sensory systems and stimuli:

\[ p = cm^x \]  

in which

- \( m \) is the magnitude or intensity of the stimulus,
- \( p \) is the perceived magnitude,
- \( x \) relates the actual magnitude to the perceived magnitude, and is the most important part of the equation, and
- \( c \) is an uninteresting constant that depends on units.

Stevens’s power law captures the relationship between the magnitude of a stimulus and its perceived magnitude. The model is an exponential curve, and the exponent depends on the stimulus type.

Figure 2.22: Stevens’s power law captures the relationship between the magnitude of a stimulus and its perceived magnitude. The model is an exponential curve, and the exponent depends on the stimulus type.

- \( p \) is the perceived magnitude,
- \( x \) relates the actual magnitude to the perceived magnitude, and is the most important part of the equation, and
- \( c \) is an uninteresting constant that depends on units.

Note that for \( x = 1 \), it is a linear relationship, \( p = cm \); see Figure 2.22. An example of this is our perception of the length of an isolated line segment directly in front of our eyes. The length we perceive is proportional to its actual length. The more interesting cases are when \( x \neq 1 \). For the case of perceiving the brightness of a target in the dark, \( x = 0.33 \), which implies that a large increase in brightness is perceived as a smaller increase. In the other direction, our perception of electric shock as current through the fingers yields \( x = 3.5 \). A little more shock is a lot more uncomfortable!

Just noticeable difference  
Another key psychophysical concept is the Just Noticeable Difference (JND). This is the amount that the stimulus needs to be changed so that subjects would perceive it to have changed in at least 50 percent of trials. For a large change, all or nearly all subjects would report a change. If the change is too small, then none or nearly none of the subjects would notice. The experimental challenge is to vary the amount of change until the chance of someone reporting a change is 50 percent.

Consider the JND for a stimulus with varying magnitude, such as brightness. How does the JND itself vary as the magnitude varies? This relationship is cap-
tured by Weber’s law:
\[
\frac{\Delta m}{m} = c,
\]
(2.2)
in which \(\Delta m\) is the JND, \(m\) is the magnitude of the stimulus, and \(c\) is a constant.

**Design of experiments**  VR disrupts the ordinary perceptual processes of its users. It should be clear from this section that proposed VR systems and experiences need to be evaluated on users to understand whether they are yielding the desired effect while also avoiding unwanted side effects. This amounts to applying the scientific method to make observations, formulate hypotheses, and design experiments that determine their validity. When human subjects are involved, this becomes extremely challenging. How many subjects are enough? What happens if they adapt to the experiment? How does their prior world experience affect the experiment? What if they are slightly sick the day that they try the experiment? What did they eat for breakfast? The answers to these questions could dramatically affect the outcome.

It gets worse. Suppose they already know your hypothesis going into the experiment. This will most likely bias their responses. Also, what will the data from the experiment look like? Will you ask them to fill out a questionnaire, or will you make inferences about their experience from measured data such as head motions, heart rate, and skin conductance? These choices are also critical. See Section 11.3 for more on this topic.

**Further Reading**
VR sickness survey paper: [29]
Dynamical simulation literature:
More neuroscience: [61]
Chapter 3
The Geometry of Virtual Worlds

Section 2.2 introduced the Virtual World Generator (VWG), which maintains the geometry and physics of the virtual world. This chapter covers the geometry part, which is needed to make models and move them around. The models could include the walls of a building, furniture, clouds in the sky, the user’s avatar, and so on. Section 3.1 covers the basics of how to define consistent, useful models. Section 3.2 explains how to apply mathematical transforms that move them around in the virtual world. This involves two components: Translation (changing position) and rotation (changing orientation). Section 3.3 presents the best ways to express and manipulate 3D rotations, which are the most complicated part of moving models. Section 3.4 then covers how the virtual world appears if we try to “look” at it from a particular perspective. This is the geometric component of visual rendering, which is covered in Chapter 4. Finally, Section 3.5 puts all of the transformations together, so that you can see how to go from defining a model to having it appear in the right place on the display.

If you work with high-level engines to build a VR experience, then most of the concepts from this chapter might not seem necessary. You might need only to select options from menus and write simple scripts. However, an understanding of the basic transformations, such as how to express 3D rotations or move a camera viewpoint, is essential to making the software do what you want. Furthermore, if you want to build virtual worlds from scratch, or at least want to understand what is going on under the hood of a software engine, then this chapter is critical.

Figure 3.1: Points in the virtual world are given coordinates in a right-handed coordinate system in which the y axis is pointing upward. The origin (0, 0, 0) lies at the point where axes intersect. Also shown is a 3D triangle defined by its three vertices, each of which is a point in $\mathbb{R}^3$.

3.1 Geometric Models

We first need a virtual world to contain the geometric models. For our purposes, it is enough to have a 3D Euclidean space with Cartesian coordinates. Therefore, let $\mathbb{R}^3$ denote the virtual world, in which every point is represented as a triple of real-valued coordinates: $(x, y, z)$. The coordinate axes of our virtual world are shown in Figure 3.1. We will consistently use right-handed coordinate systems in this book because they represent the predominant choice throughout physics and engineering; however, left-handed systems appear in some places, with the most notable being Microsoft’s DirectX graphical rendering library. In these cases, one of the three axes points in the opposite direction in comparison to its direction in a right-handed system. This inconsistency can lead to hours of madness when writing software; therefore, be aware of the differences and their required conversions if you mix software or models that use both. If possible, avoid mixing right-handed and left-handed systems altogether.

Geometric models are made of surfaces or solid regions in $\mathbb{R}^3$ and contain an infinite number of points. Because representations in a computer must be finite, models are defined in terms of primitives in which each represents an infinite set of points. The simplest and most useful primitive is a 3D triangle, as shown in Figure 3.1. A planar surface patch that corresponds to all points “inside” and on the boundary of the triangle is fully specified by the coordinates of the triangle vertices:

$$((x_1, y_1, z_1), (x_2, y_2, z_2), (x_3, y_3, z_3)).$$

(3.1)

To model a complicated object or body in the virtual world, numerous triangles can be arranged into a mesh, as shown in Figure 3.2. This provokes many
3.1 GEOMETRIC MODELS

Figure 3.2: A geometric model of a dolphin, formed from a mesh of 3D triangles (from Wikipedia user Chrschn).

important questions:

1. How do we specify how each triangle “looks” whenever viewed by a user in VR?

2. How do we make the object “move”? 

3. If the object surface is sharply curved, then should we use curved primitives, rather than trying to approximate the curved object with tiny triangular patches?

4. Is the interior of the object part of the model, or is it represented only by its surface?

5. Is there an efficient algorithm for determining which triangles are adjacent to a given triangle along the surface?

6. Should we avoid duplicating vertex coordinates that are common to many neighboring triangles?

We address these questions in reverse order.

Data structures Consider listing all of the triangles in a file or memory array. If the triangles form a mesh, then most or all vertices will be shared among multiple triangles. This is clearly a waste of space. Another issue is that we will frequently want to perform operations on the model. For example, after moving an object, can we determine whether it is in collision with another object (see Section 8.2)? A typical low-level task might be to determine which triangles share a common vertex or edge with a given triangle. This might require linearly searching through the triangle list to determine whether they share a vertex or two. If there are millions of triangles, which is not uncommon, then it would cost too much to perform this operation repeatedly.

For these reasons and more, geometric models are usually encoded in clever data structures. The choice of the data structure should depend on which operations will be performed on the model. One of the most useful and common is the doubly connected edge list, also known as half-edge data structure [15, 53]. See Figure 3.3. In this and similar data structures, there are three kinds of data elements: faces, edges, and vertices. These represent two, one, and zero-dimensional parts, respectively, of the model. In our case, every face element represents a triangle. Each edge represents the border of one or two, without duplication. Each vertex is shared between one or more triangles, again without duplication. The data structure contains pointers between adjacent faces, edges, and vertices so that algorithms can quickly traverse the model components in a way that corresponds to how they are connected together.

Inside vs. outside Now consider the question of whether the object interior is part of the model (recall Figure 3.2). Suppose the mesh triangles fit together perfectly so that every edge borders exactly two triangles and no triangles intersect unless they are adjacent along the surface. In this case, the model forms a complete barrier between the inside and outside of the object. If we were to hypothetically fill the inside with a gas, then it could not leak to the outside. This is an example of a coherent model. Such models are required if the notion of inside or outside is critical to the VWG. For example, a penny could be inside of the dolphin, but not intersecting with any of its boundary triangles. Would this ever need to be detected? If we remove a single triangle, then the hypothetical gas would leak
3.1 GEOMETRIC MODELS

There would no longer be a clear distinction between the inside and outside of the object, making it difficult to answer the question about the penny and the dolphin. In the extreme case, we could have a single triangle in space. There is clearly no natural inside or outside. Furthermore, the model could be as bad as polygon soup, which is a jumble of triangles that do not fit together nicely and could even have intersecting interiors. In conclusion, be careful when constructing models so that the operations you want to perform later will be logically clear. If you are using a high-level design tool, such as Blender or Maya, to make your models, then coherent models will be automatically built.

Why triangles? Continuing upward through the questions above, triangles are used because they are the simplest for algorithms to handle, especially if implemented in hardware. GPU implementations tend to be biased toward smaller representations so that a compact list of instructions can be applied to numerous model parts in parallel. It is certainly possible to use more complicated primitives, such as quadrilaterals, splines, and semi-algebraic surfaces [15, 25, 31]. This could lead to smaller model sizes, but often comes at the expense of greater computational cost for handling each primitive. For example, it is much harder to determine whether two spline surfaces are colliding, in comparison to two 3D triangles.

Stationary vs. movable models There will be two kinds of models in the virtual world $\mathbb{R}^3$:

- **Stationary models**, which keep the same coordinates forever. Typical examples are streets, floors, and buildings.
- **Movable models**, which can be transformed into various positions and orientations. Examples include vehicles, avatars, and small furniture.

Motion can be caused in a number of ways. Using a tracking system (Chapter 9), the model might move to match the user’s motions. Alternatively, the user might operate a controller to move objects in the virtual world, including a representation of himself. Finally, objects might move on their own according to the laws of physics in the virtual world. Section 3.2 will cover the mathematical operations that move models to the their desired places, and Chapter 8 will describe velocities, accelerations, and other physical aspects of motion.

Choosing coordinate axes One often neglected point is the choice of coordinates for the models, in terms of their placement and scale. If these are defined cleverly at the outset, then many tedious complications can be avoided. If the virtual world is supposed to correspond to familiar environments from the real world, then the axis scaling should match common units. For example, (1, 0, 0) should mean one meter to the right of (0, 0, 0). It is also wise to put the origin (0, 0, 0) in a convenient location. Commonly, $y = 0$ corresponds to the floor of a building or sea level of a terrain. The location of $x = 0$ and $z = 0$ could be in the center of the virtual world so that it nicely divides into quadrants based on sign. Another common choice is to place it in the upper left when viewing the world from above so that all $x$ and $z$ coordinates are nonnegative. For movable models, the location of the origin and the axis directions become extremely important because they affect how the model is rotated. This should become clear in Sections 3.2 and 3.3 as we present rotations.

**Viewing the models** Of course, one of the most important aspects of VR is how the models are going to “look” when viewed on a display. This problem is divided into two parts. The first part involves determining where the points in the virtual world should appear on the display. This is accomplished by viewing transformations in Section 3.4 which are combined with other transformations in Section 5.3 to produce the final result. The second part involves how each part of the model should appear after taking into account lighting sources and surface properties that are defined in the virtual world. This is the rendering problem, which is covered in Chapter 6.

3.2 Changing Position and Orientation

Suppose that a movable model has been defined as a mesh of triangles. To move it, we apply a single transformation to every vertex of every triangle. This section first considers the simple case of translation, followed by the considerably complicated case of rotations. By combining translation and rotation, the model can be placed anywhere, and at any orientation in the virtual world $\mathbb{R}^3$.

**Translations** Consider the following 3D triangle,

\[(x_1, y_1, z_1), (x_2, y_2, z_2), (x_3, y_3, z_3)\],

in which its vertex coordinates are expressed as generic constants.

Let $x_t$, $y_t$, and $z_t$ be the amount we would like to change the triangle’s position, along the $x$, $y$, and $z$ axes, respectively. The operation of changing position is called translation, and it is given by

\[
\begin{align*}
(x_1, y_1, z_1) &\rightarrow (x_1 + x_t, y_1 + y_t, z_1 + z_t) \\
(x_2, y_2, z_2) &\rightarrow (x_2 + x_t, y_2 + y_t, z_2 + z_t) \\
(x_3, y_3, z_3) &\rightarrow (x_3 + x_t, y_3 + y_t, z_3 + z_t),
\end{align*}
\]

in which $a \rightarrow b$ denotes that $a$ becomes replaced by $b$ after the transformation is applied. Applying (3.3) to every triangle in a model will translate all of it to the desired location. If the triangles are arranged in a mesh, then it is sufficient to apply the transformation to the vertices alone. All of the triangles will retain their size and shape.
3.2. CHANGING POSITION AND ORIENTATION

Figure 3.4: Every transformation has two possible interpretations, even though the math is the same. Here is a 2D example, in which a triangle is defined in (a). We could translate the triangle by \( x_t = -8 \) and \( y_t = -7 \) to obtain the result in (b). If we instead wanted to hold the triangle fixed but move the origin up by 8 in the \( x \) direction and 7 in the \( y \) direction, then the coordinates of the triangle vertices change the exact same way, as shown in (c).

Relativity Before the transformations become too complicated, we want to caution you about interpreting them correctly. Figures 3.4(a) and 3.4(b) show an example in which a triangle is translated by \( x_t = -8 \) and \( y_t = -7 \). The vertex coordinates are the same in Figures 3.4(b) and 3.4(c). Figure 3.4(b) shows the case we are intended to cover so far: The triangle is interpreted as having moved in the virtual world. However, Figure 3.4(c) shows another possibility: The coordinates of the virtual world have been reassigned so that the triangle is closer to the origin. This is equivalent to having moved the entire world, with the triangle being the only part that does not move. In this case, the translation is applied to the coordinate axes, but they are negated. When we apply more general transformations, this extends so that transforming the coordinate axes results in an inverse of the transformation that would correspondingly move the model. Negation is simply the inverse in the case of translation.

Thus, we have a kind of “relativity”: Did the object move, or did the whole world move around it? This idea will become important in Section 3.4 when we want to change viewpoints. If we were standing at the origin, looking at the triangle, then the result would appear the same in either case; however, if the origin moves, then we would move with it. A deep perceptual problem lies here as well. If we perceive ourselves as having moved, then VR sickness might increase, even though it was the object that moved. In other words, our brains make their best guess as to which type of motion occurred, and sometimes get it wrong.

Getting ready for rotations How do we make the wheels roll on a car? Or turn a table over onto its side? To accomplish these, we need to change the model’s orientation in the virtual world. The operation that changes the orientation is called rotation. Unfortunately, rotations in three dimensions are much more complicated than translations, leading to countless frustrations for engineers and developers. To improve the clarity of 3D rotation concepts, we first start with a simpler problem: 2D linear transformations.

Consider a 2D virtual world, in which points have coordinates \((x, y)\). You can imagine this as a vertical plane in our original, 3D virtual world. Now consider a generic two-by-two matrix

\[
M = \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix}
\]

in which each of the four entries could be any real number. We will look at what happens when this matrix is multiplied by the point \((x, y)\), when it is written as a column vector.

Performing the multiplication, we obtain

\[
\begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x' \\ y' \end{bmatrix},
\]

where \((x', y')\) is the transformed point. Using simple algebra, the matrix multiplication yields

\[
x' = m_{11}x + m_{12}y \\
y' = m_{21}x + m_{22}y.
\]

Using notation as in (3.5), \(M\) is a transformation for which \((x, y) \mapsto (x', y')\).

Applying the 2D matrix to points Suppose we place two points \((1, 0)\) and \((0, 1)\) in the plane. They lie on the \(x\) and \(y\) axes, respectively, at one unit of distance from the origin \((0, 0)\). Using vector spaces, these two points would be the standard unit basis vectors (sometimes written as \(i\) and \(j\)). Watch what happens if we substitute them into (3.5):

\[
\begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} m_{11} \\ m_{21} \end{bmatrix}
\]

and

\[
\begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} m_{12} \\ m_{22} \end{bmatrix}.
\]

These special points simply select the column vectors on \(M\). What does this mean? If \(M\) is applied to transform a model, then each column of \(M\) indicates precisely how each coordinate axis is changed.

Figure 3.4 illustrates the effect of applying various matrices \(M\) to a model. Starting with the upper right, the identity matrix does not cause the coordinates to change: \((x, y) \mapsto (x, y)\). The second example causes a flip as if a mirror were placed at the \(y\) axis. In this case, \((x, y) \mapsto (-x, y)\). The second row shows examples of scaling. The matrix on the left produces \((x, y) \mapsto (2x, 2y)\), which doubles the size. The matrix on the right only stretches the model in the \(y\) direction, causing an aspect ratio distortion. In the third row, it might seem that
3.2. CHANGING POSITION AND ORIENTATION

67

Figure 3.5: Eight different matrices applied to transform a square face. These examples nicely cover all of the possible cases, in a qualitative sense.

The matrix on the left produces a mirror image with respect to both $x$ and $y$ axes. This is true, except that the mirror image of a mirror image restores the original. Thus, this corresponds to the case of a 180-degree ($\pi$ radians) rotation, rather than a mirror image. The matrix on the right produces a shear along the $x$ direction: $(x, y) \mapsto (x + y, y)$. The amount of displacement is proportional to $y$. In the bottom row, the matrix on the left shows a skew in the $y$ direction: $(x, y) \mapsto (x, x, y)$. The two-dimensional shape collapses into a single dimension when $M$ is applied: $(x, y, x, y) \mapsto (mx_1 + my_2, mx_1 + my_2)$. This corresponds to the case of a singular matrix, which means that its columns are not linearly independent (they are in fact identical).

Only some matrices produce rotations. The examples in Figure 3.5 span the main qualitative differences between various two-by-two matrices $M$. Two of them were rotation matrices: the identity matrix, which is 0 degrees of rotation, and the 180-degree rotation matrix. Among the set of all possible $M$, which ones are valid rotations? We must ensure that the model does not become distorted. This is achieved by ensuring that $M$ satisfies the following rules:

1. No stretching of axes.
2. No shearing.
3. No mirror images.

If none of these rules is violated, then the result is a rotation.

To satisfy the first rule, the columns of $M$ must have unit length:

$$m_{11}^2 + m_{21}^2 = 1 \quad \text{and} \quad m_{12}^2 + m_{22}^2 = 1.$$  \hfill (3.9)

The scaling and shearing transformations in Figure 3.5 violated this.

To satisfy the second rule, the coordinate axes must remain perpendicular. Otherwise, shearing occurs. Since the columns of $M$ indicate how axes are transformed, the rule implies that their inner (dot) product is zero:

$$m_{11}m_{12} + m_{21}m_{22} = 0.$$  \hfill (3.10)

The shearing transformations in Figure 3.5 violate this rule, which clearly causes right angles in the model to be destroyed.

Satisfying the third rule requires that the determinant of $M$ is positive. After satisfying the first two rules, the only possible remaining determinants are 1 (the normal case) and $-1$ (the mirror-image case). Thus, the rule implies that:

$$\det \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} = m_{11}m_{22} - m_{12}m_{21} = 1.$$  \hfill (3.11)

The mirror image example in Figure 3.5 results in $\det M = -1$. 
3.2. CHANGING POSITION AND ORIENTATION

The first constraint (3.9) indicates that each column must be chosen so that its components lie on a unit circle, centered at the origin. In standard planar coordinates, we commonly write the equation of this circle as \( x^2 + y^2 = 1 \). Recall the common parameterization of the unit circle in terms of an angle \( \theta \) that ranges from 0 to \( 2\pi \) radians (see Figure 3.6):

\[
x = \cos \theta \quad \text{and} \quad y = \sin \theta.
\]  

(3.12)

Instead of \( x \) and \( y \), we use the notation of the matrix components. Let \( m_{11} = \cos \theta \) and \( m_{21} = \sin \theta \). Substituting this into \( M \) yields \[
\begin{bmatrix}
\cos \theta & -\sin \theta \\
\sin \theta & \cos \theta
\end{bmatrix},
\]  

(3.13)

in which \( m_{12} \) and \( m_{22} \) were uniquely determined by applying (3.10) and (3.11). By allowing \( \theta \) to range from 0 to \( 2\pi \), the full range of all allowable rotations is generated.

Think about degrees of freedom. Originally, we could chose all four components of \( M \) independently, resulting in 4 DOFs. The constraints in (3.9) each removed a DOF. Another DOF was removed by (3.11). Note that (3.11) does not reduce the DOFs; it instead eliminates exactly half of the possible transformations: The ones that are mirror flips and rotations together. The result is one DOF, which was nicely parameterized by the angle \( \theta \). Furthermore, we were lucky that set of all possible 2D rotations can be nicely interpreted as points along a unit circle.

**The 3D case**

Now we try to describe the set of all 3D rotations by following the same general template as the 2D case. The matrix from (3.3) is extended from 2D to 3D, resulting in 9 components:

\[
M = \begin{bmatrix}
m_{11} & m_{12} & m_{13} \\
m_{21} & m_{22} & m_{23} \\
m_{31} & m_{32} & m_{33}
\end{bmatrix}.
\]  

(3.14)

Thus, we start with 9 DOFs and want to determine what matrices remain as valid rotations. Follow the same three rules from the 2D case. The columns must have unit length. For example, \( m_{11}^2 + m_{21}^2 + m_{31}^2 = 1 \). This means that the components of each column must lie on a unit sphere. Thus, the unit-length rule reduces the DOFs from 9 to 6. By following the second rule to ensure perpendicular axes result, the pairwise inner products of the columns must be zero. For example, by choosing the first two columns, the constraint is

\[
m_{11}m_{12} + m_{21}m_{22} + m_{31}m_{32} = 0.
\]  

(3.15)

We must also apply the rule to the remaining pairs: The second and third columns, and then the first and third columns. Each of these cases eliminates a DOF, resulting in only 3 DOFs remaining. To avoid mirror images, the constraint \( \text{det} M = 1 \) is applied, which does not reduce the DOFs.

Finally, we arrive at a set of matrices that must satisfy the algebraic constraints; however, they unfortunately do not fall onto a nice circle or sphere. We only know that there are 3 degrees of rotational freedom, which implies that it should be possible to pick three independent parameters for a 3D rotation, and then derive all 9 elements of (3.14) from them.

**Yaw, pitch, and roll**

One of the simplest ways to parameterize 3D rotations is to construct them from “2D-like” transformations, as shown in Figure 3.7. First consider a rotation about the \( z \)-axis. Let \( \text{roll} \) be a counterclockwise rotation of \( \gamma \).
3.2. CHANGING POSITION AND ORIENTATION

about the z-axis. The rotation matrix is given by

\[
R_z(\gamma) = \begin{bmatrix}
\cos \gamma & -\sin \gamma & 0 \\
\sin \gamma & \cos \gamma & 0 \\
0 & 0 & 1
\end{bmatrix}.
\] (3.16)

The upper left of the matrix looks exactly like the 2D rotation matrix \( R(\theta) \), except that \( \theta \) is replaced by \( \gamma \). This causes yaw to behave exactly like 2D rotation in the \( xy \) plane. The remainder of \( R(\gamma) \) looks like the identity matrix, which causes \( z \) to remain unchanged after a roll.

Similarly, let \( \text{pitch} \) be a counterclockwise rotation of \( \beta \) about the x-axis:

\[
R_x(\beta) = \begin{bmatrix}
1 & 0 & 0 \\
0 & \cos \beta & -\sin \beta \\
0 & \sin \beta & \cos \beta
\end{bmatrix}.
\] (3.17)

In this case, points are rotated with respect to \( y \) and \( z \) while the \( x \) coordinate is left unchanged.

Finally, let \( \text{yaw} \) be a counterclockwise rotation of \( \alpha \) about the y-axis:

\[
R_y(\alpha) = \begin{bmatrix}
\cos \alpha & 0 & \sin \alpha \\
0 & 1 & 0 \\
-\sin \alpha & 0 & \cos \alpha
\end{bmatrix}.
\] (3.18)

In this case, rotation occurs with respect to \( x \) and \( z \) while leaving \( y \) unchanged.

Combining rotations Each of \( R_z(\gamma) \), \( R_x(\beta) \), and \( R_y(\alpha) \) provides a single DOF of rotations. The yaw, pitch, and roll rotations can be combined sequentially to attain any possible 3D rotation:

\[
R(\alpha, \beta, \gamma) = R_y(\alpha)R_x(\beta)R_z(\gamma).
\] (3.19)

In this case, the range of \( \alpha \) is from 0 to \( 2\pi \); however, the pitch \( \beta \) need only range from \( -\pi/2 \) to \( \pi/2 \) while nevertheless reaching all possible 3D rotations.

Be extra careful when combining rotations in a sequence because the operations are not commutative. For example, a yaw by \( \pi/2 \) followed by a pitch by \( \pi/2 \) does not produce the same result as the pitch followed by the yaw. You can easily check this by substituting \( \pi/2 \) into \( R_x(\beta) \) and \( R_y(\alpha) \), and observing how the result depends on the order of matrix multiplication. The 2D case is commutative because the rotation axis is always the same, allowing the rotation angles to additively combine. Having the wrong matrix ordering is one of the most frustrating problems when writing software for VR.

Matrix multiplications are “backwards” Which operation is getting applied to the model first when we apply a product of matrices? Consider rotating a point \( p = (x, y, z) \). We have two rotation matrices \( R \) and \( Q \). If we rotate \( p \) using \( R \),

we obtain \( p' = Rp \). If we then apply \( Q \), we get \( p'' = Qp' \). Now suppose that we instead want to first combine the two rotations and then apply them to \( p \) to get \( p'' \). Programmers are often tempted to combine them as \( RQ \) because we read from left to right and also write sequences in this way. However, it is backwards for linear algebra because \( Rp \) is already acting from the left side. Thus, it “reads” from right to left. We therefore must combine the rotations as \( QR \) to obtain \( p'' = QRp \). Later in this chapter, we will be chaining together several matrix transforms. Read them from right to left to understand what they are doing!

Translation and rotation in one matrix It would be convenient to apply both rotation and translation together in a single operation. Suppose we want to apply a rotation matrix \( R \), and follow it with a translation by \((x_t, y_t, z_t)\). Algebraically, this is

\[
\begin{bmatrix}
x' \\
y' \\
z'
\end{bmatrix} = R \begin{bmatrix}
x \\
y \\
z
\end{bmatrix} + \begin{bmatrix}
x_t \\
y_t \\
z_t
\end{bmatrix}.
\] (3.20)

Although there is no way to form a single 3 by 3 matrix to accomplish both operations, it can be done by increasing the matrix dimensions by one. Consider the following 4 by 4 homogeneous transform matrix:

\[
T_{rb} = \begin{bmatrix}
x_t \\
y_t \\
z_t \\
1
\end{bmatrix}
\] (3.21)

The notation \( T_{rb} \) is used to denote that the matrix is a rigid body transform, meaning that it does not distort objects. A homogeneous transform matrix could include other kinds of transforms, which will appear in Section 3.5.

The same result as in (3.20) can be obtained by performing multiplication with \( T_{rb} \) as follows:

\[
\begin{bmatrix}
x' \\
y' \\
z'
\end{bmatrix} = \begin{bmatrix}
x \\
y \\
z
\end{bmatrix} + \begin{bmatrix}
x_t \\
y_t \\
z_t
\end{bmatrix}
\] (3.22)

Because of the extra dimension, we extended the point \((x, y, z)\) by one dimension, to obtain \((x, y, z, 1)\). Note that (3.21) represents rotation followed by translation, not the other way around. Translation and rotation do not commute; therefore, this is an important point.

Inverting transforms We frequently want to invert (or undo) transformations. For a translation \((x_t, y_t, z_t)\), we simply apply the negation \((-x_t, -y_t, -z_t)\). For a

1 Perhaps coders who speak Arabic or Hebrew are not confused about this.
Figure 3.8: (a) A rigid model that is contained in a one-by-one square. (b) The result after rotation by \( \pi/4 \) (45 degrees), following by translation by \( x_t = 2 \). (c) The result after reversing the order: Translation by \( x_t = 2 \), following by rotation by \( \pi/4 \).

general matrix transform \( M \), we apply the matrix inverse \( M^{-1} \) (if it exists). This is often complicated to calculate. Fortunately, inverses are much simpler for our cases of interest. In the case of a rotation matrix \( R \), the inverse is equal to the transpose \( R^{-1} = R^T \). To invert the homogeneous transform matrix (3.21), it is tempting to write

\[
R^T = \begin{bmatrix}
-x_t & -y_t & 0 \\
0 & 0 & 1
\end{bmatrix}.
\] (3.23)

This will undo both the translation and the rotation; however, the order is wrong. Remember that these operations are not commutative, which implies that order must be correctly handled. See Figure 3.3(a). The algebra for very general matrices (part of noncommutative group theory) works out so that the inverse of a product of matrices reverses their order:

\[
(ABC)^{-1} = C^{-1}B^{-1}A^{-1}.
\] (3.24)

This can be seen by putting the inverse next to the original product: \( ABC^{-1}B^{-1}A^{-1} \). In this way, \( C \) cancels with its inverse, followed by \( B \) and its inverse, and finally \( A \) and its inverse. If the order were wrong, then these cancellations would not occur.

The matrix \( T_{rb} \) (from 3.21) applies the rotation first, followed by translation. Applying (3.24) undoes the rotation first and then translation, without reversing columns into rows.

The order. Thus, the inverse of \( T_{rb} \) is

\[
\begin{bmatrix}
R^T & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}.
\] (3.25)

The matrix on the right first undoes the translation (with no rotation). After that, the matrix on the left undoes the rotation (with no translation).

### 3.3 Axis-Angle Representations of Rotation

As observed in Section 3.2, 3D rotation is complicated for several reasons: 1) Nine matrix entries are specified in terms of only three independent parameters, and with no simple parameterization, 2) the axis of rotation is not the same every time, and 3) the operations or noncommutative, implying that the order of matrices is crucial. None of these problems existed for the 2D case.

**Kinematic singularities** An even worse problem arises when using yaw, pitch, roll angles (and related Euler-angle variants). Even though they start off being intuitively pleasing, the representation becomes degenerate, leading to kinematic singularities that are nearly impossible to visualize. An example will be presented shortly. To prepare for this, recall how we represent locations on the Earth. These are points in \( \mathbb{R}^3 \), but are represented with longitude and latitude coordinates. Just like the limits of yaw and pitch, longitude ranges from 0 to 2\( \pi \) and latitude only ranges from \(-\pi/2\) to \(\pi/2\). (Longitude is usually expressed as 0 to 180 degrees west or east, which is equivalent.) As we travel anywhere on the Earth, the latitude and longitude coordinates behave very much like \( xy \) coordinates; however, we tend to stay away from the poles. Near the North Pole, the latitude behaves normally, but the longitude could vary a large amount while corresponding to a tiny distance traveled. Recall how a wall map of the world looks near the poles: Greenland is enormous and Antarctica wraps across the entire bottom (assuming it uses a projection that keeps longitude lines straight). The poles themselves are the kinematic singularities: At these special points, you can vary longitude, but the location on the Earth is not changing. One of two DOFs seems to be lost.

The same problem occurs with 3D rotations, but it is harder to visualize due to the extra dimension. If the pitch angle is held at \( \beta = \pi/2 \), then a kind of “North Pole” is reached in which \( \alpha \) and \( \gamma \) vary independently but cause only one DOF (in the case of latitude and longitude, it was one parameter varying but causing DOFs). Here is how it looks when combining the yaw, pitch, and roll matrices:

\[
\begin{bmatrix}
\cos \alpha & \sin \alpha & 0 \\
0 & 1 & 0 \\
-\sin \alpha & 0 & \cos \alpha
\end{bmatrix}
\begin{bmatrix}
\cos \gamma & -\sin \gamma & 0 \\
\sin \gamma & \cos \gamma & 0 \\
0 & 0 & 1
\end{bmatrix}
= \begin{bmatrix}
\cos(\alpha - \gamma) & \sin(\alpha - \gamma) & 0 \\
-\sin(\alpha - \gamma) & \cos(\alpha - \gamma) & 0 \\
0 & 0 & 1
\end{bmatrix}.
\] (3.26)
3.3. AXIS-ANGLE REPRESENTATIONS OF ROTATION

The second matrix above corresponds to pitch with \( \beta = \pi/2 \). The result on the right is obtained by performing matrix multiplication and applying a subtraction trigonometric identity. You should observe that the resulting matrix is a function of both \( \alpha \) and \( \gamma \), but there is one DOF because only the difference \( \alpha - \gamma \) affects the resulting rotation. In the video game industry there has been some back-and-forth battles about whether this problem is crucial. In an FPS game, the avatar is usually not allowed to pitch his head all the way to \( \pi/2 \), thereby forcing his head straight up or down. The kinematic singularity often causes the virtual avatar to stop. The problems can be easily solved with axis-angle representations of rotation. They are harder to learn than yaw, pitch, and roll; however, it is a worthwhile investment because it avoids these problems. Furthermore, many well-written software libraries and game engines work directly with these representations. Thus, to use them effectively, you should understand what they are doing.

The most important insight to solving the kinematic singularity problems is Euler’s rotation theorem (1775), shown in Figure 3.9. Even though the rotation axis may change after rotations are combined, Euler showed that any 3D rotation can be expressed as a rotation \( \theta \) about some axis that passes through the origin. This matches the three DOFs for rotation: It takes two parameters to specify the direction of an axis and one parameter for \( \theta \). The only trouble is that conversions back and forth between rotation matrices and the axis-angle representation are somewhat inconvenient. This motivates the introduction of a mathematical object that is close to the axis-angle representation, closely mimics the algebra of 3D rotations, and can even be applied directly to rotate models. The perfect representation: Quaternions.

Two-to-one problem Before getting to quaternions, it is important point out one annoying problem with Euler’s rotation theorem. As shown in Figure 3.10 it does not claim that the axis-angle representation is unique. In fact, for every 3D rotation other than the identity, there are exactly two representations. This is due to the fact that the axis could “point” in either direction. We could insist that the axis always point in one direction, such as positive \( y \), but this does not fully solve the problem because of the boundary cases (horizontal axes). Quaternions, which are coming next, nicely handle all problems with 3D rotations except this one, which is unavoidable.

Quaternions were introduced in 1843 by William Rowan Hamilton. When seeing them the first time, most people have difficulty understanding their peculiar algebra. Therefore, we will instead focus on precisely which quaternions correspond to which rotations. After that, we will introduce some limited quaternion algebra. The algebra is much less important for developing VR systems, unless you want to implement your own 3D rotation library. The correspondence between quaternions and 3D rotations, however, is crucial.

A quaternion \( h \) is a 4D vector:

\[
q = (a, b, c, d),
\]

in which \( a, b, c, \) and \( d \) can take on real values. Thus, \( q \) can be considered as a point in \( \mathbb{R}^4 \). It turns out that we will only use unit quaternions, which means that

\[
a^2 + b^2 + c^2 + d^2 = 1
\]

must always hold. This should remind you of the equation of a unit sphere \((x^2 + y^2 + z^2 = 1)\), but it is one dimension higher. A sphere is a 2D surface, whereas the set of all unit quaternions is a 3D “hypersurface”, more formally known as a
3.3. AXIS-ANGLE REPRESENTATIONS OF ROTATION

<table>
<thead>
<tr>
<th>Quaternion</th>
<th>Axis-Angle</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1, 0, 0, 0)</td>
<td>(undefined, 0)</td>
<td>Identity rotation</td>
</tr>
<tr>
<td>(0, 1, 0, 0)</td>
<td>(1, 0, 0, π)</td>
<td>Pitch by π</td>
</tr>
<tr>
<td>(0, 0, 1, 0)</td>
<td>(1, 0, 0, π)</td>
<td>Yaw by π</td>
</tr>
<tr>
<td>(0, 0, 0, 1)</td>
<td>(0, 0, 1, π)</td>
<td>Roll by π</td>
</tr>
<tr>
<td>(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0, 0)</td>
<td>(1, 0, 0, π/2)</td>
<td>Pitch by π/2</td>
</tr>
<tr>
<td>(\frac{1}{\sqrt{2}}, 0, \frac{1}{\sqrt{2}}, 0)</td>
<td>(0, 1, 0, π/2)</td>
<td>Yaw by π/2</td>
</tr>
<tr>
<td>(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0, -\frac{1}{\sqrt{2}})</td>
<td>(0, 0, 1, π/2)</td>
<td>Roll by π/2</td>
</tr>
</tbody>
</table>

Figure 3.11: For these cases, you should be able to look at the quaternion and quickly picture the axis and angle of the corresponding 3D rotation.

\[ (a, b, c, d) \leftrightarrow \text{equivalent} \leftrightarrow (-a, -b, -c, -d) \]

\[ (a, b, c, d) \leftrightarrow \text{equivalent} \leftrightarrow (-a, -b, -c, -d) \]

\[ (a, b, c, d) \leftrightarrow \text{equivalent} \leftrightarrow (-a, b, c, d) \]

Figure 3.12: Simple relationships between equivalent quaternions and their inverses.

**manifold** [3, 20]. We will use the space of unit quaternions to represent the space of all 3D rotations. Both have 3 DOFs, which seems reasonable.

Let \((v, \theta)\) be an axis-angle representation of a 3D rotation, as depicted in Figure 3.10. Let this be represented by the following quaternion:

\[ q = \left( \cos \frac{\theta}{2}, v_1 \sin \frac{\theta}{2}, v_2 \sin \frac{\theta}{2}, v_3 \sin \frac{\theta}{2} \right). \]  

(3.29)

Think of \(q\) as a data structure that encodes the 3D rotation. It is easy to recover \((v, \theta)\) from \(q\):

\[ \theta = 2 \cos^{-1} a \text{ and } v = \frac{1}{\sqrt{1 - a^2}}(b, c, d). \]  

(3.30)

If \(a = 1\), then (3.30) breaks; however, this corresponds to the case of the identity rotation.

You now have the mappings \((v, \theta) \leftrightarrow q\) and \(q \leftrightarrow (v, \theta)\). To test your understanding, Figure 3.11 shows some simple examples, which commonly occur in practice. Furthermore, Figure 3.12 shows some simple relationships between quaternions and their corresponding rotations. The horizontal arrows indicate that \(q\) and \(-q\) represent the same rotation. This is true because of the double representation issue shown in Figure 3.10. Applying (3.29) to both cases establishes their equivalence. The vertical arrows correspond to inverse rotations. These hold because reversing the direction of the axis causes the rotation to be be reversed (rotation by \(\theta\) becomes rotation by \(-\theta\).

How do we apply the quaternion \(h = (a, b, c, d)\) to rotate the model? One way is to use the following conversion into a 3D rotation matrix:

\[ R(h) = \begin{bmatrix} 2(a^2 + b^2) - 1 & 2(bc - ad) & 2(bd + ac) \\ 2(bc + ad) & 2(a^2 + c^2) - 1 & 2(cd - ab) \\ 2(bd - ac) & 2(cd + ab) & 2(a^2 + d^2) - 1 \end{bmatrix}. \]  

(3.31)

A more efficient way exists which avoids converting into a rotation matrix. To accomplish this, we need to define quaternion multiplication. For any two quaternions, \(q_1\) and \(q_2\), let \(q_1 * q_2\) denote the product, which is defined as

\[ a_3 = a_1 a_2 - b_1 b_2 - c_1 c_2 - d_1 d_2 \]
\[ b_3 = a_1 b_2 + a_2 b_1 + c_1 d_2 - c_2 d_1 \]
\[ c_3 = a_1 c_2 + a_2 c_1 + b_2 d_1 - b_1 d_2 \]
\[ d_3 = a_1 d_2 + a_2 d_1 + b_1 c_2 - b_2 c_1. \]

(3.32)

In other words, \(q'' = q * q'\) as defined in (3.32).

Here is a way to rotate the point \((x, y, z)\) using the rotation represented by \(h\). Let \(p = (x, y, z, 1)\), which is done to give the point the same dimensions as a quaternion. Believe it or not, the point is rotated by applying quaternion multiplication as

\[ p' = q * p * q^{-1}, \]  

(3.33)

in which \(q^{-1} = (a, -b, -c, -d)\) (recall from Figure 3.12). The rotated point is \((x', y', z')\), which is taken from the result \(p' = (x', y', z', 1').\)

Here is a simple example for the point \((1, 0, 0, 0)\). Let \(p = (1, 0, 0, 1)\) and consider executing a yaw rotation by \(\pi/2\). According to Figure 3.11, the corresponding quaternion is \(q = (0, 0, 1, 0)\). The inverse is \(q^{-1} = (0, 0, -1, 0)\). After tediously applying (3.32) to calculate (3.33), the result is \(p' = (0, 0, 1, 1)\). Thus, the rotated point is \((0, 1, 0, 1)\), which is a correct yaw by \(\pi/2\).

3.4 Viewing Transformations

This section describes how to transform the models in the virtual world so that they appear on a virtual screen. The main purpose is to set the foundation for graphical rendering, which adds effects due to lighting, material properties, and quantization. Ultimately, the result appears on the physical display. One side effect of these transforms is that they also explain how cameras form images, at least the idealized mathematics of the process. Think of this section as describing a virtual camera that is placed in the virtual world. What should the virtual picture, taken by that camera, look like? To make VR work correctly, the “camera” should actually be one of two virtual human eyes that are placed into the virtual world.
3.4. VIEWING TRANSFORMATIONS

Figure 3.13: If we placed a virtual eye or camera into the virtual world, what would it see? Section 3.3 provides transformations that place objects from the virtual world onto a virtual screen, based on the particular viewpoint of a virtual eye. A flat rectangular shape is chosen for engineering and historical reasons, even though it does not match the shape of our retinas.

Thus, what should a virtual eye see, based on its position and orientation in the virtual world? Rather than determine precisely what would appear on the retina, which should become clear after Section 3.4, here we merely calculate where the model vertices would appear on a flat, rectangular screen in the virtual world. See Figure 3.13.

**An eye’s view** Figure 3.14 shows a virtual eye that is looking down the negative z axis. It is placed in this way so that from the eye’s perspective, x increases to the right and y is upward. This corresponds to familiar Cartesian coordinates. The alternatives would be: 1) to face the eye in the positive z direction, which makes the xy coordinates appear backwards, or 2) reverse the z axis, which would unfortunately lead to a left-handed coordinate system. Thus, we have made an odd choice that avoids worse complications.

Suppose that the eye is an object model that we want to place into the virtual world \( \mathbb{R}^3 \) at some position \( e = (e_1, e_2, e_3) \) and orientation given by the matrix

\[
R_{\text{eye}} = \begin{bmatrix}
\hat{x}_1 & \hat{y}_1 & \hat{z}_1 \\
\hat{x}_2 & \hat{y}_2 & \hat{z}_2 \\
\hat{x}_3 & \hat{y}_3 & \hat{z}_3
\end{bmatrix}.
\] (3.34)

If the eyeball in Figure 3.14 were made of triangles, then rotation by \( R_{\text{eye}} \) and translation by \( e \) would be applied to all vertices to place it in \( \mathbb{R}^3 \).

This does not, however, solve the problem of how the virtual world should appear to the eye. Rather than moving the eye in the virtual world, we need to move all of the models in the virtual world to the eye’s frame of reference. This means that we need to apply the inverse transformation. Recall the inverse transform. The inverse rotation is \( R_{\text{eye}}^T \), the transpose of \( R_{\text{eye}} \). The inverse of \( e \) is \(-e\). Applying (3.25) results in the appropriate transform:

\[
T_{\text{eye}} = \begin{bmatrix}
\hat{x}_1 & \hat{x}_2 & \hat{x}_3 & 0 \\
\hat{y}_1 & \hat{y}_2 & \hat{y}_3 & 0 \\
\hat{z}_1 & \hat{z}_2 & \hat{z}_3 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
1 & 0 & 0 & -e_1 \\
0 & 1 & 0 & -e_2 \\
0 & 0 & 1 & -e_3 \\
0 & 0 & 0 & 1
\end{bmatrix}.
\] (3.35)

Note that \( R_{\text{eye}} \), as shown in (3.34), has been transposed and placed into the left matrix above. Also, the order of translation and rotation have been swapped, which is required for the inverse, as mentioned in Section 3.2.

Following Figure 3.3, there are two possible interpretations of (3.35). As stated, this could correspond to moving all of the virtual world models (corresponding to Figure 3.3(b)). A more appropriate interpretation in the current setting is that the virtual world’s coordinate frame is being moved so that it matches the eye’s frame from Figure 3.3(c). This corresponds to the case of Figure 3.4(c), which was not the appropriate interpretation in Section 3.2.

**Starting from a look-at** For VR, the position and orientation of the eye in the virtual world are given by a tracking system and possibly controller inputs. By contrast, in computer graphics, it is common to start with a description of where the eye is located and which way it is looking. This is called a look-at, and has the following components:
3.4. VIEWING TRANSFORMATIONS

1. Position of the eye: e

2. Central looking direction of the eye: c


Both c and u are unit vectors. The first direction c corresponds to the center of the view. Whatever c is pointing at should end up in the center of the display. If we want this to be a particular point p in \( \mathbb{R}^3 \) (see Figure 3.15), then c can be calculated as

\[
\hat{c} = \frac{p - e}{\|p - e\|}
\]

in which \( \| \cdot \| \) denotes the length of a vector. The result is just the vector from e to p, but normalized.

The second direction u indicates which way is up. Imagine holding a camera out as if you are about to take a photo and then perform a roll rotation. You can make level ground appear to be slanted or even upside down in the picture. Thus, u indicates the up direction for the virtual camera or eye.

We now construct the resulting transform from (3.35). The translation components are already determined by e, which was given in the look-at. We need only to determine the rotation \( R_{eye} \), as expressed in (3.36). Recall from Section 3.2 that the matrix columns indicate how the coordinate axes are transformed by the matrix (refer to (3.8) and (3.9)). This simplifies the problem of determining \( R_{eye} \). Each column vector is calculated as

\[
\begin{align*}
\hat{z} &= -\hat{c} \\
\hat{x} &= \hat{u} \times \hat{z} \\
\hat{y} &= \hat{z} \times \hat{x}.
\end{align*}
\]

The minus sign appears for calculating \( \hat{z} \) because the eye is looking down the negative z axis. The \( \hat{x} \) direction is calculated using the standard cross product \( \hat{z} \). For the third equation, we could use \( \hat{y} = \hat{u} \); however, \( \hat{z} \times \hat{x} \) will cleverly correct cases in which \( \hat{u} \) generally points upward but is not perpendicular to \( \hat{c} \). The unit vectors from (3.36) are substituted into (3.34) to obtain \( R_{eye} \). Thus, we have all the required information to construct \( T_{eye} \).

Orthographic projection Let \((x, y, z)\) denote the coordinates any point, after \( T_{eye} \) has been applied. What would happen if we took all points and directly projected them into the vertical xy plane by forcing each z coordinate to be 0? In other words, \((x, y, z) \rightarrow (x, y, 0)\), which is called orthographic projection. If we imagine the xy plane as a virtual display of the models, then there would be several problems:

1. A jumble of objects would be superimposed, rather than hiding parts of a model that are in front of another.

2. The display would extend infinitely in all directions (except z). If the display is a small rectangle in the xy plane, then the model parts that are outside of its range can be eliminated.

3. Objects that are closer should appear larger than those further away. This happens in the real world. Recall from Section 3.3 (Figure 3.12) paintings that correctly handle perspective.

The first two problems are important graphics operations that are deferred until Chapter 4. The third problem is addressed next.

Perspective projection Instead of using orthographic projection, we define a perspective projection. For each point \((x, y, z)\), consider a line through the origin. This is the set of all points with coordinates

\[
(\lambda x, \lambda y, \lambda z),
\]

in which \( \lambda \) can be any real number. In other words \( \lambda \) is a parameter that reaches all points on the line that contains both \((x, y, z)\) and \((0, 0, 0)\). See Figure 3.16.

Now we can place a planar “movie screen” anywhere in the virtual world and see where all of the lines pierce it. To keep the math simple, we pick the \( z = -1 \) plane to place our virtual screen directly in front of the eye. Using the third component of (3.38), we have \( \lambda z = -1 \), implying that \( \lambda = -1/z \). Using the first two components of (3.38), the coordinates for the points on the screen are calculated as \( x' = -x/z \) and \( y' = -y/z \). Note that since \( x \) and \( y \) are scaled by the same amount \( z \) for each axis, their aspect ratio is preserved on the screen.

More generally, suppose the vertical screen is placed some location \( d \) along the z axis. In this case, we obtain more general expressions for the location of a point

\[
\begin{align*}
\lambda &= \frac{d}{z + d} \\
x' &= x/z \cdot \frac{d}{z + d} \\
y' &= y/z \cdot \frac{d}{z + d}.
\end{align*}
\]
3.5. CHAINING THE TRANSFORMATIONS

This section links all of the transformations of this chapter together while also slightly adjusting their form to match what is currently used in the VR and computer graphics industries. Some of the matrices appearing in this section may seem unnecessarily complicated. The reason is that the expressions are motivated by algorithm and hardware issues, rather than mathematical simplicity. In particular, there is a bias toward putting every transformation into a $4 \times 4$ homogeneous transform matrix, even in the case of perspective projection which is not even linear (recall (3.8)). In this way, an efficient matrix multiplication algorithm can be iterated over the chain of matrices to produce the result.

Figure 3.17: An illustration of perspective projection. The model vertices are projected onto a virtual screen by drawing lines through them and the origin $(0,0,0)$. The “image” of the points on the virtual screen corresponds to the intersections of the line with the screen.

on the screen:
\[
x' = dx/z \\
y' = dy/z.
\]

This was obtained by solving $d = \lambda z$ for $\lambda$ and substituting it into (3.38).

This is all we need to project the points onto a virtual screen, while respecting the scaling properties of objects at various distances. Getting this right in VR helps in the perception of depth and scale, which are covered in Section 5.1. In Section 5.3 we will adapt (3.39) using transformation matrices. Furthermore, only points that lie within a zone in front of the eye will be projected onto the virtual screen. Points that are too close, too far, or in outside the normal field of view will not be rendered on the virtual screen; this is addressed in Section 5.3 and Chapter 6.

3.5 Chaining the Transformations

The chain generally appears as follows:
\[
T = T_{ep}T_{cam}T_{eye}T_{rb}.
\] (3.40)

When $T$ is applied to a point $(x,y,z,1)$, the location of the point on the screen is produced. Remember that these matrix multiplications are not commutative, and the operations are applied from right to left. The first matrix $T_{rb}$ is the rigid body transform $T_{rb}$ applied to points on a movable model. For each rigid object in the model, $T_{rb}$ remains the same; however, different objects will generally be placed in various positions and orientations. For example, the wheel of a virtual car will move differently than the avatar's head. After $T_{rb}$ is applied, $T_{eye}$ transforms the virtual world into the coordinate frame of the eye, according to (3.38). At a fixed instant in time, this and all remaining transformation matrices are the same for all points in the virtual world. Here we assume that the eye is positioned at the midpoint between the two virtual human eyes, leading to a cyclopean viewpoint. Later in this section, we will extend it to the case of left and right eyes so that stereo viewpoints can be constructed.

**Canonical view transform** The next transformation, $T_{cam}$ performs the perspective projection as described in Section 4.4; however, we must explain how it is unnaturally forced into a $4 \times 4$ matrix. We also want the result to be in a canonical form that appears to be unitless, which is again motivated by industrial needs. Therefore, $T_{cam}$ is called the canonical view transform. Figure 3.18 shows a viewing frustum, which is based on the four corners of a rectangular virtual screen. At $z = n$ and $z = f$ lie a near plane and far plane, respectively. Note that $z < 0$ for these cases because the $z$ axis points in the opposite direction. The virtual screen is contained in the near plane. The perspective projection should place all of the points inside of the frustum onto a virtual screen that is centered in the near plane. This implies $d = n$ using (3.39).

We now want to try to reproduce (3.39) using a matrix. Consider the result of
applying the following matrix multiplication:

\[
\begin{bmatrix}
  n & 0 & 0 & 0 \\
  0 & n & 0 & 0 \\
  0 & 0 & n & 0 \\
  0 & 0 & 0 & 1 \\
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  z \\
  1 \\
\end{bmatrix}
= \begin{bmatrix}
  nx \\
  ny \\
  nz \\
  z \\
\end{bmatrix}.
\]

(3.41)

In the first two coordinates, we obtain the numerator of (3.39). The nonlinear part of (3.39) is the \(1/z\) factor. To handle this, the fourth coordinate is used to represent \(z\), rather than 1 as in the case of \(T_b\). From this point onward, the resulting 4D vector is interpreted as a 3D vector that is scaled by dividing out its fourth component. For example, \((v_1, v_2, v_3, v_4)\) is interpreted as

\[
(v_1/v_4, v_2/v_4, v_3/v_4).
\]

(3.42)

Thus, the result from (3.41) is interpreted as

\((nx/z, ny/z, n)\),

(3.43)

in which the first two coordinates match (3.41) with \(d = n\), and the third coordinate is the location of the virtual screen along the \(z\) axis.

**Keeping track of depth for later use** The following matrix is commonly used in computer graphics, and will be used here in our chain:

\[
T_p = \begin{bmatrix}
  n & 0 & 0 & 0 \\
  0 & n & 0 & 0 \\
  0 & 0 & n + f & -fn \\
  0 & 0 & 1 & 0 \\
\end{bmatrix}.
\]

(3.44)

It is identical to the matrix in (3.41) except in how it transforms the \(z\) coordinate. For purposes of placing points on the virtual screen, it is unnecessary because we already know they are all placed at \(z = n\). The \(z\) coordinate is therefore co-opted for another purpose: Keeping track of the distance of each point from the eye so that graphics algorithms can determine which objects are in front of other objects. The matrix \(T_p\) calculates the third coordinate as

\[(n + f)z - fn\]

(3.45)

When divided by \(z\), (3.45) does not preserve the exact distance, but the graphics methods (some of which are covered in Chapter 7) require only that the distance ordering is preserved. In other words, if point \(p\) is further from the eye than point \(q\), then it remains further after the transformation, even if the distances are distorted. It does, however, preserve the distance in two special cases: \(z = n\) and \(z = f\). This can be seen by substituting these into (3.41) and dividing by \(z\).

---

Figure 3.19: The rectangular region formed by the corners of the viewing frustum, after they are transformed by \(T_p\). The coordinates of the selected opposite corners provide the six parameters, \(\ell, r, b, t, n,\) and \(f\), which used in \(T_{st}\).

**Additional translation and scaling** After \(T_p\) is applied, the 8 corners of the frustum are transformed into the corners of a rectangular box, shown in Figure 3.19. The following performs a simple translation of the box along the \(z\) axis and some rescaling so that it is centered at the origin and the coordinates of its corners are \((\pm 1, \pm 1, \pm 1)\):

\[
T_{st} = \begin{bmatrix}
  \frac{2}{\ell} & 0 & 0 & -\frac{\ell + f}{2} \\
  0 & \frac{2}{r-b} & 0 & -\frac{r + b}{2} \\
  0 & 0 & n - f & -n - f \\
  0 & 0 & 0 & 1 \\
\end{bmatrix}.
\]

(3.46)

If the frustum is perfectly centered in the \(xy\) plane, then the first two components of the last column become 0. Finally, we define the canonical view transform \(T_{can}\) from (3.40) as

\[
T_{can} = T_{st}T_p.
\]

(3.47)

**Viewport transform** The last transform to be applied in the chain (3.40) is the viewport transform \(T_{vp}\). After \(T_{can}\) has been applied, the \(x\) and \(y\) coordinates each range from \(-1\) to 1. One last step is required to bring the projected points to the coordinates used to index pixels on a physical display. Let \(m\) be the number of horizontal pixels and \(n\) be the number of vertical pixels. For example, \(n = 1080\) and \(m = 1920\) for a 1080p display. Suppose that the display is indexed with rows running from 0 to \(n - 1\) and columns from 0 to \(m - 1\). Furthermore, \((0, 0)\) is in the lower left corner. In this case, the viewport transform is

\[
T_{vp} = \begin{bmatrix}
  \frac{m}{2} & 0 & 0 & \frac{m - 1}{2} \\
  0 & \frac{n}{2} & 0 & \frac{n - 1}{2} \\
  0 & 0 & 1 & 0 \\
  0 & 0 & 0 & 1 \\
\end{bmatrix}.
\]

(3.48)

**Left and right eyes** We now address how the transformation chain (3.40) is altered for stereoscopic viewing. Let \(t\) denote the distance between the left and
right eyes. Its value in the real world varies across people, and its average is around \( t = 0.064 \) meters. To handle the left eye view, we need to simply shift the cyclopean (center) eye horizontally to the left. Recall from Section 3.4 that the inverse actually gets applied. The models need to be shifted to the right. Therefore, let

\[
T_{\text{left}} = \begin{bmatrix}
1 & 0 & 0 & \frac{t}{2} \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix},
\]

(3.49)

which corresponds to a right shift of the models, when viewed from the eye. This transform is placed after \( T_{\text{eye}} \) to adjust its output. The appropriate modification to (3.40) is:

\[
T = T_{\text{vp}}T_{\text{can}}T_{\text{left}}T_{\text{eye}}T_{\text{rb}}.
\]

(3.50)

By symmetry, the right eye is similarly handled by replacing \( T_{\text{left}} \) in (3.50) with

\[
T_{\text{right}} = \begin{bmatrix}
1 & 0 & 0 & -\frac{t}{2} \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}.
\]

(3.51)

This concludes the explanation of the entire chain of transformations to place and move models in the virtual world and then have them appear in the right place on a display. After reading Chapter 4, it will become clear that one final transformation may be needed after the entire chain has been applied. This is done to compensate for nonlinear optical distortions that occur due to wide-angle lenses in VR headsets.

**Further Reading**

References on transforming chains of bodies (35 Chapter 3), and animating articulated structures.

The fact that mesh orientations cannot be consistently labeled for some surfaces is the basis of homology! Should include some topology references.

Euler angle references.

Need quaternion algebra references, more conversions, and derivations of all the given conversions.
Chapter 4
Light and Optics

Knowing how light propagates in the physical world is crucial to understanding VR. One reason is the interface between visual displays and our eyes. Light is emitted from displays and arrives on our retinas in a way that convincingly reproduces how light arrives through normal vision in the physical world. In the current generation of VR headsets, a system of both engineered and natural lenses (parts of our eyes) guide the light. Another reason to study light propagation is the construction of virtual worlds. Chapter 3 covered purely geometric aspects of modeling. The next logical step is to model how light propagates through virtual worlds to be rendered on a display; this will be continued in Chapter 7. Finally, light propagation is also helpful to understanding how cameras work, which provides another way present a virtual world: Through panoramic videos.

Section 4.1 covers basic physical properties of light, including its interaction with materials and its spectral properties. Section 4.2 provides idealized models of how lenses work. Section 4.3 then shows many ways that lens behavior deviates from the ideal model, thereby degrading VR experiences. Section 4.4 introduces the human eye as an optical system of lenses, before eyes and human vision are covered in much more detail in Chapter 5. Cameras, which can be considered as engineered eyes, are introduced in Section 4.5.

4.1 Basic Behavior of Light

Light can be described in three ways that appear to be mutually incompatible:

1. Photons: Tiny particles of energy moving through space at high speeds (no need for quantum mechanics in this book!). This interpretation is helpful when considering the amount of light received by a sensor or receptor.

2. Waves: Ripples through space that are similar to waves propagating on the surface of water, but are 3D. The wavelength is the distance between peaks. This interpretation is helpful when considering the spectrum of colors.

3. Rays: A ray traces the motion of a single hypothetical photon. The direction is perpendicular to the wavefronts (see Figure 4.1). Finally, light propagation is also helpful to understanding how cameras work, which provides another way present a virtual world: Through panoramic videos.

Figure 4.1: Waves and visibility rays emanating from a point light source.

Fortunately, modern physics has explained how these interpretations are in fact compatible; each is useful in this book.

Spreading waves Figure 4.1 shows how waves would propagate from a hypothetical point light source. The density would be the same in all directions (radial symmetry), but would decrease as the light source becomes more distant. Recall that the surface area of a sphere with radius $r$ is $4\pi r^2$. Consider centering a spherical screen around the light source. The total number of photons per second hitting a screen of radius 1 should be the same as for a screen of radius 2; however, the density (photons per second per area) should decrease by a factor of $1/4$ because they are distributed over 4 times the area. Thus, photon density decreases quadratically as a function of distance from a point light source.

The curvature of the wavefronts also decreases as the point light source becomes further away. If the waves were to propagate infinitely far away, then they would completely flatten as shown in Figure 4.2. This results in the important case...
4.1. BASIC BEHAVIOR OF LIGHT

Figure 4.2: If the point light source were “infinitely far” away, then parallel wavefronts would be obtained. Other names for this setting are: Collimated light, parallel rays, rays from infinity, rays to infinity, and zero vergence.

of parallel wavefronts. Without the help of lenses or mirrors, it is impossible to actually obtain this case from a tiny light source in the physical world because it cannot be so far away; however, it serves as both a useful approximation for distant light sources and as an ideal way to describe lenses mathematically. Keep in mind that at any finite distance from a point light source, the rays of light always diverge; it is impossible to make them converge without the help of lenses or mirrors.

Interactions with materials As light strikes the surface of a material, one of three behaviors might occur, as shown in Figure 4.3. In the case of transmission, the energy travels through the material and exits the other side. For a transparent material, such as glass, the transmitted light rays are slowed down and bend according to Snell’s law, which will be covered in Section 4.2. For a translucent material that is not transparent, the rays scatter into various directions before exiting. In the case of absorption, energy is absorbed by the material as the light becomes trapped. The third case is reflection, in which the light is deflected from the surface. Along a perfectly smooth or polished surface, the rays reflect in the same way: The exit angle is equal to the entry angle. See Figure 4.4. This case is called specular reflection, in contrast to diffuse reflection, in which the reflected rays scatter in arbitrary directions. Usually, all three cases of transmission, absorption, and reflection occur simultaneously. The amount of energy divided between the cases depends on many factors, such as the angle of approach, the wavelength, and differences between the materials.

A jumble of wavelengths Figure 4.1 presented an oversimplified view that will make it easy to understand idealized lenses in Section 4.2. Unfortunately, it misses many details that become important in other settings, such as understanding lens aberrations (Section 4.3) or how light interacts with materials in the physical world.
4.1. BASIC BEHAVIOR OF LIGHT

Figure 4.5: Visible light spectrum corresponds to the range of electromagnetic waves that have wavelengths between 400nm and 700nm. (Figure by David Eccles for Wikipedia.)

The remainder of this section therefore considers various realistic complications that arise.

Coherent versus jumbled light  The first complication is that light sources usually do not emit coherent light, a term that means the wavefronts are perfectly aligned in time and space. A laser is an exceptional case that indeed produces coherent light. It emits parallel waves of a constant wavelength that are also synchronized in time so that their peaks align as they propagate. Common light sources, such as light bulbs and the sun, instead emit a jumble of waves that have various wavelengths and do not have their peaks aligned.

Wavelengths and colors  To make sense out of the jumble of waves, we will describe how they are distributed in terms of wavelengths. Figure 4.6 shows the range of wavelengths that are visible to humans. Each wavelength corresponds to a spectral color, which is what we would perceive with a coherent light source fixed at that wavelength alone. Wavelengths between 700 and 1000nm are called infrared, which are not visible to us, but our cameras can sense them (see Section 9.2). Wavelengths between 100 and 400nm are called ultraviolet; they are not part of our visible spectrum, but some birds, insects, and fish can perceive ultraviolet wavelengths over 300nm. Thus, our notion of visible light is already tied to human perception.

Spectral power  Figure 4.7 shows how the wavelengths are distributed for common light sources. An ideal light source would have all visible wavelengths represented with equal energy, leading to idealized white light. The opposite is total darkness, which is black. We usually do not allow a light source to propagate light directly onto our retinas (don’t stare at the sun!). Instead, we observe light that is reflected from objects all around us, causing us to perceive their color. Each surface has its own distribution of wavelengths that it reflects. The fraction of light energy that is reflected back depends on the wavelength, leading to the plots shown in Figure 4.7. For us to perceive an object surface as red, the red wavelengths must be included in the light source and the surface must strongly...
4.2 LENSES

Lenses have been made for thousands of years, with the oldest known artifact shown in Figure 4.8(a). It was constructed before 700 BC in Assyrian Nimrud, which was coincidentally mentioned in Figure 1.15 of Chapter 1. Whether constructed from transparent materials or from polished surfaces that act as mirrors, lenses bend rays of light so that a focused image is formed. Over the centuries, their uses have given rise to several well-known devices, such as eyeglasses (Figure 4.8(b)), telescopes, magnifying glasses, binoculars, cameras, and microscopes. Optical engineering is therefore filled with design patterns that indicate how to optimize the designs of these well-designed devices. VR headsets are a newcomer among existing optical devices, leading to many new challenges that are outside of standard patterns that have existed for centuries. Thus, the lens design patterns for VR are still being written. To first step toward addressing the current challenges is to understand how simple lenses work.

Snell’s Law

Lenses work because of Snell’s Law, which expresses how much rays of light bend when entering and exiting a transparent material. Recall that the speed of light in a vacuum is a universal constant $c$ with value approximately $3 \times 10^8$ m/s. In this case, $s = c$ in (4.1). Light propagates roughly 0.03 percent faster in a vacuum than in air, causing the difference to be neglected in most engineering calculations. Visible light in air has a frequency range of roughly 400 to 800 terahertz, which is obtained by applying (4.1). As light propagates through denser media, such as water or lenses, $s$ is significantly smaller; that difference is the basis of optical systems, which are covered next.

**Figure 4.8:** (a) The earliest known artificially constructed lens, which was made between 750 and 710 BC in Assyrian Nimrud. It is not known whether this artifact was purely ornamental or used to produce focused images. Picture from the British Museum. (b) A painting by Conrad con Soest from 1403, which shows the use of reading glasses for an elderly male.

reflect red wavelengths. Other wavelengths must also be suppressed. For example, the light source could be white (containing all wavelengths) and the object could strongly reflect all wavelengths, causing the surface to appear white, not red. Section 6.3 will provide more details on color perception.

**Frequency** Often times, it is useful to talk about frequency instead of wavelength. The frequency $f$ is the number of times per second that wave peaks pass through a fixed location. Using both the wavelength $\lambda$ and the speed $s$, the frequency $f$ is calculated as:

$$f = \frac{s}{\lambda}.$$  

(4.1)

The speed of light in a vacuum is a universal constant $c$ with value approximately equal to $3 \times 10^8$ m/s. In this case, $s = c$ in (4.1). Light propagates roughly 0.03 percent faster in a vacuum than in air, causing the difference to be neglected in most engineering calculations. Visible light in air has a frequency range of roughly 400 to 800 terahertz, which is obtained by applying (4.1). As light propagates through denser media, such as water or lenses, $s$ is significantly smaller; that difference is the basis of optical systems, which are covered next.

**Snell’s Law** Lenses work because of Snell’s Law, which expresses how much rays of light bend when entering and exiting a transparent material. Recall that the speed of light in a medium is less than the speed $c$ in a vacuum. For a given material, let its *refractive index* be defined as

$$n = \frac{c}{s},$$

(4.2)
in which $s$ is the speed of light in the medium. For example, $n = 2$ means that light takes twice as long to traverse the medium as through a vacuum. For some common examples, $n = 1.000293$ for air, $n = 1.33$ for water, and $n = 1.523$ for crown glass.

**Figure 4.9** shows what happens to incoming light waves and rays. Suppose in this example that the light is traveling from air into glass, so that $n_1 < n_2$. Let $\theta_1$ represent the incoming angle with respect to the surface normal, and let $\theta_2$ represent the resulting angle as it passes through the material. Snell’s law relates the four quantities as

$$n_1 \sin \theta_1 = n_2 \sin \theta_2.$$  

(4.3)

Typically, $n_1/n_2$ and $\theta_1$ are given, so that (4.3) is solved for $\theta_2$ to obtain

$$\theta_2 = \sin^{-1}\left(\frac{n_1 \sin \theta_1}{n_2}\right).$$

(4.4)

If $n_1 < n_2$, then $\theta_2$ is closer to perpendicular than $\theta_1$. If $n_2 > n_1$, then $\theta_2$ is further from perpendicular. The case of $n_1 > n_2$ is also interesting in that light may not penetrate the surface if the incoming angle $\theta_1$ is too large. The range of $\sin^{-1}$ is 0 to 1, which implies that (4.4) provides a solution for $\theta_2$ only if $(n_1/n_2) \sin \theta_1 \leq 1$. If the condition does not hold, then the light rays always reflect from the surface. This situation occurs while under water and looking up at the surface. Rather than being able to see the world above, you might instead see a reflection, depending on the viewing angle.
4.2. LENSES

Figure 4.9: Propagating wavefronts from a medium with low refractive index (such as air) to one with a higher index (such as glass). (a) The effect of slower propagation on the wavefronts is shown as they enter the lower medium. (b) This shows the resulting bending of a light ray, which is always perpendicular to the wavefronts. Snell’s Law relates the refractive indices and angles as \( n_1 \sin \theta_1 = n_2 \sin \theta_2 \).

Prisms Imagine shining a laser beam through a prism, as shown in Figure 4.10. Snell’s Law can be applied to calculate how the light ray bends after it enters and exits the prism. Note that for the upright prism, a ray pointing slightly upward becomes bent downward. Recall that a larger refractive index inside the prism would cause greater bending. By placing the prism upside down, rays pointing slightly downward are bent upward. Once the refractive index is fixed, the bending depends only on the angles at which the rays enter and exist the surface, rather than the thickness of the prism. To construct a lens, we will exploit this principle and construct a kind of curved version of Figure 4.10.

Simple convex lens Figure 4.11 shows a simple convex lens, which should remind you of the prisms in Figure 4.10. Instead of making a diamond shape, the lens surface is spherically curved so that incoming, parallel, horizontal rays of light converge to a point on the other side of the lens. This special place of convergence is called the focal point. Its distance from the lens center is called the focal depth or focal length.

The incoming rays in Figure 4.11 are special in two ways: 1) They are parallel, thereby corresponding to a source that is infinitely far away, and 2) they are perpendicular to the plane in which the lens is centered. If the rays are parallel but not perpendicular to the lens plane, then the focal point shifts accordingly, as shown in Figure 4.12. In this case, the focal point is not on the optical axis. There are two DOFs of incoming ray directions, leading to a focal plane that contains all of the focal points. Unfortunately, this planarity is just an approximation; Section

Figure 4.10: The upper part shows how a simple prism bends ascending rays into descending rays, provided that the incoming ray slope is not too high. This was achieved by applying Snell’s Law at the incoming and outgoing boundaries. Placing the prism upside down causes descending rays to become ascending. Putting both of these together, we will see that a lens is like a stack of prisms that force diverging rays to converge through the power of refraction.

Figure 4.11: A simple convex lens causes parallel rays to converge at the focal point. The dashed line is the optical axis, which is perpendicular to the lens and pokes through its center.
4.2. LENSES

Figure 4.12: If the rays are not perpendicular to the lens, then the focal point is shifted away from the optical axis.

Figure 4.13: In the real world, an object is not infinitely far away. When placed at distance \( s_1 \) from the lens, a real image forms in a focal plane at distance \( s_2 > f \) behind the lens, as calculated using (4.5).

(4.3) explains what really happens. In this idealized setting, a real image is formed in the image plane, as if it were a projection screen that is showing how the world looks in front of the lens (assuming everything in the world is very far away).

If the rays are not parallel, then it may still be possible to focus them into a real image, as shown in Figure 4.13. Suppose that a lens is given that has focal length \( f \). If the light source is placed at distance \( s_1 \) from the lens, then the rays from that will be in focus if and only if the following equation is satisfied:

\[
\frac{1}{s_1} + \frac{1}{s_2} = \frac{1}{f}.
\]  

Figure 4.13 corresponds to the idealized case in which \( s_1 = \infty \), for which solving (4.5) yields \( s_2 = f \). What if the object being viewed is not completely flat and lying in a plane perpendicular to the lens? In this case, there does not exist a single plane behind the lens that would bring the entire object into focus. We must tolerate the fact that most of it will be approximately in focus. Unfortunately, this is the situation almost always encountered in the real world, including the focus provided by our own eyes (see Section 4.4).

If the light source is placed too close to the lens, then the outgoing rays might be diverging so much that the lens cannot force them to converge. If \( s_1 = f \), then the outgoing rays would be parallel (\( s_2 = \infty \)). If \( s_1 < f \), then (4.5) yields \( s_2 < 0 \).

Figure 4.14: If the object is very close to the lens, then the lens cannot force its outgoing light rays to converge to a focal point. In this case, however, a virtual image appears and the lens works as a magnifying glass. This is the way lenses are commonly used for VR headsets.

Figure 4.15: In the case of a concave lens, parallel rays are forced to diverge. The rays can be extended backward through the lens to arrive at a focal point on the left side. The usual sign convention is that \( f < 0 \) for concave lenses.

In this case, a real image is not formed; however, something interesting happens: The phenomenon of magnification. A virtual image appears when looking into the lens, as shown in Figure 4.14. This exactly what happens what happens in the case of the View-Master and the VR headsets that were shown in Figure 2.11. The screen is placed so that it appears magnified. To the user viewing looking through the screen, it appears as if the screen is infinitely far away (and quite enormous!).

**Lensmaker’s equation** For a given simple lens, the focal length \( f \) can be calculated using the Lensmaker’s Equation,

\[
(n_2 - n_1) \left( \frac{1}{r_1} + \frac{1}{r_2} \right) = \frac{1}{f},
\]  

(4.6)
which is derived from Snell’s law \[ n_1 \sin \theta_1 = n_2 \sin \theta_2 \]. The parameters \( r_1 \) and \( r_2 \) represent the radius of curvature of each of the two lens surfaces (front and back). This version assumes a thin lens approximation, which means that the lens thickness is small relative to \( r_1 \) and \( r_2 \). Also, it is typically assumed that \( n_1 = 1 \), which is approximately true for air.

**Concave lenses** For the sake of completeness, we include the case of a concave simple lens, shown in Figure 4.15. Parallel rays are forced to diverge, rather than converge; however, a meaningful notion of negative focal length exists by tracing the diverging rays backwards through the lens. The Lensmaker’s Equation (4.3) can be slightly adapted to calculate negative \( f \) in this case \[ f = \frac{1}{D} \].

**Diopters** For optical systems used in VR, several lenses will be combined in succession. What is the effect of the combination? A convenient method to answer this question with simple arithmetic was invented by ophthalmologists. The idea is to define a dioptr, which is \( D = 1/f \). Thus, it is the reciprocal of the focal length. If a lens focuses parallel rays at a distance of 0.2m in behind the lens, then \( D = 5 \). A larger dioptr \( D \) means greater converging power. Likewise, a concave lens yields \( D < 0 \) by using a negative lens. To combine several lenses in succession, we simply add their dioplers to determine their equivalent power as a single, simple lens. Figure 4.16 shows a simple example.

### 4.3 Optical Aberrations

If lenses in the real world behaved exactly as described in Section 4.2, then VR systems would be much simpler and more impressive than they are today. Unfortunately, numerous imperfections, called aberrations, degrade the images formed by lenses. Because these problems are perceptible in everyday uses, such as viewing content through VR headsets or images from cameras, they are important to understand so that some compensation for them can be designed into the VR system or content.

- **Chromatic aberration** Recall from Section 4.1 that light energy is usually a jumble of waves with a spectrum of wavelengths. You have probably seen that the colors of the entire visible spectrum nicely separate when white light is shined through a prism. This is a beautiful phenomenon, but for lenses it is terrible annoyance because it separates the focused image based on color. This problem is called chromatic aberration.

  The problem is that the speed of light through a medium depends on the wavelength. We therefore should write a material’s refractive index as \( n(\lambda) \) to indicate that it is a function of \( \lambda \). Figure 4.17 shows the effect on a simple convex lens. The focal depth becomes a function of wavelength. If we shine red, green, and blue lasers directly into the lens along the same ray, then each color would cross the optical axis in a different place, resulting in red, green, and blue focal points. Recall the spectral power distribution and reflection functions from Section 4.1. For common light sources and materials, the light passing through a lens results in a whole continuum of focal points. Figure 4.18 shows an image with chromatic aberration artifacts. Chromatic aberration can be reduced at greater expense by combining convex and concave lenses of different materials so that the spreading rays are partly coerced into converging.

- **Spherical aberration** Figure 4.19 shows spherical aberration, which is caused by rays further away from the lens center being refracted more than rays near the center. The result is similar to that of chromatic aberration, but this phenomenon is a monochromatic aberration because it is independent of the light wavelength. Incoming parallel rays are focused at varying depths, rather than being concentrated at a single point. The result is some blur that cannot be compensated by moving the object, lens, or image plane. Alternatively, the image might instead focus onto a curved surface, called the Petzval surface, rather then the image plane. This aberration arises due to the spherical shape of the lens. An aspheric lens is more complex and has non-spherical surfaces that are designed to specifically
4.3. OPTICAL ABERRATIONS

Figure 4.18: The upper image is properly focused whereas the lower image suffers from chromatic aberration. (Figure by Stan Zurek.)

Figure 4.19: Spherical aberration causes imperfect focus because rays away from the optical axis are refracted more than those at the periphery.

Optical distortion

Even if the image itself projects onto the image plane it might be distorted at the periphery. Assuming that the lens is radially symmetric, the distortion can be described as a stretching or compression of the image that becomes increasingly severe away from the optical axis. Figure 4.20 shows how this affects the image for two opposite cases: barrel distortion and pincushion distortion. For lenses that have a wide field-of-view, the distortion is stronger, especially in the extreme case of a fish-eyed lens. Figure 4.21 shows an image that has strong barrel distortion. Correcting this distortion is an important component of VR headsets; otherwise, the virtual world would appear to be warped.

Astigmatism

Figure 4.22 depicts astigmatism, which is a lens aberration that occurs for incoming rays that are not perpendicular to the lens. Up until now, our lens drawings have been 2D; however, a third dimension is needed to understand this new aberration. The rays can be off-axis in one dimension, but aligned in another. By moving the image plane along the optical axis, it becomes impossible to bring the image into focus. Instead, horizontal and vertical focal depths appear,
4.3 OPTICAL ABERRATIONS

Figure 4.21: An image with barrel distortion, taken by a fish-eyed lens. (Image by Wikipedia user Ilveon.)

Figure 4.22: Astigmatism is primarily caused by incoming rays being off-axis in one plane, but close to perpendicular in another. (Figure from [70].)

Coma Finally, coma is yet another aberration. In this case, the image magnification varies dramatically as the rays are far from perpendicular to the lens. The result is a “comet” pattern in the image plane. You might have seen this while tilting a lens outside and observing bright disc patterns produced by direct sunlight. All of the aberrations of this section complicate the system or degrade the experience in a VR headset; therefore, substantial engineering effort is spent on mitigating these problems.

4.4 The Human Eye

We have covered enough concepts in this chapter to describe the basic operation of the human eye, which is clearly an important component in any VR system. Here it will be considered as part of an optical system of lenses and images. The physiological and perceptual parts of human vision are deferred until Chapter 5.

Figure 4.24 shows a cross section of the human eye facing left. Parallel light rays are shown entering from the left; compare to Figure 4.11, which showed a similar situation for an engineered convex lens. Although the eye operation is similar to the engineered setting, several important differences arise at this stage. The focal plane is replaced by a spherically curved surface called the retina. The retina contains photoreceptors that convert the light into neural pulses; this is covered in Sections 5.3 and 5.4. The interior of the eyeball is actually liquid, as opposed to air. The refractive indices of materials along the path from the outside air to the retina are shown in Figure 4.25.
4.4. THE HUMAN EYE

The human eye can be viewed as a highly sophisticated optical system that allows us to see the world around us. It comprises several key components, including the cornea, lens, aqueous fluid, vitreous fluid, and retina. Each of these components plays a crucial role in the process of vision.

![Image Surface (Non-Planar)](image)

**Figure 4.24:** A simplified view of the human eye as an optical system.

A ray of light travels through five media before hitting the retina: air, cornea, aqueous fluid, lens, and vitreous fluid. The indices of refraction for these media are indicated in the diagram.

1. $n_1 = 1.000$ (air)
2. $n_2 = 1.376$ (cornea)
3. $n_3 = 1.336$ (aqueous fluid)
4. $n_4 = 1.413$ (lens)
5. $n_5 = 1.337$ (vitreous fluid)

![Light Rays](light-rays)

**Figure 4.25:** A ray of light travels through five media before hitting the retina. The indices of refraction are indicated. Considering Snell’s law, the greatest bending occurs due to the transition from air to the cornea. Note that once the ray enters the eye, it passes through only liquid or solid materials.

**The optical power of the eye**

The outer diameter of the eyeball is roughly 24mm, which implies that a lens of at least 40D would be required to cause convergence of parallel rays onto the retina center inside of the eye (recall diopters from Section 4.2). There are effectively two convex lenses: The cornea and the lens. The cornea is the outermost part of the eye where the light first enters and has the greatest optical power, approximately 40D. The eye lens is less powerful and provides an additional 20D. By adding diopters, the combined power of the cornea and lens is 60D, which means that parallel rays are focused onto the retina at a distance of roughly 17mm from the outer cornea. Figure 4.26 shows how this system acts on parallel rays for a human with normal vision. Images of far away objects are thereby focused onto the retina.

**Figure 4.26:** Normal eye operation, with relaxed lens.

**Figure 4.27:** A closer object yields diverging rays, but with a relaxed lens, the image is blurry on the retina.

**Accommodation**

What happens when we want to focus on a nearby object, rather than one “infinitely far” away? Without any changes to the optical system, the image would be blurry on the retina, as shown in Figure 4.27. Fortunately, and miraculously, the lens changes its diopter to accommodate the closer distance. This process is appropriately called *accommodation*, as is depicted in Figure 4.28. The diopter change is effected through muscles that pull on the lens to change its shape. In young children, the lens can increase its power by an additional 15 to 20D, which explains why a child might hold something right in front of your face and expect you to focus on it; they can! At 20D, this corresponds to focusing on...
4.4. THE HUMAN EYE

Figure 4.28: The process of accommodation: The eye muscles pull on the lens, causing it to increase the total optical power and focus the image on the retina.

Figure 4.29: Placing a convex lens in front of the eye is another way to increase the optical power so that nearby objects can be brought into focus by the eye. This is the principle of reading glasses.

an object that is only 50cm from the cornea. Young adults already lose this ability and can accommodate up to about 10D. Thus, with normal vision they can read a book down to a distance of about 10cm (with some eye strain). Once adults reach 50 years old, little or no accommodation ability remains. This condition is called presbyopia. Figure 4.29 shows the most common treatment, which is to place reading glasses in front of the eye.

Vision abnormalities. The situations presented so far represent normal vision throughout a person’s lifetime. One problem could be that the optical system simply does not have enough optical power to converge parallel rays onto the retina. This condition is called hyperopia or farsightedness. Eyeglasses come to the rescue. The simple fix is to place a convex lens (positive dipter) in front of the eye, as in the case of reading glasses. In the opposite direction, some eyes have too much optical power. This case is called myopia or nearsightedness, and a concave lens (negative dipter) is placed in front of the eye to reduce the optical power appropriately. Recall that we have two eyes, not one. This allows the possibility for each eye to have a different problem, resulting in different lens dioplers per eye. Other vision problems may exist beyond optical power. The most common is astigmatism, which was covered in Section 4.3. In human eyes this is caused by the eyeball having an excessively elliptical shape, rather than being perfectly spherical. Specialized, non-simple lenses are needed to correct this condition. You might also wonder whether the aberrations from Section 4.3 occur in the human eye. They do, however, the problems, such as chromatic aberration, are corrected automatically by our brains because we have learned to interpret such flawed images our entire lives.

A simple VR headset. Now suppose we are constructing a VR headset by placing a screen very close to the eyes. Young adults would already be unable to bring it into focus if it were closer than 10cm. We want to bring it close so that it fills the view of the user. Therefore, the optical power is increased by using a convex lens, functioning in the same way as reading glasses. See Figure 4.30. This is also the process of magnification, from Section 4.2. The lens is placed at the distance of its focal depth. Using (4.5), this implies that \( s_2 = -f \), resulting in \( s_1 = \infty \). The screen appears as an enormous virtual image that is infinitely far away. Note, however, that a real image is nevertheless projected onto the retina. We do not perceive the world around us unless real images are formed on our retinas!

To account for people with vision problems, a focusing knob may be appear on the headset, which varies the distance between the lens and the screen. This adjusts the optical power so that the rays between the lens and the cornea are no longer parallel. They can be made to converge, which helps people with hyperopia. Alternatively, they can be made to diverge, which helps people with myopia. Thus, they can focus sharply on the screen without placing their eyeglasses in front of the lens. However, if each eye requires a different diopler, then a focusing knob
4.5. CAMERAS

would be required for each eye. Furthermore, if they have an astigmatism, then it cannot be corrected. Placing eyeglasses inside of the headset may be the only remaining solution, but it may be uncomfortable and will reduce the field of view.

Many details have been skipped or dramatically simplified in this section. One important detail for a VR headset is each lens should be centered perfectly in front of the cornea. If the distance between the two lenses is permanently fixed, then this is impossible to achieve for everyone who uses the headset. The interpupillary distance, or IPD, is the distance between human eye centers. The average among humans is around 64mm, but it varies greatly by race, gender, and age (in the case of children). To be able to center the lenses for everyone, the distance between lens centers should be adjustable from around 55 to 75mm. This is a common range for binoculars. Unfortunately, the situation is not even this simple because our eyes also rotate within their sockets, which changes the position and orientation of the cornea with respect to the lens. This amplifies optical aberration problems that were covered in Section 4.3. Eye movements will be covered in Section 5.3. Another important detail is the fidelity of our vision: What pixel density is needed for the screen that is placed in front of our eyes so that we do not notice the pixels? A similar question is how many dots-per-inch (DPI) are needed on a printed piece of paper so that we do not see the dots, even when viewed under a magnifying glass? We return to this question in Section 5.4.

Figure 4.30: In VR headsets, the lens is placed so that the screen appears to be infinitely far away.

Figure 4.31: A pinhole camera that is recommended for viewing a solar eclipse. (Figure from TimeAndDate.com.)

Figure 4.32: (a) A CMOS active-pixel image sensor. (b) A low-cost CMOS camera module (SEN-11745), ready for hobbyist projects.
4.5 Cameras

Now that we have covered the human eye, it seems natural to describe an engineered eye, otherwise known as a camera. People have built and used cameras for hundreds of years, starting with a camera obscura that allows light to pass through a pinhole and onto a surface that contains the real image. Figure 4.31 shows an example that you might have constructed to view a solar eclipse. (Recall the perspective transformation math from Section 3.4.) Eighteenth-century artists incorporated a mirror and tracing paper to un-invert the image and allow it to be perfectly copied. Across the 19th century, various chemically based technologies were developed to etch the image automatically from the photons hitting the imaging surface. Across the 20th century, film was in widespread use, until digital cameras avoided the etching process altogether by electronically capturing the image using a sensor. Two popular technologies have been a Charge-Coupled Device (CCD) array and a CMOS active-pixel image sensor, which is shown in Figure 4.32(a). Such digital technologies record the amount of light hitting each pixel location along the image, which directly produces a captured image. The costs of these devices has plummeted in recent years, allowing hobbyists to buy a camera module such as the one shown in Figure 4.32(b) for under $30 US.

Shutters Several practical issues arise when capturing digital images. The image is an 2D array of pixels, each of which having red (R), green (G), and blue (B) values that typically range from 0 to 255. Consider the total amount of light energy that hits the image plane. For a higher-resolution camera, there will generally be less photons per pixel because the pixels are smaller. Each sensing element (one per color per pixel) can be imagined as a bucket that collects photons, much like drops of rain. To control the amount of photons, a shutter blocks all the light, opens for a fixed interval of time, and then closes again. For a long interval (low shutter speed), more light is collected; however, the drawbacks are that moving objects in the scene will become blurry and that the sensing elements could become saturated with too much light. Photographers must strike a balance when determining the shutter speed to account for the amount of light in the scene, the sensitivity of the sensing elements, and the motion of the camera and objects in the scene.

Also relating to shutters, CMOS sensors unfortunately work by sending out the image information sequentially, line-by-line. The sensor is therefore coupled with a rolling shutter, which allows light to enter for each line, just before the information is sent. This means that the capture is not synchronized over the entire image, which leads to odd artifacts, such as the one shown in Figure 4.33. Image processing algorithms that work with rolling shutters and motion typically transform the image to correct for this problem. CCD sensors grab and send the entire image at once, resulting in a global shutter. Unfortunately, CCDs are more expensive than CMOS sensors, which has resulted in widespread appearance of rolling shutter cameras in smartphones.

Figure 4.33: The wings of a flying helicopter are apparently bent backwards due to the rolling shutter effect.

Figure 4.34: A spectrum of aperture settings, which control the amount of light that enters the lens. The values shown are called the focal ratio or f-stop.
Aperture  The optical system also impacts the amount of light. Using a pinhole, as shown in Figure 4.31, light would fall onto the image sensor, but it would not be bright enough for most purposes (other than viewing a solar eclipse). Therefore, a convex lens is used instead so that multiple rays are converged to the same point in the image plane; recall Figure 4.11. This generates more photons per sensing element. The main drawback is that the lens sharply focuses objects at a single depth, while blurring others; recall (4.5). In the pinhole case, all depths are essentially “in focus”, but there might not be enough light. Photographers therefore want to tune the optical system to behave more like a pinhole or more like a full lens, depending on the desired outcome. This result is a controllable aperture (Figure 4.34), which appears behind the lens and sets the size of the hole through which the light rays enter. A small radius mimics a pinhole by blocking all but the center of the lens. A large radius allows light to pass through the entire lens. Our eyes control the light levels in a similar manner by contracting or dilating our pupils. Finally, note that the larger the aperture, the more that the aberrations covered in Section 4.3 affect the imaging process.

Further Reading
A classic, popular text on optical engineering: [68].
Chapter 5
The Physiology of Human Vision

What you perceive about the world around you is “all in your head”. After reading Chapter 4, especially Section 4.4, you should understand that the light around us forms images on our retinas that capture colors, motions, and spatial relationships in the physical world. For someone with normal vision, these captured images may appear to have perfect clarity, speed, accuracy, and resolution, while being distributed over a large field of view. However, we are being fooled. We will see in this chapter that this apparent perfection of our vision is mostly an illusion because neural structures are filling in plausible details to generate a coherent picture in our heads that is consistent with our life experiences. When building VR technology that co-opts these processes, it important to understand how they work. They were designed to do more with less, and fooling these processes with VR produces many unexpected side effects because the display technology is not a perfect replica of the surrounding world.

Section 5.1 continues where Section 4.4 left off by adding some biology of the human eye to the optical system. Most of the section is on photoreceptors, which are the “input pixels” that get paired with the “output pixels” of a digital display for VR. Section 5.2 offers a taste of neuroscience by explaining what is known about the visual information that hierarchically propagates from the photoreceptors up to the visual cortex. Section 5.3 explains how our eyes move, which serves a good purpose, but incessantly interferes with the images in our retinas. Section 5.4 concludes the chapter by applying the knowledge gained about visual physiology to determine VR display requirements, such as the screen resolution.

5.1 From the Cornea to Photoreceptors

Parts of the eye  Figure 5.1 shows the physiology of a human eye. The shape is approximately spherical, with a diameter of around 24mm and only slight variation among people. The cornea is a hard, transparent surface through which light enters and provides the greatest optical power (recall from Section 4.4). The rest of the outer surface of the eye is protected by a hard, white layer called the sclera. Most of the eye interior consists of vitreous humor, which is a transparent, gelatinous mass that allows light rays to penetrate with little distortion or attenuation.

As light rays cross the cornea, they pass through a small chamber containing aqueous humour, which is another transparent, gelatinous mass. After crossing this, rays enter the lens by passing through the pupil. The size of the pupil is controlled by a disc-shaped structure called the iris, which provides an aperture that regulates the amount of light that is allowed to pass. The optical power of the lens is altered by ciliary muscles. After passing through the lens, rays pass through...
5.1. FROM THE CORNEA TO PHOTORECEPTORS

the vitreous humor and strike the retina which lines more than 180° of the inner eye boundary. Since Figure 5.1 shows a 2D cross section, the retina looks more like an arc; however, keep in mind that it is a 2D surface. Imagine it as curved counterpart to a visual display. To catch the light from the output pixels, it is lined with photoreceptors, which behave like “input pixels”. The most important part of the retina is the fovea; the highest visual acuity, which is a measure of the sharpness or clarity of vision, is provided for rays that land on it. The optic disc is a small hole in the retina through which neural pulses are transmitted outside of the eye through the optic nerve. It is on the same side of the fovea as the nose.

Photoreceptors The retina contains two kinds of photoreceptors for vision: 1) rods, which are triggered by very low levels of light, and 2) cones, which require more light and are designed to distinguish between colors. See Figure 5.2. Each human retina contains about 120 million rods and 6 million cones. Figure 5.3 shows the detection capabilities of each photoreceptor type. Rod sensitivity peaks at 498nm, between blue and green in the spectrum. There are three categories of cones, based on whether they are designed to sense blue, green, or red light.

Photoreceptors respond to light levels over a large dynamic range. Figure 5.4 shows several familiar examples. The luminance is measured in SI units of candelas per square meter, which corresponds directly to the amount of light power per area.
5.1 FROM THE CORNEA TO PHOTORECEPTORS

Figure 5.5: Photoreceptor density as a function of angle. The right of the plot is the nasal side (which corresponds to rays entering from the opposite, temporal side). (Figure based on [56])

The range spans seven orders of magnitude, from 1 photon hitting a photoreceptor every 100 seconds up to 100,000 photons per receptor per second. At low light levels, only rods are triggered. Our inability to distinguish colors at night is caused by the inability of rods to distinguish colors. Our eyes may take up to 35 minutes to fully adapt to low light, resulting in a monochromatic mode called scotopic vision. By contrast, our cones become active in brighter light. Adaptation to this trichromatic mode, called photopic vision, may take up to ten minutes (you have undoubtedly noticed the adjustment period when someone unexpectedly turns on lights while you are lying in bed at night).

Photoreceptor density  The density of photoreceptors across the retina varies greatly, as plotted in Figure 5.5. The most interesting region is the fovea, which has the greatest concentration of photoreceptors. The innermost part of the fovea has a diameter of only 0.5mm or an angular range of ±0.85 degrees, and contains almost entirely cones. This implies that the eye must be pointed straight at a target to perceive a sharp, colored image. The entire fovea has diameter 1.5mm (±2.6 degrees angular range), with the outer ring having a dominant concentration of rods. Rays that enter the cornea from the sides land on parts of the retina with lower rod density and very low cone density. This corresponds to the case of peripheral vision. We are much better at detecting movement in our periphery, but cannot distinguish colors effectively. Peripheral movement detection may have helped our ancestors from being eaten by predators. Finally, the most intriguing part of the plot is the blind spot, where there are no photoreceptors. This is due to our retinas being inside-out and having no other way to route the neural signals to the brain; see Section 5.2.

The photoreceptor densities shown in Figure 5.5 leave us with a conundrum. With 20/20 vision, we perceive the world as if our eyes are capturing a sharp, colorful image over a huge angular range. This seems impossible, however, because we can only sense sharp, colored images in a narrow range. Furthermore, the blind spot should place a black hole in our image. Surprisingly, our perceptual processes produce an illusion that a complete image is being captured. This is accomplished by filling in the missing details using contextual information, which is described in Section 5.2 and by frequent eye movements, the subject of Section 5.3. If you are still not convinced that your brain is fooling you into seeing a complete image, try the blind spot experiment shown in Figure 5.6.

5.2 From Photoreceptors to the Visual Cortex

Photoreceptors are transducers that convert the light-energy stimulus into an electrical signal called a neural impulse, thereby inserting information about the outside world into our neural structures. Recall from Section 2.3 that signals are propagated upward in a hierarchical manner, from photoreceptors to the visual cortex (Figure 2.19). Think about the influence that each photoreceptor has on the network of neurons. Figure 5.7 shows a simplified model. As the levels increase, the number of influenced neurons grows rapidly. Figure 5.8 shows the same diagram, but highlighted in a different way by showing how the number of photoreceptors that influence a single neuron increases with level. Neurons at the lowest levels are able to make simple comparisons of signals from neighboring photoreceptors. As the levels increase, the neurons may respond to a larger patch of the retinal image. This principle will become clear when seeing more neural structures in this section. Eventually, when signals reach the highest levels (beyond these figures), information from the memory of a lifetime of experiences is fused with the information that propagated up from photoreceptors. As the brain performs significant perceptual processing, a perceptual phenomenon results, such as recognizing a face or judging the size of a tree. It takes the brain over 100ms to produce a result that enters our consciousness.
5.2 FROM PHOTORECEPTORS TO THE VISUAL CORTEX

Level 3: Neural Cells
Level 2: Neural Cells
Level 1: Neural Cells
Level 0: Photoreceptors

Figure 5.7: Four levels in a simple hierarchy are shown. Each disk corresponds to a neural cell or photoreceptor, and the arrows indicate the flow of information. Photoreceptors generate information at Level 0. In this extremely simplified and idealized view, each photoreceptor and neuron connects to exactly three others at the next level. The red and gold part highlights the growing zone of influence that a single photoreceptor can have as the levels increase.

Level 3: Neural Cells
Level 2: Neural Cells
Level 1: Neural Cells
Level 0: Photoreceptors

Figure 5.8: This diagram is the same as Figure 5.7 except that the information feeding into a single neuron is highlighted. Consider the set of photoreceptors involved in the reaction of a single neural cell. This is called the receptive field. As the level increases, the receptive field size grows dramatically. Due to the spatial arrangement of the photoreceptors, this will imply that each neuron responds to a growing patch in the image on the retina. The patch increases in size at higher levels.

Figure 5.9: Light passes through a few neural layers before hitting the rods and cones. (Figure by the Institute for Dynamic Educational Advancement.)
5.2. FROM PHOTORECEPTORS TO THE VISUAL CORTEX

Figure 5.10: Vertebrates (including humans) have inside-out retinas, which lead to a blind spot and photoreceptors aimed away from the incoming light. The left shows a vertebrate eye, and the right shows a cephalopod eye, for which nature got it right. The photoreceptors face the light and there is no blind spot. (Figure by Jerry Crimson Mann.)

Now consider the first layers of neurons in more detail, as shown in Figure 5.9. The information is sent from right to left, passing from the rods and cones to the bipolar, amacrine, and horizontal cells. These three types of cells are in the inner nuclear layer. From there, the signals reach the ganglion cells, which form the ganglion cell layer. Note that the light appears to be entering from the wrong direction: It passes over these neural cells before reaching the photoreceptors. This is due to the fact that the human retina is inside-out, as shown in Figure 5.10. Evolution got it right with octopuses and other cephalopods, for which the light directly reaches the photoreceptors. One consequence of an inside-out retina is that the axons of the ganglion cells cannot be directly connected to the optic nerve (item 3 in Figure 5.10), which sends the signals outside of the eye. Therefore, a hole has been punctured in our retinas so that the “cables” from the ganglion cells can be routed outside of the eye (item 4 in Figure 5.10). This causes the blind spot that was illustrated in Figure 5.6.

Upon studying Figure 5.9 closely, it becomes clear that the neural cells are not arranged in the ideal way of Figure 5.8. The bipolar cells transmit signals from the photoreceptors to the ganglion cells. Some bipolars connect only to cones, with the number being between 1 and 10 per bipolar. Others connect only to rods, with about 30 to 50 rods per bipolar. There are two types of bipolar cells based on their function. An ON bipolar activates when the rate of photon absorption in its connected photoreceptors increases. An OFF bipolar activates for decreasing photon absorption. The bipolars connected to cones have both kinds; however, the bipolars for rods have only ON bipolars. The bipolar connections are considered to be vertical because they connect directly from photoreceptors to the ganglion cells. This is in contrast to the remaining two cell types in the inner nuclear layer. The horizontal cells are connected by inputs (dendrites) to photoreceptors and bipolar cells within a radius of up to 1mm. Their output (axon) is fed into photoreceptors, causing lateral inhibition, which means that the activation of one photoreceptor tends to decrease the activation of its neighbors. Finally, amacrine cells connect horizontally between bipolar cells, other amacrine cells, and vertically to ganglion cells. There are dozens of types, and their function is not well understood. Thus, scientists do not have a complete understanding of human vision, even at the lowest layers. Nevertheless, the well understood parts contribute greatly to our ability to design effective VR systems and predict other human responses to visual stimuli.

At the ganglion cell layer, several kinds of cells process portions of the retinal image. Each ganglion cell has a large receptive field, which corresponds to the photoreceptors that contribute to its activation as shown in Figure 5.8. The three most common and well understood types of ganglion cells are called midget, parasol, and bistratified. They perform simple filtering operations over their receptive fields based on spatial, temporal, and spectral (color) variations in the stimulus across the photoreceptors. Figure 5.11 shows one example. In this case, a ganglion cell is triggered when red is detected in the center but not green in the surrounding area. This condition is an example of spatial opponency, for which neural struc-
5.3. Eye Movements

Eye rotations are a complicated and integral part of human vision. They occur both voluntarily and involuntarily, and allow humans to fixate on features in the world, even as the head or target features are moving. One of the main reasons for movement is to position the feature of interest on the fovea. Recall from Section 5.2 that only the fovea can sense dense, color images, and it unfortunately spans a very narrow field of view. To gain a coherent, detailed view of a large object,
the eyes rapidly scan over it while fixating on points of interest. Figure 5.13 shows an example. Another reason for eye movement is that our photoreceptors are slow to respond to stimuli due to their chemical nature. They take up to 10ms to fully respond to stimuli and produce a response for up to 100ms. Eye movements help keep the image fixed on the same set of photoreceptors so that they can fully charge. This is similar to the image blurring problem that occurs in cameras at low light levels and slow shutter speeds. Additional reasons for eye movement are to maintain a stereoscopic view and to prevent adaptation to a constant stimulation. To support the last claim, it has been shown experimentally that when eye motions are completely suppressed, visual perception disappears completely [25]. As movements combine to build a coherent view, it is difficult for scientists to predict and explain how we will interpret some stimuli. For example, the optical illusion in Figure 5.16 appears to be moving when our eyes scan over it.

Eye muscles The rotation of each eye is controlled by six muscles that are each attached to the sclera (outer eyeball surface) by a tendon. Figures 5.17 and 5.18 show their names and arrangement. The tendons pull on the eye in opposite pairs. For example, to perform a yaw (side-to-side) rotation, the tensions on the medial rectus and lateral rectus are varied while the other muscles are largely unaffected. To cause a pitch motion, four muscles per eye become involved. All six are involved to perform both a pitch and yaw, for example, looking upward and to the right. A small amount of roll motion can be generated; however, our eyes are generally not designed for much roll motion. Imagine if you could turn your eyeballs upside-down inside of their sockets! Thus, it is reasonable in most cases to approximate eye rotations as a 2D set that includes only yaw and pitch, rather than the full 3 DOFs obtained...
5.3. EYE MOVEMENTS

Types of movements We now consider movements based on their purpose, resulting in six categories: 1) saccades, 2) smooth pursuit, 3) vestibulo-ocular reflex, 4) optokinetic reflex, 5) vergence, and 6) microsaccades. All of motions cause both eyes to rotate approximately the same way, except for vergence, which causes the eyes to rotate in opposite directions. We will skip a seventh category of motion, called rapid eye movements (REMs), because they only occur while we are sleeping and therefore do not contribute to a VR experience. The remaining six categories will now be discussed in detail.

Saccades The eye can move in a rapid motion called a saccade, which lasts less than 45ms and rotates at a rate of about $900^\circ$ per second. The purpose is to quickly relocate the fovea so that important features in a scene are sensed with highest visual acuity. Figure 5.15 showed an example in which a face is scanned by fixating on various features in rapid succession. Each transition between features is accomplished by a saccade. Interestingly, our brains use saccadic masking to hide the intervals of time over which saccades occur from our memory. This results in distorted time perception, as in the case when second hands click into position on an analog clock. The result of saccades is that we obtain the illusion of high acuity over a large angular range. Although saccades frequently occur while we have little or no awareness of them, we have the ability to consciously control them as we choose features for fixation.

Smooth pursuit In the case of smooth pursuit, the eye slowly rotates to track a moving target feature. Examples are a car, a tennis ball, or a person walking by. The rate of rotation usually less than $30^\circ$ per second, which is much slower than for saccades. The main function of smooth pursuit is to reduce motion blur on the retina; this is also known as image stabilization. The blur is due to the slow response time of photoreceptors, as discussed in Section 5.1. If the target is moving too fast, then saccades may be intermittently inserted into the pursuit motions to catch up to it.

Vestibulo-ocular reflex One of the most important motions to understand for VR is the vestibulo-ocular reflex or VOR. Hold your finger at a comfortable distance in front of your face. Next, yawn your head back and forth (like you are nodding “no”), turning about 20 or 30 degrees to the left and right sides each time. You may notice that your eyes are effortlessly rotating to counteract the rotation of your head so that your finger remains in view. The eye motion is involuntary. If you do not believe it, then try to avoid rotating your eyes while paying attention to your finger and rotating your head. It is called a reflex because the motion control bypasses higher brain functions. Figure 5.10 shows how this circuitry works. Based on angular accelerations sensed by our vestibular organs, signals are sent to the eye muscles to provide the appropriate counter motion.
5.3. EYE MOVEMENTS

Figure 5.19: The vestibulo-ocular reflex (VOR). The eye muscles are wired to angular accelerometers in the vestibular organ to counter head movement with the opposite eye movement with less than 10ms of latency. The connection between the eyes and vestibular organ is provided by specialized vestibular and extraocular motor nuclei, thereby bypassing higher brain functions.

Figure 5.20: In the process of stereopsis, both eyes are fixated on the same feature in the world. To transition from a close to far feature, a divergence motion occurs. A convergence motion happens for the opposite transition.

The main purpose of the VOR is to provide image stabilization, as in the case of smooth pursuit. For more details about the vestibular organ, see Section 8.3.

**Optokinetic reflex** The next category is called the optokinetic reflex, which occurs when a fast object speeds along. This occurs when watching a fast-moving train while standing nearby on fixed ground. The eyes rapidly and involuntarily choose features for tracking on the object, while alternating between smooth pursuit and saccade motions.

**Vergence** Stereopsis refers to the case in which both eyes are fixated on the same object, resulting in a single perceived image. Two kinds of vergence motions occur to align the eyes with an object. See Figure 5.20. If the object is closer than a previous fixation, then a convergence motion occurs. This means that the eyes are rotating so that the pupils are becoming closer. If the object is further, then divergence motion occurs, which causes the pupils to move further apart. The eye orientations resulting from vergence motions provide important information about the distance of objects.

**Microsaccades** The sixth category of movements is called microsaccades, which are small, involuntary jerks of less than one degree that trace out an erratic path. They are believed to augment many other processes, including control of fixations, reduction of perceptual fading due to adaptation, improvement of visual acuity, and resolving perceptual ambiguities [59]. Although these motions have been known since the 18th century [14], their behavior is extremely complex and not fully understood. Microsaccades are an active topic of research in perceptual psychology, biology, and neuroscience.

**Eye and head movements together** Although this section has focused on eye movement, it is important to understand that most of the time the eyes and head are moving together. Figure 5.21 shows the angular range for yaw rotations of the head and eyes. Although eye yaw is symmetric by allowing 35° to the left or right,
5.4 Implications for VR

This chapter has so far covered the human hardware for vision. Basic physiological properties, such as photoreceptor density or VOR circuitry directly impact the engineering requirements for visual display hardware. The engineered systems must be good enough to adequately fool our senses, but they need not have levels of quality that are well beyond the limits of our receptors. Thus, the VR display should ideally be designed to perfectly match the performance of the sense it is trying to fool.

How good does the VR visual display need to be? Three crucial factors for the display are:

1. **Spatial resolution**: How many pixels per square area are needed?
2. **Intensity resolution and range**: How many intensity values can be produced, and what are the minimum and maximum intensity values?
3. **Temporal resolution**: How fast do displays need to change their pixels?

The spatial resolution factor will be addressed in the next paragraph. The second factor could also be called color resolution and range because the intensity values of each red, green, or blue subpixel produce points in the space of colors; see Section 6.3. Recall the range of intensities from Figure 5.4 that trigger photoreceptors. Photoreceptors can span seven orders of magnitude of light intensity. However, displays have only 256 intensity levels per color to cover this range. Entering scotopic vision mode does not even seem possible using current display technology because of the high intensity resolution needed at extremely low light levels. Temporal resolution is extremely important, but is deferred until Section 6.2 in the context of motion perception.
5.4. IMPLICATIONS FOR VR

Figure 5.24: Red, green, and blue cone photoreceptors are distributed in a complicated mosaic in the center of the fovea. (Figure by Mark Fairchild.)

We now address the spatial resolution. Insights into needed spatial resolution are obtained from the photoreceptor densities. As shown in Figure 5.22, we see individual lights when a display is highly magnified. As it is zoomed out, we may still perceive sharp diagonal lines as being jagged, as shown in Figure 5.23(a); aliasing this phenomenon is known as 

Another artifact is the screen-door effect shown in Figure 5.23(b); this is commonly noticed in an image produced by a digital LCD projector. What does the display pixel density need to be so that we do not perceive individual pixels? In 2010, Steve Jobs of Apple Inc. claimed that 2326 pixels per inch (165 pixels per mm) is enough, achieving what they called a retina display.

Assume that the fovea is pointed directly at the display to provide the best sensing possible. The first issue is that red, green, and blue cones are arranged in a mosaic, as shown in Figure 5.24. The patterns are more erratic than the engineered versions in Figure 5.22. Vision scientists and neurobiologists have studied the effective input resolution through measures of visual acuity. One measure is cycles per degree, which roughly corresponds to the number of stripes that can be seen as separate along a viewing arc; see Figure 5.20. This has been estimated to be around 77 per degree, based on photoreceptor density and neural processes. Supposing that one cycle is equivalent to the distance between pixel centers, this implies that $(77)^2 = 5929$ pixels should exist over a square area of one degree by one degree.

Using simple trigonometry,

$$s = d \tan \theta,$$

we can determine what the span $s$ should be for a viewing angle $\theta$ at a distance $d$ from the eye. For very small $\theta$, $\tan \theta \approx \theta$ (in radians). If a smartphone screen is placed 30cm in front of an eye, then $s = 5.236$mm per degree. This means that 5929 pixels should be distributed over $(5.236)^2 = 27.416$mm, which implies that the display should have at least 216.3 pixels per one mm to be a retina display.

The density can be significantly lower and nevertheless be a retina display due to optical aberrations in the human eye, thereby making Apple’s claim of 165 reasonable. Instead of 77 cycles per degree, 60 cycles per degree is commonly used to account for this, which would require only 131.3 pixels per mm.

In the case of VR, we are not looking directly at a screen as in the case of smartphones. If we bring the screen up to 10cm from the eye, then $s = 1.745$mm per degree and the required density increases (using 77 cycles per degree) to 1946 pixels per mm; however, it becomes uncomfortable or impossible for people to focus on it at such close range. By inserting a lens for magnification, the display can be brought even closer to the eye. This is commonly done for VR headsets, as was shown in Figure 4.30. Suppose the lens is positioned at its focal distance away $s = 0.698$mm and 12165 pixels per mm would be needed. For comparison, the most dense smartphone screen available today is the Super AMOLED 1440x2560 5.1 inch screen on the Samsung S6. It has about 516 pixels per mm; however, the requirement for an effective retina display would be 2801 pixels per inch. Using 60 cycles per degree, it would be only 7386 pixels per mm or 2183 pixels per inch. Thus, to approach the threshold for a VR retina display using a screen and simple lenses, the pixel density needs to be about four times higher in each dimension.

How much FOV is enough? What if the screen is brought even closer to the eye to fill more of the field of view? Based on the photoreceptor density plot in Figure 5.24 and the limits of eye rotations shown in Figure 5.21, the maximum FOV seems to be around 270°, which is larger than what could be provided by a flat screen (less than 180°). Increasing the FOV by bringing the screen closer
5.4. IMPLICATIONS FOR VR

would require even higher pixel density, but lens aberrations (Section 4.3) at the periphery may limit the effective field of view. Furthermore, if the lens is too thick and too close to the eye, then the eyelashes may scrape it; Fresnel lenses may provide a thin alternative, but present additional artifacts. Thus, the quest for a VR retina display may end with a balance between optical system quality and limitations of the human eye. Curved screens may help alleviate some of the problems.

Foveated rendering One of the frustrations with this analysis is that we have not been able to exploit that fact that photoreceptor density decreases away from the fovea. We had to keep the pixel density high everywhere because we have no control over which part of the display the user will be look at. If we could track where the eye is looking and have a tiny, movable display that is always positioned in front of the pupil, with zero delay, then much fewer pixels would be needed. This would greatly decrease computational burdens on graphical rendering systems (covered in Chapter 4). Instead of moving a tiny screen, the process can be simulated by keeping the fixed display but focusing the graphical rendering only in the spot where the eye is looking. This is called foveated rendering, which has been shown to work [23], but is currently too costly and there is too much delay and other discrepancies between the eye movements and the display updates. In the near future, it may become an effective approach for the mass market.

VOR gain adaptation The VOR gain is a ratio that compares the eye rotation rate (numerator) to counter the rotation and translation rate of the head (denominator). Because head motion has six DOFs, it is appropriate to break the gain into six components. For example, if you yaw your head to the left at $10^\circ$ per second, then the eye yaws at $10^\circ$ per second in the opposite direction. The VOR roll gain is very small because the eyes have a tiny roll range. The VOR translational gain depends on the distance to the features.

Recall from Section 2.2.3 that adaptation is a universal feature of our sensory systems. VOR gain is no exception. For those who wear eyeglasses, the VOR gain must adapt due to the optical transformations described in Section 1.2. Lenses affect the field of view and perceived size and distance of objects. The VOR comfortably adapts to this problem by changing the gain. Now suppose that you are wearing a VR headset that may suffer from flaws such as an imperfect optical system, tracking latency, and incorrectly rendered objects on the screen. In this case, adaptation may occur as the brain attempts to adapt its perception of stationarity to compensate for the flaws. In this case, your visual system could convince your brain that the headset is functioning correctly, and then your perception of stationarity in the real world would become distorted until you readapt. For example, after a flawed VR experience, you might yaw your head in the real world and have the sensation that truly stationary objects are sliding back and forth.

![Figure 5.26: Most displays still work in the same way as old TV sets and CRT monitors: By updating pixels line-by-line. For a display that has 60 FPS (frames per second), this could take up to 16.67ms.](image)

Display scanout Recall from Section 4.3 that cameras have either a rolling or global shutter based on whether the sensing elements are scanned line-by-line or in parallel. Displays work the same way, but whereas cameras are an input device, displays are the output analog. The vast majority of displays today have a rolling scanout (called raster scan), rather than global scanout. This implies that the pixels are updated line by line, as shown in Figure 5.26. This procedure is an artifact of old TV sets and monitors, which each had a cathode ray tube (CRT) with phosphor elements on the screen. An electron beam was bent by electromagnets so that it would repeatedly strike and refresh the glowing phosphors.

Due to the slow charge and response time of photoreceptors, we do not perceive the scanout pattern during normal use. However, when our eyes, features in the scene, or both are moving, then side effects of the rolling scanout may become perceptible. Think about the operation of a line-by-line printer, as in the case of a receipt printer on a cash register. If we pull the tape while it is printing, then the lines would become stretched apart. If it is unable to print a single line at once, then the lines themselves would become slanted. If we could pull the tape to the side while it is printing, then the entire page would become slanted. You can also achieve this effect by repeatedly drawing a horizontal line with a pencil while using the other hand to gently pull the paper in a particular direction. The paper in this analogy is the retina and the pencil corresponds to light rays attempting to charge photoreceptors. Figure 5.27 shows how a rectangle would distort under cases of smooth pursuit and VOR. Current displays have an option called vsync, which synchronizes output to the display so that the displayed image corresponds to a single, undistorted frame. By turning off this option, more frames per second could be rendered to the display then it can handle. The display buffer simply gets updated in the middle of the scan. This reduces the side effects due to scanout, but introduces another artifact called tearing (as in tearing a sheet of paper). Further improvements could be made by beam racing and just-in-time pixels, which means rendering each line by taking into account the precise time at which it will be drawn [11, 5, 34]. Yet another problem with displays is that the pixels could take so long

---

\[^1^\text{This frequently happened to the author while developing and testing the Oculus Rift.}\]
5.4. IMPLICATIONS FOR VR

141

Figure 5.27: Artifacts due to display scanout: (a) A vertical rectangle in the scene. (b) How it may distort during smooth pursuit while the rectangle moves to the right in the virtual world. (c) How a stationary rectangle may distort when rotating the head to the right while using the VOR to compensate. The cases of (b) are (c) are swapped if the direction of motion is reversed in each case.

to switch (up to 20ms) that sharp edges appear to be blurred. We will continue discussing these problems in Section 6.2 in the context of motion perception.

Retinal image slip  Recall that eye movements contribute both to maintaining a target in a fixed location on the retina (smooth pursuit, VOR) and also changing its location slightly to reduce perceptual fading (microsaccades). During ordinary activities (not VR), the eyes move and the image of a feature may move slightly on the retina due to motions and optical distortions. This is called retinal image slip. Once a VR headset is used, the motions of image features on the retina might not match what would happen in the real world. This is due to many factors already mentioned, such as optical distortions, tracking latency, and display scanout. Thus, the retinal image slip due to VR artifacts does not match the retinal image slip encountered in the real world? The consequences of this barely been identified, much less characterized scientifically. They are likely to contribute to fatigue, and possibly VR sickness. As an example of the problem, there is evidence that microsaccades are triggered by the lack of retinal image slip [17]. This implies that differences in retinal image slip due to VR usage could interfere with microsaccade motions, which are already not fully understood.

Vergence-accommodation mismatch  Recall from Section 4.4 that accommodation is the process of changing the eye lens’ optical power so that close objects can be brought into focus. This normally occurs with both eyes fixated on the same object, resulting in a stereoscopic view that is brought into focus. In the real world, the vergence motion of the eyes and the accommodation of the lens are tightly coupled. For example, if you place your finger 10cm in front of your face, then your eyes will try to increase the lens power while the eyes are strongly converging. If a lens is placed at a distance of its focal length from a screen, then with normal eyes it will always be in focus while the eye is relaxed (recall Figure 4.30). What if an object is rendered to the screen so that it appears to be only 10cm away? In this case, the eyes strongly converge, but they do not need to change the optical power of the eye lens. The eyes may nevertheless try to accommodate, which would have the effect of blurring the perceived image. The result is called vergence-accommodation mismatch because the stimulus provided by VR is inconsistent with the real world. Even if the eyes become accustomed to the mismatch, the user may feel extra strain or fatigue after prolonged use. The eyes are essentially being trained to allow a new degree of freedom: Separating vergence from accommodation, rather than coupling them. Engineering solutions may provide some relief from this problem, but they are currently too costly and imprecise. For example, the mismatch can be greatly reduced by employing eye tracking to estimate the amount of vergence and then altering the power of the optical system [2, 12].

Further Reading

For further reading on the photoreceptor mosaic, see Chapter 3 of [72].
Photoreceptor density variation over humans: [13].
Retina display analysis (non-VR): [28].
More about eyes and lenses together: [67].
All about eye movement from a neuroscience perspective: [10].
VOR gain adaptation: [10, 20, 64].
Survey of microsaccades: [59].
Smooth pursuit and saccade coordination: [15].
Monkey paper for head/eye coordination: [54].
See Oxford Handbook of Eye Movements: [41].
For more neuroscience, see Chapter 7 of [46].
Nice vision summary:
You can also figure out whether it is worthwhile to upgrade your TV by using the retina display analysis:
http://www.rtings.com/tv/learn/size-to-distance-relationship
Abrash blog post about scanout: [1].
Comfort of vergence-accommodation mismatch: [65].
Chapter 6
Visual Perception

This chapter continues where Chapter 5 left off by transitioning from the physiology of human vision to perception. If we were computers, then this transition might seem like going from low-level hardware to higher-level software and algorithms. How do our brains interpret the world around us so effectively in spite of our limited biological hardware? To understand how we may be fooled by visual stimuli presented by a display, you must first understand how our we perceive or interpret the real world under normal circumstances. It is not always clear what we will perceive. We have already seen several optical illusions. VR itself can be considered as a grand optical illusion. Under what conditions will it succeed or fail?

Section 6.1 covers perception of the distance of objects from our eyes, which is also related to the perception of object scale. Section 6.2 explains how we perceive motion. An important part of this is the illusion of motion that we perceive from videos, which are merely a sequence of pictures. Section 6.3 covers the perception of color, which may help explain why displays use only three colors (red, green, and blue) to simulate the entire spectral power distribution of light (recall from Section 4.1). Finally, Section 6.4 presents a statistically based model of how information is combined from multiple sources to produce a perceptual experience.

6.1 Perception of Depth

This section explains how humans judge the distance from their eyes to objects in the real world using vision. The perceived distance could be metric, which means that an estimate of the absolute distance is obtained. For example, a house may appear to be about 100 meters away. Alternatively, the distance information could be ordinal, which means that the relative arrangement of visible objects can be inferred. For example, one house appears to be closer than another one because it is partially blocking the view of the further one.

Monocular vs. stereo cues A piece of information that is derived from sensory stimulation and is relevant for perception is called a sensory cue or simply a cue. In this section, we consider only depth cues, which contribute toward depth perception. If a depth cue is derived from the photoreceptors or movements of a single eye, then it is called a monocular depth cue. If both eyes are required, then it is a stereo depth cue. There are many more monocular depth cues than stereo, which explains why we are able to infer so much depth information from a single photograph. Figure 6.1 shows an example. The illusions in Figure 6.2 show that even simple line drawings are enough to provide strong cues. Interestingly, the cues used by humans also work in computer vision algorithms to extract depth information from images [71].
6.1. PERCEPTION OF DEPTH

Figure 6.2: Even simple line drawings provide significant depth cues. (a) The Ponzo illusion: The upper yellow bar appears to be longer, but both are the same length. (b) The Müller-Lyer illusion: The lower horizontal segment appears to be shorter than the one above, but they are the same length.

6.1.1 Monocular depth cues

Retinal image size Many cues result from the geometric distortions caused by perspective projection; recall the “3D” appearance of Figure 1.19(c). For a familiar object, such as a human, coin, or basketball, we often judge its distance by how “large” it appears to be. Recalling the perspective projection math from Section 3.4, the size of the image on the retina is proportional to $1/z$, in which $z$ is the distance from the eye (or the common convergence point for all projection lines). See Figure 6.3. The same thing happens when taking a picture with a camera: A picture of a basketball would occupy a larger part of the image, covering more pixels, as it becomes closer to the camera. Two important factors exist. First, the viewer must be familiar with the object to the point of comfortably knowing its true size. For familiar objects, such as people or cars, our brains perform size constancy scaling by assuming that the distance, rather than the size, of the person is changing if they come closer. Size constancy falls under the general heading of subjective constancy, which appears through many aspects of perception, including shape, size, and color. The second factor is that the object must appear naturally so that it does not conflict with other depth cues.

If there is significant uncertainty about the size of an object, then knowledge of its distance should contribute to estimating its size. This falls under size perception, which is closely coupled to depth perception. Cues for each influence the other, in a way discussed in Section 6.4.

One controversial theory is that our perceived visual angle differs from the actual visual angle. The visual angle is proportional to the retinal image size. This theory is used to explain the illusion that the moon appears to be larger when it is near the horizon. For another example, see Figure 6.4.

Figure 6.3: The retinal image size of a familiar object is a strong monocular depth cue. The closer object projects onto a larger number of photoreceptors, which cover a larger portion of the retina.

Figure 6.4: For the Ebbinghaus illusion, the inner disc appears larger when surrounded by smaller discs. The inner disc is the same size in either case. This may be evidence of discrepancy between the true visual angle (or retinal image size) and the perceived visual angle.
6.1. PERCEPTION OF DEPTH

(a) (b)

Figure 6.5: Height in visual field. (a) Trees closer to the horizon appear to be further away, even though all yield the same retinal image size. (b) Incorrect placement of people in the visual field illustrates size constancy scaling, which is closely coupled with depth cues.

Height in the visual field Figure 6.5(a) illustrates another important cue, which is the height of the object in the visual field. The Ponzo illusion in Figure 6.2(a) exploits this cue. Suppose that we can see over a long distance without obstructions. Due to perspective projection, the horizon is a line that divides the view in half. The upper half is perceived as the sky, and the lower half is the ground. The distance of objects from the horizon line corresponds directly to their distance due to perspective projection: The closer to the horizon, the further the perceived distance. Size constancy scaling, if available, combines with the height in the visual field, as shown in Figure 6.5(b).

Accommodation Recall from Section 4.4 that the human eye lens can change its optical power through the process of accommodation. For young adults, the amount of change is around 10D (diopters), but it decreases to less than 1D for adults over 50 years old. The ciliary muscles control the lens and their tension level is reported to the brain through efference copies of the motor control signal. This is the first depth cue that does not depend on signals generated by the photoreceptors.

Motion parallax Up until now, the depth cues have not exploited motions. If you have ever looked out the side window of a fast-moving vehicle, you might have noticed that the nearby objects race by much faster than further objects. The relative difference in speeds is called parallax and is an important depth cue; see Figure 6.6. Even two images, from varying viewpoints within a short amount of time, provide strong depth information. Imagine trying to simulate a stereo rig of cameras by snapping one photo and quickly moving the camera sideways to snap another. If the rest of the world is stationary, then the result is roughly equivalent to having two side-by-side cameras. Pigeons frequently bob their heads back and forth to obtain stronger depth information than is provided by their pair of eyes. Finally, closely related to motion parallax is optical flow, which is a characterization of the rates at which features move across the retina. This will be revisited in Sections 6.2 and 8.4.

Other monocular cues Figure 6.7 shows several other monocular cues. As shown in Figure 6.7(a), shadows that are cast by a light source encountering an object provide an important cue. Figure 6.7(b) shows a simple drawing that provides an ordinal depth cue called interposition by indicating which objects are in front of others. Figure 6.7(c) illustrates the image blur cue, where levels are depth are inferred from the varying sharpness of focus. Figure 6.7(d) shows an atmospheric cue in which air humidity causes far away scenery to have lower contrast, thereby appearing to be further away.

6.1.2 Stereo depth cues

As you may expect, focusing both eyes on the same object enhances depth perception. Humans perceive a single focused image over a surface in space called the horopter; see Figure 6.8. Recall the vergence motions from Section 5.3. Similar to the accommodation cue case, motor control of the eye muscles for vergence mo-
Figure 6.7: Several more monocular depth cues: (a) *Shadows* resolve ambiguous depth in the *ball and shadow illusion*. (b) The *interposition* of objects provides an ordinal depth cue. (c) Due to *image blur*, one gnome appears to be much closer than the others. (d) This scene provides an *atmospheric cue*: Some scenery is perceived to be further away because it has lower contrast.

---

Figure 6.8: The *horopter* is the loci of points over which the eyes can converge and focus on a single depth. The T curve shows the theoretical horopter based on simple geometry. The E curve shows the empirical horopter, which is much larger and correspond to the region over which a single focused image is perceived. (Figure by Rainer Zenz.)
6.1 PERCEPTION OF DEPTH

tions provides information to the brain about the amount of convergence, thereby providing a direct estimate of distance. Each eye provides a different viewpoint, which results in different images on the retina. This phenomenon is called binocular disparity. Recall from (3.49) in Section 3.5 that the viewpoint is shifted to the right or left to provide a lateral offset for each of the eyes. The transform essentially shifts the virtual world to either side. The same shift would happen for a stereo rig of side-by-side cameras in the real world. However, the binocular disparity for humans is different because the eyes can rotate to converge, in addition to having a lateral offset. Thus, when fixating on an object, the retinal images between the left and right eyes may vary only slightly, but this nevertheless provides a powerful cue used by the brain.

Furthermore, when converging on an object at one depth, we perceive double images of objects at other depths (although we usually pay no attention to it). This double-image effect is called diplopia. You can perceive it by placing your finger about 20cm in front of your face and converging on it. While fixating on your finger, you should perceive double images on other objects around the periphery. You can also stare into the distance while keeping your finger in the same place. You will then see a double image of your finger. If you additionally roll your head back and forth, you should appear as if the left and right versions of your finger are moving up and down with respect to each other. These correspond to dramatic differences in the retinal image, but we are usually not aware of them because we perceive a both retinal images as a single image.

6.1.3 Implications for VR

Incorrect scale perception A virtual world may be filled with objects that are not familiar to us in the real world. In many cases, they might resemble familiar objects, but their precise scale might be difficult to determine. Consider the Tuscany demo world from Oculus VR, shown in Figure 6.9. The virtual villa is designed to be inhabited with humans, but it is difficult to judge the relative sizes and distances of objects because there are not enough familiar objects. Further complicating the problem is that the user’s height in VR might not match his height in the virtual world. Is the user too short, or is the world too big? A common and confusing occurrence is that the user might be sitting down in the real world, but standing in the virtual world. An additional complication occurs if the interpupillary distance (recall from Section 4.4) is not matched with the real world. For example, if the user’s pupils are 64mm apart in the real world but only 50mm apart in the virtual world, then the virtual world will seem much larger, which dramatically affects depth perception. Likewise, if the pupils are very far apart, the user could either feel enormous or the virtual world might seem small. Imagine simulating a Godzilla experience, where the user is 200 meters tall and the entire city appears to be a model. It is fine to experiment with such scale and depth distortions in VR, but it is important to understand their implications on the user’s perception.

Mismatches In the real world, all of the depth cues work together in harmony. We are sometimes fooled by optical illusions that are designed to intentionally cause inconsistencies among cues. Sometimes a simple drawing is sufficient. Figure 6.10 shows an elaborate illusion that requires building a distorted room in the real world. It is perfectly designed so that when viewed under perspective projection from one location, it appears to be a rectangular box. Once our brains accept this, we unexpectedly perceive the size of people changing as they walk across the room! This is because all of the cues based on perspective appear to be functioning correctly. Section 6.4 may help you to understand how multiple cues are resolved, even in the case of inconsistencies.

In a VR system, it is easy to cause mismatches and in many cases they are unavoidable. Recall from Section 5.4 that vergence-accommodation mismatch occurs in VR headsets. Another source of mismatch may occur from imperfect head tracking. If there is significant latency, then the visual stimuli will not appear in the expected place at the expected time. Furthermore, many tracking systems track the head orientation only. This makes it impossible to use motion parallax as a depth cue if the user moves from side to side without any rotation. To preserve most depth cues based on motion, it is important to track head position, in addition to orientation; see Section 9.2. Optical distortions may cause even more mismatch.

Monocular cues are powerful! A common misunderstanding among the general public is that depth perception enabled by stereo cues alone. We are bom-
6.1. PERCEPTION OF DEPTH

(b) The room is designed so that it only appears to be rectangular after perspective projection is applied. One person is actually much further away than the other. (Figure by Alex Valavanis.)

6.2 Perception of Motion

We rely on our vision to perceive motion for many crucial activities. One use to separate a moving figure from a stationary background. For example, a camouflage animal in the forest might only become noticeable when moving. This is clearly useful whether humans are the hunter or the hunted. Motion also helps people to assess the 3D structure of an object. Imagine assessing the value of a piece of fruit in the market by rotating it around. Another use is to visually guide actions, such as walking down the street or hammering a nail. VR systems have the tall order of replicating these uses in a virtual world in spite of limited technology. Just as important as the perception of motion is the perception of non-motion, which we called perception of stationarity in Section 2.3. For exam-
Motion detection neurons
Feature detection neurons
Intermediate neurons
Image on photoreceptors

Figure 6.12: The neural circuitry directly supports motion detection. As the image feature moves across the retina, nearby feature detection neurons (labeled a and b) activate in succession. Their outputs connect to motion detection neurons (labeled c). Due to different path lengths from a and b to c, the activation signal arrives at different times. Thus, c activates when the feature was detected by a slightly before being detected by b.

6.2.1 Detection mechanisms

Reichardt detector Figure 6.12 shows a neural circuitry model, called a Reichardt detector, which responds to directional motion in the human visual system. Neurons in the ganglion layer and LGN detect simple features in different spots in the retinal image. At higher levels, motion detection neurons exist that respond when the feature moves from one spot on the retina to another nearby. The motion detection neuron activates for a feature speed that depends on the difference in path lengths from its input neurons. It is also sensitive to a particular direction of motion based on the relative locations of the receptive fields of the input neurons. Due to the simplicity of the motion detector, it can be easily fooled. Figure 6.12 shows a feature moving from left to right. Suppose that a train of features moves from right to left. Based on the speed of the features and the spacing between them, the detector may inadvertently fire, causing motion to be perceived in the opposite direction. This is the basis of the wagon-wheel effect, for which a wheel with spokes or a propeller may appear to be rotating in the opposite direction, depending on the speed. The process can be further disrupted by causing eye vibrations from humming. This simulates stroboscopic conditions, which discussed in Section 6.2.2. Another point is that the motion detectors are subject

Figure 6.13: Due to local nature of motion detectors, the aperture problem results. The motion of the larger body is ambiguous when perceived through a small hole because a wide range of possible body motions could produce the same effect inside of the hole. An incorrect motion inference usually results. To adaptation. Therefore, several illusions exist, such as the waterfall illusion and the spiral aftereffect, in which incorrect motions are perceived due to aftereffects.

From local data to global conclusions Motion detectors are local in the sense that a tiny portion of the visual field causes each to activate. In most cases, data from detectors across large patches of the visual field are integrated to indicate coherent motions of rigid bodies. (An exception would be staring a pure analog TV static.) All pieces of a rigid body move through space according to the equations from Section 5.2. This coordinated motion is anticipated by our visual experience to match common expectations. If too much of the moving body is blocked, then the aperture problem results, which is shown in Figure 6.13. A clean mathematical way to describe the global motions across the retina is by a vector field, which assigns a velocity vector at every position. The global result is called the optical flow, which provides powerful cues for both object motion and self-motion. The latter case results in section, which is a leading cause of VR sickness; see Sections 8.1 and 10.1 for details.

Distinguishing object motion from observer motion Figure 6.14 shows two cases that produce the same images across the retina over time. In Figure 6.14(a), the eye is fixed while the object moves. In Figure 6.14(b), the situation is reversed: The object is fixed, but the eye moves. The brain uses several cues to differentiate between these cases. Saccadic suppression, which was mentioned in Section 5.5, suppresses vision during movements, which may suppress motion detectors in the second case. Another cue is provided by proprioception, which is the body’s ability to estimate its own motions due to motor commands. This
6.2. PERCEPTION OF MOTION

Two motions that cause equivalent movement of the image on the retina: (a) The eye is fixed and the object moves; (b) the eye moves while the object is fixed. Both of these are hard to achieve in practice due to eye rotations (smooth pursuit and VOR).

includes the use of eye muscles in the second case. Finally, information is provided by large-scale motion. If it appears that the entire scene is moving, then the brain assumes the most likely interpretation, which is that the user must be moving. This is why the haunted swing illusion, shown in Figure 2.20, is so effective.

6.2.2 Stroboscopic apparent motion

Nearly everyone on Earth has seen a motion picture, whether through a TV, smartphone, or movie screen. The motions we see are an illusion because a sequence of still pictures is being flashed onto the screen. This phenomenon is called stroboscopic apparent motion; it was discovered and refined across the 19th century. The zoetrope, shown in Figure 6.15 was developed around 1834. It consists of a rotating drum with slits that allow each frame to be visible for an instant while the drum rotates. In Section 1.3, Figure 1.20 showed the Horse in Motion film from 1878.

Why does this illusion of motion work? An early theory, which has largely been refuted in recent years, is called persistence of vision. The theory states that images persist in the vision system during the intervals in between frames, thereby causing them to be perceived as continuous. One piece of evidence against this theory is that images persist for up to 100ms [ ], which implies that the 10 FPS (Frames Per Second) is the slowest speed that stroboscopic apparent motion would
**6.2. PERCEPTION OF MOTION**

<table>
<thead>
<tr>
<th>FPS Occurrence</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 Stroboscopic apparent motion starts</td>
</tr>
<tr>
<td>10 Ability to distinguish individual frames is lost</td>
</tr>
<tr>
<td>16 Old home movies; early silent films</td>
</tr>
<tr>
<td>24 Hollywood classic standard</td>
</tr>
<tr>
<td>25 PAL television before interlacing</td>
</tr>
<tr>
<td>30 NTSC television before interlacing</td>
</tr>
<tr>
<td>48 Two-blade shutter; proposed new Hollywood standard</td>
</tr>
<tr>
<td>50 Interlaced PAL television</td>
</tr>
<tr>
<td>60 Interlaced NTSC television; perceived flicker in some displays</td>
</tr>
<tr>
<td>72 Three-blade shutter; minimum CRT refresh rate for comfort</td>
</tr>
<tr>
<td>90 Modern VR headsets; no more discomfort from flicker</td>
</tr>
<tr>
<td>1000 Ability to see zipper effect for fast, blinking LED</td>
</tr>
<tr>
<td>5000 Cannot perceive zipper effect</td>
</tr>
</tbody>
</table>

Figure 6.17: Various frame rates and comments on the corresponding stroboscopic apparent motion. Units are in Frames Per Second (FPS).

Work; however, it is also perceived down to 2 FPS [9]. Another piece of evidence for the persistence of vision is the existence of stroboscopic apparent motions that cannot be accounted for by it. The phi phenomenon and beta movement are examples of motion perceived in a sequence of blinking lights, rather than flashing frames (see Figure 6.14). The most likely reason that stroboscopic apparent motion works is that it triggers the neural motion detection circuitry illustrated in Figure 6.14 [10, 15].

**Frame rates** How many frames per second are appropriate for a motion picture? The answer depends on the intended use. Figure 6.17 shows a table of significant frame rates from 2 to 5000. Stroboscopic apparent motion begins at 2 FPS. Imagine watching a security video at this rate. It is easy to distinguish individual frames, but the motion of a person would also be perceived. Once 10 FPS is reached, the motion is obviously more smooth and we start to lose the ability to distinguish individual frames. Early silent films ranged from 16 to 24 FPS. The frame rates were often fluctuating and at a faster speed than they were filmed. Once sound was added to film, incorrect speeds and fluctuations in the speed were no longer tolerated because both sound and video needed to be synchronized. This motivated a fixed rate of 24 FPS that is still used today by the movie industry. Personal video cameras remained at 16 or 18 FPS into the 1970s. The famous Zapruder film of the Kennedy assassination in 1963 was taken at 18.3 FPS. Although 24 FPS may be enough to perceive motions smoothly, a large part of cinematography is devoted to ensuring that motions are not so fast that jumps are visible due to the slow frame rate.

Such low frame rates unfortunately lead to perceivable flicker as the images rapidly flash on the screen with black in between. This motivated several workarounds.

In the case of movie projectors, two-blade and three-blade shutters were invented so that they would show each frame two or three times, respectively. This enabled movies to be shown at 48 FPS and 72 FPS, thereby reducing discomfort from flickering. Analog television broadcasts in the 20th century were at 25 (PAL standard) or 30 FPS (NTSC standard), depending on the country. To double the frame rate and reduce perceived flicker, they used interlacing to draw half the image in one frame time, and then half in the other. Every other horizontal line is drawn in the first half, and the remaining lines are drawn in the second. This increased the frames rates on television screens to 50 and 60 FPS. The game industry has used 60 FPS standard target for smooth game play.

As people started sitting close to giant CRT monitors in the early 1990s, the flicker problem became problematic again. Our perception of flicker is stronger at the periphery, particularly at about 30° from center [4]. Furthermore, even when flicker cannot be directly perceived, it may still contribute to fatigue or headaches. Therefore, frame rates were increased to even higher levels. A minimum acceptable ergonomic standard for large CRT monitors was 72 FPS, with 85 to 90 FPS being widely considered as sufficiently high to eliminate flicker problems. The problem has been carefully studied by psychologists under the heading of flicker fusion threshold; the precise rates at which flicker is perceptible or causes fatigue depends on many factors in addition to FPS, such as position on retina, age, color, and light intensity. Thus, the actual limit depends on the kind of display size, specifications, how it is used, and who is using it. Modern LCD and LED displays, used as televisions, computer screens, and smartphone screens, have 60, 120, and even 240 FPS.

The story does not end there. If you connect an LED to a pulse generator (put a resistor in series), then flicker can be perceived at much higher rates. Go to a dark room and hold the LED in your hand. If you wave it around so fast that your eyes cannot track it, then the flicker becomes perceptible as a zipper pattern. This happens because each time the LED pulses on, it is imaged in a different place on the retina. Without image stabilization, it appears as an array of lights. The faster the motion, the further apart the images will appear. The higher the pulse rate (or FPS), the closer together the images will appear. Therefore, to see the zipper effect at very high speeds, you need to move the LED very quickly. It is possible to see the effect for a few thousand FPS.

### 6.2.3 Implications for VR

Unfortunately, VR systems require much higher display performance than usual. We have already seen in Section 5.3 that much higher resolution is needed so that pixels and aliasing artifacts are not visible. The next problem is that higher frame rates are needed in comparison to ordinary television or movie standards of 24 FPS or even 60 FPS. To understand why, see Figure 5.18. The problem is easiest to understand for the perception of stationarity, which was mentioned in Section 5.4. Fixate on a nearby object and yaw your head to the left. Your eyes should then
6.2. PERCEPTION OF MOTION

Figure 6.18: A problem with perception of stationarity under stroboscopic apparent motion: The image of a feature slips across the retina in a repeating pattern as the VOR is performed.

rotate to the right to maintain the object in a fixed location on the retina, due to the VOR (Section 5.3). If you do the same while wearing an HMD and fixating on an object in the virtual world, then the image of the object needs to shift across the screen while you turn your head. Assuming that the pixels instantaneously change at each new frame time, the image of the virtual object will slip across the retina as shown in Figure 6.18. The result is a kind of pudder in which the object appears to be wobbling from side to side with high frequency but small amplitude.

The problem is that the image is fixed on the screen for too long while it is supposed to be moving continuously across the screen. At 60 FPS, it is fixed for 16.67ms during each frame (in an idealized setting). If the screen is instead turned on for only one or two milliseconds for each frame, and then made black during the remaining times, then the amount of retinal image slip is greatly reduced. This display mode is called low persistence, and is shown in Figure 6.18 (a). The short amount of time that the display is illuminated is sufficient for the photoreceptors to collect enough photons to cause the image to be perceived. The problem is that at 60 FPS in low-persistence mode, flicker is perceived, which can lead to fatigue or headaches. This can be easily perceived at the periphery in a bright scene in the Samsung Gear VR headset. If the frame rate is increased to 90 FPS or above, then the adverse side effects of flicker subside for nearly everyone. If the frame rate is increased to 500 FPS or beyond, then it would not even need to flicker, as depicted in Figure 6.18 (b).

One final point is that fast pixel switching speed is implied in the Figure 6.18. In a modern OLED display panel, the pixels can reach their target intensity values in less than 0.1ms. However, many LCD displays change pixel values much more slowly. The delay to reach the target intensity may be as long as 20ms, depending on the amount and direction of intensity change. In this case, a fixed virtual object appears to smear or blur in the direction of motion. This was easily observable in the Oculus Rift DK1, which used an LCD display panel.

6.3 Perception of Color

Figure 6.19: An engineering solution to reduce retinal image slip: (a) Using low persistence, the display is lit for a short enough time to trigger photoreceptors (t₁ − t₀) and then blanked for the remaining time (t₂ − t₁). Typically, t₁ − t₀ is around one to two milliseconds. (b) If the frame rate were extremely fast (at least 500 FPS), then the blank interval would not be needed.

What makes an object “purple”, “pink”, or “gray”? Color perception is unusual because it is purely the result of our visual physiology and neural structures, rather than something that can be measured in the physical world. In other words, “It’s all in your head.” If two people have comparable color perception systems, then they can discuss colors using commonly agreed upon names as they perceive an object as having the same color. This contrasts other perception topics such as motion, depth, and scale, all of which correspond to measurable activity in the surrounding world. The size of an object or the speed of its motion relative to some frame could be determined by instrumentation. Humans would be forced to agree on the numerical outcomes regardless of how their individual perceptual systems are functioning.

The dress Figure 6.20 illustrates this point with the dress color illusion. It was born by Cecilia Blesadale and became an internet meme when millions of people quickly began to argue about the color of the dress. Based on the precise combination of colors and lighting conditions, its appearance fell on the boundary of what human color perceptual systems can handle. About 57% perceive it as blue and black (correct), 30% percent perceive it as white and gold, 10% perceive blue and brown, and 10% could switch between perceiving any of the color combinations [34].

Dimensionality reduction Recall from Section 4.4 that light energy is a jumble of wavelengths and magnitudes that form the spectral power distribution. Figure 4.4 provided an illustration. As we see objects, the light in the environment is
6.3. PERCEPTION OF COLOR

Figure 6.20: In 2014, this dress photo became an Internet sensation as people were unable to agree upon whether it was “blue and black” or “white and gold”, which are strikingly different perceptions of color.

reflected off of surfaces in a wavelength-dependent way according to the spectral distribution function (Figure 4.7). As the light passes through our eyes and is focused onto the retina, each photoreceptor receives a jumble of light energy that contains many wavelengths. Since the power distribution is a function of wavelength, the set of all possible distributions is a function space, which is generally infinite-dimensional. Our limited hardware cannot possibly sense the entire function. Instead, the rod and cone photoreceptors sample it with a bias toward certain target wavelengths, as was shown in Figure 5.3 of Section 5.1. The result is a well-studied principle in engineering called dimensionality reduction. Here, the infinite-dimensional space of power distributions collapses down to a 3D color space. It is no coincidence that we have precisely three types of cones, and that our RGB displays target the same colors as the photoreceptors.

Yellow = Green + Red To help understand this reduction, consider the perception of “yellow”. According to the visible light spectrum (Figure 4.5), a wavelength of about 580nm. Suppose we had a pure light source that shines light of exactly 580nm wavelength onto our retinas with no other wavelengths. The spectral distribution function would have a spike at 580nm and be zero everywhere else. If we had a cone with peak detection at 580nm and no sensitivity to other wavelengths, then it would perfectly detect yellow. Instead, we perceive yellow by activation of both green and red cones because their sensitivity regions (Figure 5.3) include 580nm. It should then be possible to generate the same photoreceptor response by sending a jumble of light that contains precisely two wavelengths: 1) Some “green” at 533nm, and 2) some “red” at 564nm. If the magnitudes of green and red are tuned so that the green and red cones activate in the same way as they did for pure yellow, then it becomes impossible for our visual system to distinguish the green/red mixture from pure yellow. Both are perceived as “yellow”. This matching of colors from red, green and blue components is called metamerism. Such a blending is precisely what is done on a RGB display to produce yellow. Suppose the intensity of each color ranges from 0 (dark) to 255 (bright). Red is produced by RGB = (255, 0, 0), and green is RGB = (0, 255, 0). These each activate one LED (or LCD) color, thereby producing a pure red or green. If both are turned on, then yellow is perceived. Thus, yellow is RGB = (255, 255, 0).

Color spaces For convenience, a parameterized color space is often defined. One of the most common in computer graphics is called HSV, which has the following three components (Figure 6.21).

- The hue, which corresponds directly to the perceived color, such as “red” or “green”.
- The saturation, which is the purity of the color. In other words, how much energy is coming from wavelengths other than the wavelength of the hue?
- The value, which corresponds to the brightness.

There are many methods to scale the HSV coordinates, which distort the color space in various ways. The RGB values could alternatively be used, but are sometimes more difficult for people to interpret.
6.3. PERCEPTION OF COLOR

Figure 6.22: 1931 CIE color standard with RGB triangle. This representation is correct in terms of distances between perceived colors. (Figure by Jeff Yurek.)

It would be ideal to have a representation in which the distance between two points corresponds to the amount of perceptual difference. In other words, as two points are further apart, our ability to distinguish them is increased. The distance should correspond directly to the amount of distinguishability. Vision scientists designed a representation to achieve this, resulting in the 1931 CIE color standard shown in Figure 6.22. Thus, the CIE is considered to be undistorted from a perceptual perspective. It is only two-dimensional because it disregards the brightness component, which is independent of color perception according to color matching experiments \[46\].

Mixing colors  Suppose that we have three pure source of light, as in that produced by an LED, in red, blue, and green colors. We have already discussed how to produce yellow by blending red and green. In general, most perceptible colors can be matched by a mixture of three. This is called trichromatic theory (or Young-Helmholtz theory). A set of colors that achieves this is called primary colors. Mixing all three evenly produces perceived white light, which on a display is achieved as RGB= (255, 255, 255). Black is the opposite: RGB= (0, 0, 0). Such light mixtures follow a linearity property. Suppose primary colors are used to perceptually match power distributions of two different light sources. If the light sources are combined, then their intensities of the primary colors need only to be added to obtain the perceptual match for the combination. Furthermore, the overall intensity can be scaled by multiplying the red, green, and blue components without affecting the perceived color. Only the perceived brightness may be changed.

The discussion so far has focused on additive mixtures. When mixing paints or printing books, colors mix subtractively because the spectral reflectance function is being altered. When starting with a white canvas or sheet of paper, virtually all wavelengths are reflected. Painting a green line on the page prevents all wavelengths other than green from being reflected at that spot. Removing all wavelengths results in black. Rather than using RGB components, printing presses are based on CMYK, which correspond to cyan, magenta, yellow, and black. The first three are pairwise mixes of the primary colors. A black component is included to reduce the amount of ink wasted by using the other three colors to subtractively produce black. Note that the targeted colors are observed only if the incoming light contains the targeted wavelengths. The green line would appear green under pure, matching green light, but might appear black under pure blue light.

Constancy  The dress in Figure 6.20 showed an extreme case that results in color confusion across people due to the strange lighting conditions. Ordinarily, human color perception is surprisingly robust to the source of color. A red shirt appears to be red whether illuminated under indoor lights at night or in direct sunlight. These correspond to vastly different cases in terms of the spectral power distribution that reaches the retina. Our ability to perceive an object as having the same color over a wide variety of lighting conditions is called color constancy.
6.3. PERCEPTION OF COLOR

Figure 6.23: (a) The perceived hot air balloon colors are perceived the same regardless of the portions that are in direct sunlight or in a shadow. (Figure by Wikipedia user Shanta.) (b) The checker shadow illusion from Section 2.3 is explained by the lightness constancy principle as the shadows prompt compensation of the perceived lightness. (Figure by Adrian Pingstone.)

Several perceptual mechanisms allow this to happen. One of them is chromatic adaptation, which results in a shift in perceived colors due to prolonged exposure to specific colors. Another factor in the perceived color is the expectation from the colors of surrounding objects. Furthermore, memory about objects are usually colored in the environment biases our interpretation.

The constancy principle also appears without regard to particular colors. Our perceptual system also maintains lightness constancy so that the overall brightness levels appear to be unchanged, even after lighting conditions are dramatically altered; see Figure 6.23(a). Under the ratio principle theory, only the ratio of reflectances between objects in a scene are perceptually maintained, whereas the overall amount of reflected intensity is not perceived. Further complicating matters, our perception of object lightness and color are maintained as the scene contains uneven illumination. A clear example is provided by shadows cast by one object onto another. Our perceptual system accounts for the shadow and adjusts our perception of the object shade or color. The checker shadow illusion shown in Figure 6.23 is caused by this compensation due to shadows.

Display issues Displays generally use RGB lights to generate the palette of colors and brightness. Recall Figure 5.22, which showed the subpixel mosaic of individual component colors for some common displays. Usually, the intensity of each R, G, and B value is set by selecting an integer from 0 to 255. This is a severe limitation on the number of brightness levels, as stated in Section 5.4. One cannot expect to densely cover all seven orders of magnitude of perceptible light intensity. One way to enhance the amount of contrast over the entire range is to perform gamma correction. In most displays, images are encoded with a gamma of about 0.45 and decoded with a gamma of 2.2.

Another issue is that the set of all available colors lies inside of the triangle formed by R, G, and B vertices. This limitation is shown for the case of the sRGB standard in Figure 6.22. Most the CIE is covered, but many colors that humans are capable of perceiving cannot be generated on the display.

6.4 Combining Sources of Information

Throughout this chapter, we have seen perceptual processes that combine information from multiple sources. These could be cues from the same sense, as in the numerous monocular cues used to judge depth. Perception may also combine information from two or more senses. For example, people typically combine both visual and auditory cues when speaking face to face. Information from both sources makes it easier to understand someone, especially if there is significant background noise. We have also seen that information is integrated over time, as in the case of saccades being employed to fixate on several object features. Finally, our memories and general expectations about the behavior of the surrounding world bias our conclusions. Thus, information is integrated from prior expectations and the reception of many cues, which may come from different senses at different times.

Statistical decision theory provides a useful and straightforward mathematical model for making choices that incorporate prior biases and sources of relevant, observed data. It has been applied in many fields, including economics, psychology, signal processing, and computer science. One key component is Bayes’ rule, which specifies how the prior beliefs should be updated in light of new observations, to
obtain posterior beliefs. More formally, the “beliefs” are referred as probabilities. If the probability takes into account information from previous information, it is called a conditional probability. There is no room to properly introduce probability theory here; only the basic ideas are given to provide some intuition without the rigor. For further study, find an online course or classic textbook (for example, [60]).

Let
\[ H = \{ h_1, h_2, \ldots, h_n \} \]  \hfill (6.1)
be a set of hypotheses (or interpretations). Similarly, let
\[ C = \{ c_1, c_2, \ldots, c_m \} \]  \hfill (6.2)
\( C \) be a set of possible outputs of a cue detector. For example, the cue detector might output the eye color of a face that is currently visible. In this case \( C \) is the set of possible colors:
\[ C = \{ \text{brown, blue, green, hazel} \}. \]  \hfill (6.3)
Modeling a face recognizer, \( H \) would correspond to the set of people familiar to the person.

We want to calculate probability values for each of the hypotheses in \( H \). Each probability value must lie between 0 to 1, and the sum of the probability values for every hypothesis in \( H \) must sum to one. Before any cues, we start with an assignment of values called the prior distribution, which is written as \( P(h) \). The “\( P \)” denotes that it is a probability function or assignment; \( P(h) \) means that an assignment has been applied to every \( h \) in \( H \). The assignment must be made so that
\[ P(h_1) + P(h_2) + \cdots + P(h_n) = 1, \]  \hfill (6.4)
and \( 0 \leq P(h_i) \leq 1 \) for each \( i \) from 1 to \( n \).

The prior probabilities are generally distributed across the hypotheses in a diffuse way; an example is shown in Figure 6.25(a). The likelihood of any hypothesis being true before any cues is proportional to its frequency of occurring naturally, based on evolution and the lifetime of experiences of the person. For example, if you open your eyes at a random time in your life, what is the likelihood of seeing a human being versus a wild boar?

Under normal circumstances (not VR!), we expect that the probability for the correct interpretation will rise as cues arrive. The probability of the correct hypothesis should pull upward toward 1, effectively stealing probability mass from the other hypotheses, which pushes their values toward 0; see Figure 6.25(b). A “strong” cue should lift the correct hypothesis upward more quickly than a “weak” cue. If a single hypothesis has a probability value close to 1, then the distribution is considered peaked, which implies high confidence; see Figure 6.25(c). In the other direction, inconsistent or incorrect cues have the effect of diffusing the probability across two or more hypotheses. Thus, the probability of the correct hypothesis

![Figure 6.25: Example probability distributions](image-url)
may be lowered as other hypotheses are considered plausible and receive higher values. It may also be possible that two alternative hypotheses remain strong due to ambiguity that cannot be solved from the given cues; see Figure 6.24(d).

To take into account information from a cue, a conditional distribution is defined, which is written as $P(h | c)$. This is spoken as “the probability of $h$ given $c$.” This corresponds to a probability assignment for all possible combinations of hypotheses and cues. For example, it would include $P(h_2 | c_3)$, if there are at least two hypotheses and five cues. Continuing our face recognizer, this would look like $P(\text{BARACK OBAMA} | \text{BROWN})$, which should be larger than $P(\text{BARACK OBAMA} | \text{BLUE})$ (he has brown eyes).

We now arrive at the fundamental problem, which is to calculate $P(h | c)$ after the cue arrives. This is accomplished by Bayes’ rule:

$$P(h | c) = \frac{P(c | h)P(h)}{P(c)}.$$ (6.5)

The denominator can be expressed as

$$P(c) = P(c | h_1)P(h_1) + P(c | h_2)P(h_2) + \cdots + P(c | h_n)P(h_n),$$ (6.6)

or it can be ignored as a normalization constant, at which point only relative likelihoods are calculated instead of proper probabilities.

The only thing accomplished by Bayes’ rule was to express $P(h | c)$ in terms of the prior distribution $P(h)$ and a new conditional distribution $P(c | h)$. The new conditional distribution is easy to work with in terms of modeling. It characterizes the likelihood that each specific cue will appear given that the hypothesis is true.

What if information arrives from a second cue detector? In this case, (6.5) is applied again, but $P(h | c)$ is now considered the prior distribution with respect to the new information. Let $D = \{d_1, d_2, \ldots, d_k\}$ represent the possible outputs of the new cue detector. Bayes’ rule becomes

$$P(h | c, d) = \frac{P(d | h)P(h | c)}{P(d|c)}.$$ (6.7)

Above, $P(d | h)$ makes what is called a conditional independence assumption: $P(d | h) = P(d | h, c)$. This is simpler from a modeling perspective. More generally, all four conditional parts of (6.7) should contain $c$ because it is given before $d$ arrives. As information from even more cues becomes available, Bayes’ rule is applied again as many times as needed. One difficulty that occurs in practice and modeled here is cognitive bias, which corresponds to numerous ways in which humans make irrational judgments in spite of the probabilistic implications of the data.

Multistable perception In some cases, our perceptual system may alternate between two or more conclusions. This is called multistable perception, for which the special case of two conclusions is called bistable perception. Figure 6.26(a) shows two well-known examples. For the Necker cube, it is ambiguous which cube face that is parallel to the viewing plane is in the foreground. It is possible to switch between both interpretations, resulting in bistable perception. Figure 6.26(b) shows another example, in which may see a rabbit or a duck at various times. Another well-known example is called the spinning dancer illusion by Nobuyuki Kayahara. In that case, the silhouette of a rotating dancer is shown and it is possible to interpret the motion as clockwise or counterclockwise.

McGurk effect The McGurk effect is an experiment that clearly indicates the power of integration by mixing visual and auditory cues. A video of a person speaking is shown with the audio track dubbed so that the spoken sounds do not match the video. Two types of illusions were then observed. If “ba” is heard and “ga” is shown, then most subjects perceive “da” being said. This corresponds to a plausible fusion of sounds that explains the mismatch, but does not correspond to either original cue. Alternatively, the sounds may combine to produce a perceived “bga” in the case of “ga” on the sound track and “ba” on the visual track.

Implications for VR Not all senses are taken over by VR. Thus, conflict will arise because of mismatch between the real and virtual worlds. As stated several times, the most problematic case of this is vection, which is a sickness-causing conflict between visual and vestibular cues arising from apparent self motion in VR while remaining stationary in the real world; see Section 6.2. As another example of mismatch, the user’s body may sense that it is sitting in a chair, but the VR experience may involve walking. There would then be a height mismatch between the real and virtual worlds, in addition to mismatches based on proprioception and tactile sensing. In addition to mismatches among the senses, imperfections in the VR hardware, software, content, and interfaces cause inconsistencies in...
6.4. COMBINING SOURCES OF INFORMATION

comparison with real-world experiences. The result is that incorrect or untended interpretations may arise. Even worse, such inconsistencies may increase fatigue as our neural structures use more energy to interpret the confusing combination. In light of the McGurk effect, it is easy to believe that many unintended interpretations or perceptions may arise from a VR system that does not provide perfectly consistent cues.

VR is also quite capable of generating new multistable perceptions. One example, which actually occurred in the VR industry, involved designing a popup menu. Suppose the user is in a dark environment and a large menu comes rushing up to them. The user may perceive one of two cases: 1) the menu approaches the user, or 2) the user is rushing up to the menu. The vestibular sense should be enough to resolve whether the user is moving, but the visual sense is overpowering. Prior knowledge about which is happening helps yield the correct perception. Unfortunately, if the wrong interpretation is made, then VR sickness is increased due to the sensory conflict. This, our perceptual system could by tricked into an interpretation that is worse for our health!

Further Reading

Retinal image size: [24]
Muller-Lyer Illusion:
Motion detection circuitry: Barlow, Hill, 1963; [15] [17]
Wagon wheel continuous illumination: [63]
History of film: [16]
Many optical illusions shown and explained: [33]
Silent film speed: [16]
LCD backlight scanning
The dress analysis: [33]
For more details on decision theory: Chapter 9 of [35].
McGurk effect: [17]
Bayesian analysis and decision theory: [58].
Phi phenomenon and beta movement: [71] [59].
Adaptation to conflicts in VR: [23]
Chapter 7
Visual Rendering

7.1 Ray Tracing and Shading Models

Suppose that a virtual world has been defined in terms of triangular primitives. Furthermore, a virtual eye has been placed in the world to view it from some particular position and orientation. Using the full chain of transformations from Chapter 3, the location of every triangle is correctly positioned onto a virtual screen (this was depicted in Figure 3.13). The next steps are to determine which screen pixels are touched by the triangle and then illuminate them according to the physics of the virtual world.

An important condition must also be checked: For each pixel, is the triangle even visible to the eye, or will it be blocked by part of another triangle? This classic visibility computation problem dramatically complicates the rendering process. The general problem is to determine for any pair of points in the virtual world, whether the line segment that connects them intersects with any objects (triangles). If an intersection occurs, then the line-of-sight visibility between the two points is blocked. The main difference between the two major families of rendering methods is due to the how visibility is handled.

Object-order versus image-order rendering For rendering, we need to consider all combinations of objects and pixels. This suggests a nested loop. One way to resolve the visibility is to iterate over the list of all triangles and attempt to render each one to the screen. This is called object-order rendering, and is the main topic of Section ???. For each triangle that falls into the field of view of the screen, the pixels are updated only if the corresponding part of the triangle is closer to the eye than any triangles that have been rendered so far. In this case, the outer loop iterates over triangles whereas the inner loop iterates over pixels. The other family of methods is called image-order rendering, and it reverses the order of the loops: Iterate over the image pixels and for each one, determine which triangle should influence its RGB values. To accomplish this, the path of light waves that would enter each pixel is traced out through the virtual environment. This method will be covered first, and many of its components apply to object-order rendering as well.

Ray tracing To calculate the RGB values at a pixel, a viewing ray is drawn from the focal point through the center of the pixel on a virtual screen that is placed in the virtual world; see Figure ???. The process is divided into two phases:

1. ray casting, in which the viewing ray is defined and its nearest point of intersection among all triangles in the virtual world is calculated.
2. shading, in which the pixel RGB values are calculated based on material properties at the intersection point and the lighting conditions.

The first step is based entirely on the virtual world geometry. The second step uses simulated physics of the virtual world. Both the material properties of objects
7.1. RAY TRACING AND SHADING MODELS

Ray tracing and shading models

The first step in a ray tracing approach is called ray casting, which extends a viewing ray that corresponds to a particular pixel on the image. The task is to determine what part of the virtual world model is visible. This is the closest intersection point of the viewing ray and the set of all triangles.

and the lighting conditions are artificial, and are chosen to produce the desired effect, whether realism or fantasy. Remember that the ultimate judge is the user, who understands the image through perceptual processes.

Ray casting Calculating the first triangle hit by the viewing ray after it leaves the image pixel (Figure 7.1) is straightforward if we neglect the computational performance. Starting with the triangle coordinates, focal point, and the ray direction (vector), the closed-form solution involves basic operations from analytic geometry, including dot products, cross products, and the plane equation[]. For each triangle, it must be determined whether the ray intersects it. If not, then the next triangle is considered. If it does, then the intersection is recorded as the candidate solution only if it is closer than the closest intersection encountered so far. After all triangles have been considered, the closest intersection point will be found. Although this is simple, it is far more efficient to arrange the triangles into a 3D data structure. Such structures are usually hierarchical so that many can be eliminated from consideration by quick coordinate tests. Popular examples include BSP-trees and Bounding Volume Hierarchies[]. Algorithms that sort geometric information to obtain greater efficiency generally fall under computational geometry[]. In addition to eliminating many triangles from quick tests, many methods of calculating the ray-triangle intersection have been developed to reduce the number of operations. One of the most popular is the Möller-Trumbore intersection algorithm[].

Figure 7.1: The first step in a ray tracing approach is called ray casting, which extends a viewing ray that corresponds to a particular pixel on the image. The task is to determine what part of the virtual world model is visible. This is the closest intersection point of the viewing ray and the set of all triangles.

Lambertian shading Now consider lighting each pixel and recall the basic behavior of light from Section23. The virtual world simulates the real-world physics, which includes the spectral power distribution and spectral reflection function. Suppose that a point-sized light source is placed in the virtual world. Using the trichromatic theory from Section24, its spectral power distribution is sufficiently represented by R, G, and B values. If the viewing ray the surface shown in Figure7.2 then how should the object appear? Assumptions about the spectral reflection function are taken into account by a shading model. The simplest case is Lambertian shading, for which the angle that the viewing ray strikes the surface is independent of the resulting pixel R, G, B values. This corresponds to the case of diffuse reflection, which is suitable for a “rough” surface (recall Figure13). All that matters is the angle θ that the surface makes with respect to the light source. Let n be the outward surface normal and let ē be a vector from the surface intersection point to the light source. Assume both n and ē are unit vectors. The dot product n · ē = sin θ yields the amount of attenuation between 0 and 1 due to the tilting of the surface relative to the source. Think about how the effective area of the triangle is reduced due to its tilt. A pixel under the Lambertian shading model is illuminated as

\[
\begin{align*}
R &= d_R I_R \max(0, n \cdot ē) \\
G &= d_G I_G \max(0, n \cdot ē) \\
B &= d_R I_B \max(0, n \cdot ē),
\end{align*}
\]

in which (d_R, d_G, d_B) represents the spectral reflectance property of the material (triangle) and (I_R, I_G, I_B) is represents the spectral power distribution of the light source. Under the typical case of white light, I_R = I_G = I_B. For a white or gray material, we would also have d_R = d_G = d_B.

Each triangle is assumed to be on the surface of an object, rather than the object itself. Therefore, if the light source is behind the triangle, then the triangle should not be illuminated because it is facing away from the light (it cannot be lit from behind). To handle this case, the max function appears in (7.1) to avoid n · ē < 0.
7.1. RAY TRACING AND SHADING MODELS

Figure 7.3: In the Blinn-Phong shading model, the light reaching the pixel depends on the angle between the normal \( n \) and the bisector \( b \) of the \( \ell \) and \( v \). If \( n = b \), then ideal reflection is obtained, as in the case of a mirror.

Using vector notation, (7.1) can be compressed into

\[
L = dI \max(0, n \cdot \ell)
\]  

in which \( L = (R, G, B) \), \( d = (d_R, d_G, d_B) \), and \( I = (I_R, I_G, I_B) \).

Blinn-Phong shading Now suppose that the object is “shiny”. If it were a perfect mirror, then all of the light from the source would be reflected to the pixel only if they are perfectly aligned; otherwise, no light would reflect at all. Such full reflection would occur if \( v \) and \( \ell \) are the same angle with respect to \( n \). What if the two angles are close, but do not quite match? The Blinn-Phong shading model proposes that some amount of light is reflected, depending on the amount of surface shininess and the difference between \( v \) and \( \ell \). See Figure 7.3. The bisector \( b \) is the vector obtained by averaging \( \ell \) and \( v \):

\[
b = \frac{\ell + v}{\|\ell + v\|}.
\]

Using the compressed vector notation, the Blinn-Phong shading model sets the RGB pixel values as

\[
L = dI \max(0, n \cdot \ell) + sI \max(0, n \cdot b)^x.
\]

This additively takes into account shading due to both diffuse and specular components. The first term is just the Lambertian shading model. The second component causes increasing amounts of light to be reflected as \( b \) becomes closer to \( n \). The exponent \( x \) is a material property that expresses the amount of surface shininess. A lower value, such as \( x = 100 \) results in a mild amount of shininess, whereas \( x = 10000 \) would make the surface almost like a mirror. This shading model does not correspond directly to the physics of the interaction between light and surfaces. It is merely a convenient and efficient heuristic, but widely used in computer graphics.

Figure 7.4: A Bidirectional Reflectance Distribution Function, or BRDF, meticulously specifies the ratio of incoming and outgoing light energy for all possible perspectives.

Ambient shading Another heuristic is ambient shading, which gives causes an object to glow without being illuminated by a light source. This lights surfaces that fall into the shadows of all lights; otherwise, they would be completely black. In the real world this does not happen light interreflects between objects to illuminate an entire environment. Such propagation has not been taken into account in the shading model so far, thereby requiring a hack to fix it. Adding ambient shading yields

\[
L = L_a + dI \max(0, n \cdot \ell) + sI \max(0, n \cdot b)^x,
\]

in which \( L_a \) is the ambient light component.

Multiple light sources Typically, the virtual world contains multiple light sources. In this case, the light from each is combined additively at the pixel. The result for \( N \) light sources is

\[
L = L_a + \sum_{i=1}^{N} dI_i \max(0, n \cdot \ell_i) + sI_i \max(0, n \cdot b_i)^x,
\]

in which \( I_i, \ell_i, \) and \( b_i \) correspond to each source.

BRDFs The shading models presented so far are in widespread use due to their simplicity and efficiency, even though they neglect most of the physics. To account for shading in a more precise and general way, a bidirectional reflectance distribution function (BRDF) is constructed; see Figure 7.4. The \( \theta_i \) and \( \theta_r \) parameters represent the angles of light source and viewing ray, respectively, with respect to the surface. The \( \phi_i \) and \( \phi_r \) parameters range from 0 to \( 2\pi \) and represent the angles made by the light and viewing vectors when looking straight down on the surface (the vector \( n \) would point at your eye).

The BRDF is a function of the form

\[
f(\theta_i, \phi_i, \theta_r, \phi_r) = \frac{\text{radiance}}{\text{irradiance}}.
\]
in which \(\text{radiance}\) is the light energy reflected from the surface in directions \(\theta_i\) and \(\phi_i\), and \(\text{irradiance}\) is the light energy arriving at the surface from directions \(\theta_i\) and \(\phi_i\). These are expressed at a differential level, roughly corresponding to an infinitesimal surface patch. Informally, it is the ratio of the amount of outgoing light to the amount of outgoing light at one point on the surface. The previous shading models can be expressed in terms of a simple BRDF. For Lambertian shading, the BRDF is constant because the surface reflects equally in all directions. The BRDF and its extensions can account for much more complex and physically correct lighting effects for a wide variety of surface textures. See Chapter 7 of [3] for extensive coverage.

Global illumination Recall that the ambient shading term \([\text{ambient}]\) was introduced to prevent surfaces in the shadows of the light source from appearing black. The computationally intensive but proper way to fix this problem is to calculate how light reflects from object to object in the virtual world. In this way, objects are illuminated \textit{indirectly} from the light that reflects from others, just like the real world. Unfortunately, this effectively turns all object surfaces into potential sources of light. This means that ray tracing must account for multiple reflections. This requires considering piecewise linear paths from the light source to the viewpoint, in which each bend corresponds to a reflection. An upper limit is set on the number of bounces to consider. The simple Lambertian and Blinn-Phong models are often used, but more general BRDFs are also common. Increasing levels of realism can be calculated, but with corresponding increases in computation time.

VR inherits all of the common issues from computer graphics, but also contains unique challenges. Chapters 5 and 6 mentioned the increased resolution and frame rate requirements. This provides strong pressure to reduce rendering complexity. Furthermore, many heuristics that worked well for graphics on a screen may be perceptibly wrong in VR. The combination of high field-of-view, resolution, and stereo images may bring out problems. For example, Figure 7.3 illustrates how differing viewpoints from stereopsis could affect the appearance of shiny surfaces. In general, some rendering artifacts could even contribute to VR sickness. Throughout the remainder of this chapter, complications that are unique to VR will be increasingly discussed.

7.2 Rasterization

The ray casting operation quickly becomes a bottleneck. For a 1080p image at 90Hz, it would need to be performed over 180 million times per second, and the ray-triangle intersection test would be performed for every triangle (although data structures such as a BSP would quickly eliminate many from consideration). It most common cases, it is much more efficient to switch from such image-order rendering to object-order rendering. The objects in our case are triangles and the resulting process is called \textit{rasterization} and is the main function of modern graphical processing units (GPUs). In this case, an image is rendering by iterating over every triangle and attempting to color the pixels where the triangle lands on the image. The main problem is that the method must solve the unavoidable problem of determining which part, if any, of the triangle is the closest to the focal point (roughly, the location of the virtual eye).

One way to solve it is to sort the triangles in \textit{depth order} so that the closest triangle is first. This enables the triangles to be drawn on the screen in back-to-front order. If they are properly sorted, then any later triangle to be rendered will rightfully clobber the image of previously rendered triangles at the same pixels. They can be drawn one-by-one while totally neglecting the problem of determining which is nearest. This is known as the \textit{Painter’s algorithm}. The main flaw, however, is the potential existence of \textit{depth cycles}, shown in Figure 7.3, in which three or more triangles cannot be rendered correctly in any order by the Painter’s algorithm. One possible fix is to detect such cases and split the triangles.

Depth buffer A simple and efficient method to resolve this problem is to manage the depth problem on a pixel-by-pixel basis by maintaining a \textit{depth buffer} (also called \textit{z-buffer}), which for every pixel records the distance of the triangle from the focal point to the intersection point of the ray that intersects the triangle at that pixel. In other words, if this were the ray casting approach, it would be distance along the ray from the focal point to the intersection point. Using this method, the triangles can be rendered in arbitrary order. It is also commonly applied to compute the effect of shadows by determining depth order from a light source, rather than the viewpoint. Objects that are closer to the light cast a shadow on further objects.

The depth buffer stores a positive real number (floating-point number in practice) at every pixel location. Before any triangles have been rendered, a maximum value (floating-point infinity) is stored at every location to reflect that no surface has yet been encountered at each pixel. At any time in the rendering process, each value in the depth buffer records the distance of the point on the most recently rendered triangle to the focal point, for the corresponding pixel in the image.
7.2. RASTERIZATION

Initially, all depths are at maximum to reflect that no triangles were rendered yet. Each triangle is rendered by calculating a rectangular part of the image that fully contains it. This is called a bounding box. The box is quickly determined by transforming all three of the triangle vertices to determine the minimum and maximum values for \( i \) and \( j \) (the row and column indices). An iteration is then performed over all pixels inside of the bounding box to determine which ones lie in inside the triangle and should therefore be rendered. This can be quickly determined by forming the three edge vectors shown in Figure 7.6 as

\[
e_1 = p_2 - p_1 \\
e_2 = p_3 - p_2 \\
e_3 = p_1 - p_3.
\]  

The point \( p \) lies inside the triangle if and only if

\[
p \times e_1 < 0, \ p \times e_2 < 0, \ p \times e_3 < 0,
\]

in which \( \times \) denotes the standard vector cross product. These three conditions ensure that \( p \) is “to the left” of each edge vector.

Barycentric coordinates As each triangle is rendered, information from it is mapped from the virtual world onto the screen. This is usually accomplished using barycentric coordinates (see Figure 7.7), which expresses each point in the triangle interior as a weighted average of the three vertices:

\[
p = \alpha_1 p_1 + \alpha_2 p_2 + \alpha_3 p_3
\]

for which \( 0 \leq \alpha_1, \alpha_2, \alpha_3 \leq 1 \) and \( \alpha_1 + \alpha_2 + \alpha_3 = 1 \). The closer \( p \) is to a vertex \( p_i \), the larger the weight \( \alpha_i \). If \( p \) is at the centroid of the triangle, then \( \alpha_1 = \alpha_2 = \alpha_3 = 1/3 \). If \( p \) lies on an edge, then the opposing vertex weight is zero. For example, if \( p \) lies on the edge between \( p_1 \) and \( p_2 \), then \( \alpha_3 = 0 \). If \( p \) lies on a vertex, \( p_i \), then \( \alpha_i = 1 \), and the other two barycentric coordinates are zero.

The coordinates are calculated using Cramer’s rule to solve a resulting linear system of equations. More particularly, let \( d_{ij} = e_i \cdot e_j \) for all combinations of \( i \) and \( j \).

\[
s = 1/(d_{11}d_{22} - d_{12}d_{12}).
\]

The coordinates are then given by

\[
\begin{align*}
\alpha_1 &= s * (d_{22}d_{33} - d_{12}d_{32}) \\
\alpha_2 &= s * (d_{11}d_{32} - d_{12}d_{31}) \\
\alpha_3 &= 1 - \alpha_1 - \alpha_2.
\end{align*}
\]

The same barycentric coordinates may be applied to the points on the model in \( \mathbb{R}^3 \), or on the resulting 2D projected points (with \( i \) and \( j \) coordinates) in the image plane. In other words, \( \alpha_1, \alpha_2, \) and \( \alpha_3 \) refer to the same point on the model both before, during, and after the entire chain of transformations from Section 4.2.

Furthermore, given the barycentric coordinates, the test in (7.9) can be replaced by simply determining whether \( \alpha_1 \geq 0, \alpha_2 \geq 0, \) and \( \alpha_3 \geq 0 \). If any barycentric coordinate is less than zero, then \( p \) must lie outside of the triangle.

Mapping the surface Barycentric coordinates provide a simple and efficient method for linearly interpolating values across a triangle. The simple case is the
7.2. RASTERIZATION

Figure 7.8: Texture mapping: A simple pattern or an entire image can be mapped across the triangles and then rendered in the image to provide much more detail than provided by the triangles in the model. (Figure from Wikipedia.)

propagation of RGB values. Suppose RGB values are calculated at the three triangle vertices using the shading methods of Section 7.1. This results in values \((R_i, G_i, B_i)\) for each \(i\) from 1 to 3. For a point \(p\) in the triangle with barycentric coordinates \((\alpha_1, \alpha_2, \alpha_3)\), the RGB values for the interior points are calculated as

\[
\begin{align*}
R &= \alpha_1 R_1 + \alpha_2 R_2 + \alpha_3 R_3 \\
G &= \alpha_1 G_1 + \alpha_2 G_2 + \alpha_3 G_3 \\
B &= \alpha_1 B_1 + \alpha_2 B_2 + \alpha_3 B_3.
\end{align*}
\]

(7.13)

The object need not maintain the same properties over an entire triangular patch. With texture mapping, a repeating pattern, such as tiles or stripes can be propagated over the surface; see Figure 7.8. More generally, any digital picture can be mapped onto the patch. The barycentric coordinates reference a point inside of the image to be used to influence a pixel. The picture or “texture” is treated as if it were painted onto the triangle; the lighting and reflectance properties are additionally taken into account for shading the object.

Another possibility is normal mapping, in affects the shading process by allowing the surface normal to be artificially varied over the triangle, even though geometrically it is impossible. Recall from Section 7.1 that the normal is used in the shading models. By allowing it to vary, artificial curvature can be given to an object. An important case of normal mapping is called bump mapping, which makes a flat surface look rough by irregularly perturbing the normals. If the normals appear to have texture, then the surface will look rough after shading is computed.

Figure 7.9: Bump mapping: By artificially altering the surface normals, the shading algorithms produce an effect that looks like a rough surface. (Figure by Brian Vibber.)

Figure 7.10: (a) The rasterization stage results in aliasing; straight edges appear to be staircases. (b) Pixels are selected for inclusion based on whether their center point \(p\) lies inside of the triangle.
7.2 RASTERIZATION

Figure 7.11: A mipmap stores the texture at multiple resolutions so that it can be appropriately scaled without causing significant aliasing. The overhead for storing the extra image is typically only 1/3 the size of the original (largest) image. (The image is from NASA and the mipmap was created by Wikipedia user Mulad.)

Aliasing Several artifacts arise due to discretization. Aliasing problems were mentioned in §5.4, which result in perceptible staircases in the place of straight lines, due to insufficient pixel density. Figure 7.10(a) shows the pixels selected inside of a small triangle by using (7.9). The point $p$ usually corresponds to the center of the pixel, as shown in Figure 7.10(b). Note that the point may be inside of the triangle while the entire pixel is not. Likewise, part of the pixel might be inside of the triangle while the center is not. You may notice that Figure 7.10 is not entirely accurate due to the subpixel mosaics used in displays (recall Figure 5.22). To be more precise, aliasing analysis should take this into account as well.

By deciding to fully include or exclude the triangle based on the coordinates of $p$ alone, the staircasing effect is unavoidable. A better way is to render the pixel according to the fraction of the pixel region that is covered by the triangle. This way its values could be blended from multiple triangles that are visible within the pixel region. Unfortunately, this requires supersampling, which means casting rays at a much higher density than the pixel density so that the triangle coverage fraction can be estimated. This dramatically increases cost. Commonly, a compromise is reached in a method called multisample anti-aliasing (or MSAA), in which only some values are calculated at the higher density. Typically, depth values are calculated for each sample, but shading is not.

A spatial aliasing problem results from texture mapping. The viewing transformation may dramatically reduce the size and aspect ratio of the original texture as it is mapped from the virtual world onto the screen. This may leave insufficient resolution to properly represent a repeating pattern in the texture; see Figure 7.12(a). This problem is often addressed in practice by pre-calculating and storing a mipmap for each texture; see Figure 7.11. The texture is calculated at various resolutions by performing high-density sampling and stored in images. Based on the size and viewpoint of the triangle on the screen, the appropriate scaled texture image is selected and mapped onto the triangle to reduce the aliasing artifacts.

Culling In practice, many triangles can be quickly eliminated before attempting to render them. This results in a preprocessing phase of the rendering approach called culling, which dramatically improves performance and enables faster frame rates. The efficiency of this operation depends heavily on the data structure used to represent the triangles. Thousands of triangles could be eliminated with a single comparison of coordinates if they are all arranged in a hierarchical structure. The most basic form of culling is called view volume culling, which eliminates all triangles that are wholly outside of the viewing frustum (recall Figure 3.18). For a VR headset, the frustum may have a curved cross section due to the limits of the optical system (see Figure 7.13). In this case, the frustum must be replaced with a region that has the appropriate shape. In the case of a truncated cone, a simple geometric test can quickly eliminate all objects outside of the view. For example, if

$$\sqrt{x^2 + y^2} - z > \tan \theta,$$

(7.14)
in which $2\theta$ is the angular field of view, then the point $(x, y, z)$ is outside of the cone. Alternatively, the stencil buffer can be used in a GPU to mark all pixels that would be outside of the lens view. These are quickly eliminated from consideration by a simple test as each frame is rendered.

Figure 7.12: (a) Due to the perspective transformation, the tiled texture suffers from spatial aliasing as the distance increases. (b) The problem can be fixed by performing supersampling.
Another form is called \textit{backface culling}, which removes triangles that have outward surface normals that point away from the focal point. These should not be rendered “from behind” if the model is consistently formed. Additionally, \textit{occlusion culling} may be used to eliminate parts of the model that might be hidden from view by a closer object. This can get complicated because it once again considers the depth ordering problem. For complete details, see \[3\].

\begin{figure}[h]
\centering
\includegraphics[width=0.45\textwidth]{fig7.13.png}
\caption{Due to optical system in front of the screen, the viewing frustum is replaced by a truncated cone in the case of a circularly symmetric view. Other cross-sectional shapes may be possible to account for the asymmetry of each eye view (for example, the nose is obstructing part of the view).}
\end{figure}

**VR-specific rasterization problems** The staircasing problem due to aliasing is expected to be worse for VR because current resolutions are well below the required retina display limit calculated in Section \[5.3\]. The problem is made significantly worse by the continuously changing viewpoint due to head motion. Even as the user attempts to fuse at an edge, the “stairs” appear to be more like an “escalator” because the exact choice of pixels to include in a triangle depends on subtle variations in the viewpoint. As part of our normal perceptual processes, our eyes are drawn toward this distracting motion. With stereo viewpoints, the situation is worse: The “escalator” from the right and left images will usually not match. As the brain attempts to fuse the two images into one coherent view, the aliasing artifacts provide a strong, moving mismatch. Reducing contrast at edges and using anti-aliasing techniques help alleviate the problem, but aliasing is likely to remain a significant problem until displays reach the required retina display density for VR.

A more serious difficulty caused by the enhanced depth perception afforded by a VR system. Both head motions and stereo views enable us to perceive small differences in depth across surfaces. This should be a positive outcome; however, many tricks developed in computer graphics over the decades rely on the fact that people cannot perceive these differences when a virtual world is rendered onto a fixed screen that is viewed from a significant distance. The result is that texture maps may look fake. For example, texture mapping a picture of a carpet onto the floor might inadvertently cause the floor to look as it were simply painted. In the real world we would certainly be able to distinguish painted carpet from real carpet. The same problem occurs with normal mapping. A surface that might look rough in a single static image due to bump mapping could look completely flat in VR as both eyes converge onto the surface. Thus, as the quality of VR systems improves, we should expect the rendering quality requirements to increase, causing many old tricks to be modified or abandoned.

### 7.3 Correcting Optical Distortions
Recall from Section \[4.3\] that barrel and pincushion distortions are common for an optical system with a high field of view (Figure \[4.20\]). When looking through the lens of a VR headset such as the Oculus Rift DK2, a pincushion distortion usually results. If the images are drawn on the screen without any correction, then the virtual world appears to be incorrectly warped. If the user yaws his head back and forth, fixed lines in the world, such as walls, appear to dynamically change their curvature because the distortion in the periphery is much stronger than in the center. If it is not corrected, then the perception of stationarity will fail because static objects should not appear to be warping dynamically. Furthermore, contributions may be made to VR sickness because incorrect accelerations are being visually perceived near the periphery.

How can this problem be solved? One way to avoid this effect is to replace the classical optical system with \textit{digital light processing (DLP)} technology that directly projects light into the eye using MEMS technology. Another way to greatly reduce this problem is to use a \textit{Fresnel lens} (see Figure \[7\]), which more accurately controls the bending of light rays by using a corrugated or sawtooth surface. This is used, for example, in the HTC Vive VR headset. One unfortunate side effect of Fresnel lenses is that glare can be frequently observed as light scatters across the ridges along the surface.

Whether small or large, the distortion can also be corrected in software. One assumption is that the distortion is circularly symmetric. This means that the amount of distortion depends only on the distance from the lens center, and not the particular direction from the center. Even if the lens distortion is perfectly circularly symmetric, it must also be placed so that it is centered over the eye. Some headsets offer IPD adjustment, which allows the distance between the lenses to be adjusted so that they are matched to the user’s eyes. If the eye is not centered on the lens, the asymmetric distortion arises. The situation is not perfect because as the eye rotates, the pupil moves along a spherical arc. As the position of the pupil over the lens changes laterally, the distortion varies and becomes asymmetric. This motivates making the lens as large as possible so that this problem is reduced. Another factor is that the distortion will change as the distance between the lens and the screen is altered. This adjustment may be useful to accommodate users with nearsightedness or farsightedness, as done in the Samsung Gear VR headset. The adjustment is also common in binoculars and binoculars, which explains why many people do not need their glasses to use them. To handle distortion correctly,
the headset should sense this adjustment and take it into account.

To fix radially symmetric distortion, suppose that the transformation chain $T_{\text{opt}}T_{\text{vr}}T_{\text{head}}$ has been applied to the geometry, resulting in the canonical view volume, as covered in Section 7.2. All points that were inside of the viewing frustum now have $x$ and $y$ coordinates ranging from $-1$ to $1$. Consider referring to these points using polar coordinates $(r, \theta)$:

$$r = x^2 + y^2$$
$$\theta = \text{atan2}(y, x),$$

in which $\text{atan2}$ represents the inverse tangent of $y/x$. This function is commonly used in programming languages to return angles $\theta$ over the entire range from 0 to $2\pi$. (The arctangent alone cannot do this because the quadrant that $(x, y)$ came from is needed.)

We now express the lens distortion in terms of transforming the radius $r$, without affecting the direction $\theta$ (because of symmetry). Let $f$ denote a function that applies to positive real numbers and distorts the radius. Let $r_u$ denote the undistorted radius, and let $r_d$ denote the distorted radius. Both pincushion and barrel distortion are commonly approximated using polynomials with odd powers, resulting in $f$ being defined as

$$r_d = f(r_u) = r_u + c_1 r_u^3 + c_2 r_u^5,$$

in which $c_1$ and $c_2$ are suitably chosen constants. If $c_1 < 0$, then barrel distortion occurs. If $c_1 > 0$, then pincushion distortion results. Higher-order polynomials could also be used, such as adding a term $c_3 r_u^7$ on the right above; however, in practice this is often considered unnecessary.

Correcting the distortion involves two phases:

1. Determine the radial distortion function $f$ for a particular headset, which involves a particular lens placed at a fixed distance from the screen. This is a regression or curve-fitting problem that involves an experimental setup that measures the distortion of many points and selects the coefficients $c_1$, $c_2$, and so on, that provide the best fit.

2. Determining the inverse of $f$ so that it be applied to the rendered image before the lens causes its distortion. The composition of the inverse with $f$ should cancel out the distortion function.

Unfortunately, polynomial functions generally do not have inverses that can be determined or even expressed in a closed form. Therefore, approximations are used. One commonly used approximation is

$$f^{-1}(r_d) \approx \frac{c_1 r_u^3 + c_2 r_u^5 + c_3 r_u^7 + c_4 r_u^9 + 2c_1 c_2 r_u^6}{1 + 4c_1 r_u^2 + 6c_2 r_u^4 + 4c_3 r_u^6 \approx c}.$$ (7.17)

Alternatively, the inverse can be calculated very accurately off-line and then stored in an array for fast access. It needs to be done only once per headset design.

Linear interpolation can be used for improved accuracy. The inverse values can be accurately calculated using Newton’s method, with initial guesses provided by simply plotting $f(r_u)$ against $r_u$ and swapping the axes.

The transformation $f^{-1}$ could be worked directly into the perspective transformation, thereby replacing $T_y$ and $T_{\text{tan}}$ with a nonlinear operation. By leveraging the existing graphics rendering pipeline, it is instead handled as a post-processing step. The process of transforming the image is sometimes called distortion shading because it can be implemented as a shading operation in the GPU; it has nothing to do with “shading” as defined in Section 7.2. The rasterized image that was calculated using methods in Section 7.2 can be converted into a transformed image using (7.17), or another representation of $f^{-1}$, on a pixel-by-pixel basis. If compensating for a pincushion distortion, the resulting image will appear to have a barrel distortion; see Figure ??.

To improve efficiency, multi-resolution shading is used in Nvidia GPUs to improve performance for VR [ ]. One problem is that the resolution is effectively dropped near the periphery because of the transformed image; see Figure ??.

This results in wasted shading calculations in the original image. Instead, the image can be rendered before the transformation by taking into account the final resulting resolutions after the transformation. A lower-resolution image is rendered in a region that will become compressed by the transformation.

The methods described in this section may also be used for other optical distortions that are radially symmetric. For example, chromatic aberration can be partially corrected by transforming the red, green, and blue subpixels differently. Each color is displaced radially by a different amount to compensate for the radial distortion that occurs based on its wavelength. If chromatic aberration correction is being used, then you can remove the lenses from a VR headset and see that the colors are not perfectly aligned in the images being rendered to the display. The system must create a distortion of pixel placements on the basis of color so that they will be moved closer to the correct places after they pass through the lens.

### 7.4 Improving Latency and Frame Rates

Figure ?? illustrates the motion-to-photons latency in a VR headset. This is the amount of time it takes to illuminate the display in response to a change in head orientation and position. For example, suppose the user is fixating on a stationary feature in the virtual world. As the head yaws to the right, the image of the feature on the display must immediately shift to the left. Otherwise, the feature will appear to move if the eyes remain fixated on it. This breaks the perception of stationary.

The perfect system As a thought experiment, imagine the perfect VR system. As the head moves, the viewpoint must accordingly change for visual rendering. A magic oracle perfectly indicates the head position and orientation at any time. The WVG continuously maintains the positions and orientations of all objects
in the virtual world. The visual rendering system maintains all perspective and viewport transformations, and the entire rasterization process continuously sets the RGB values on the display according to the shading models. Progressing with this fantasy, the display itself continuously updates, taking no time to switch the pixels, and it has a VR retina display resolution, as described in Section 5.4. In this case, visual stimulation provided by the virtual world should match what would occur in a similar physical world in terms of the geometry. There would be no errors in time and space (although the physics might not match anyway due to assumptions about lighting, shading, material properties, color spaces, and so on).

**Historical problems** In practice, the perfect system is not realizable. All of these operations require time to propagate information and perform computations. In early VR systems, the total motion-to-photons latency was often over 100ms. In the 1990s, 60ms was considered an acceptable amount. Latency has been stated as one of the greatest causes of VR sickness, and therefore one of the main obstructions to widespread adoption over the past decades. People generally adapt to a fixed latency, which somewhat mitigates the problem, but then causes problems when they have to readjust to the world world[]. Variable latencies are even worse due to the inability to adapt[]. Fortunately, latency is no longer the main problem in most VR systems because of the latest-generation sensing, GPU, and display technology. The latency may be around 15 to 25ms, which is even compensated for by predictive methods in the tracking systems. The result is that the effective latency is very close to zero. Thus, other factors are now contributing more strongly to VR sickness and fatigue, such as vection and optical aberrations.

**A simple example** Let \( d \) be the density of the display in pixels per degree. Let \( \omega \) be the angular velocity of the head in degrees per second. Let \( \ell \) be the latency in seconds. Due to latency \( \ell \) and angular velocity \( \omega \), the image is shifted by \( d\omega\ell \) pixels. For example, if \( d = 40 \) pixels per degree, \( \omega = 50 \) degrees per second, and \( \ell = 0.02 \) seconds, then the image is incorrectly displaced by \( d\omega\ell = 4 \) pixels.

Give average and max head rotations, translations. Figure?

**Overview of latency reduction methods** The following strategies are used together to both reduce the latency and to minimize the side effects of any remaining latency:

1. Lower the complexity of the virtual world.
2. Improve rendering pipeline performance.
3. Remove delays along the path from the rendered image to switching pixels.
4. Use prediction to estimate future viewpoints and world states.

---

**Figure 7.14**: A variety of mesh simplification algorithms can be used to reduce the model complexity while retaining the most important structures. Shown here is a simplification of a hand model made by the open-source library CGAL. (Figure by Fernando Cacciola.)

5. Shift or distort the rendered image to compensate for last-moment viewpoint errors.

Each of these will be described in succession.

**Simplifying the virtual world** Recall from Section that the virtual world is composed of geometric primitives, which are usually 3D triangles arranged in a mesh. The chain of transformations and rasterization process must be applied for each triangle, resulting in a computational cost that is directly proportional to the number of triangles. Thus, a model that contains tens of millions of triangles will take orders of magnitude longer to render than one made of a few thousand. In many cases, we obtain models that are much larger than necessary. They can often be made much smaller (fewer triangles) with no perceptible difference, much in the same way that image, video, and audio compression works. Why are they too big in the first place? If the model was captured from a 3D scan of the real world, then it is likely to contain highly dense data. Capture systems such as the FARO Focus3D X Series capture large worlds while facing outside, and others, such as the Matter and Form MFSV1 capture a small object by rotating it on a
7.4. IMPROVING LATENCY AND FRAME RATES

Turntable. As with cameras, systems that construct 3D models automatically are focused on producing highly accurate and dense representations, which maximize the model size. In the case of purely synthetic worlds, a modeling tool such as Maya or Blender will automatically construct a highly accurate mesh of triangles over a curved surface. Without taking specific care of later rendering burdens, the model could quickly become unwieldy. Fortunately, it is possible to reduce the model size by using mesh simplification algorithms; see Figure 7.14. In this case, one must be careful to make sure that the simplified model will have sufficient quality from all viewpoints that might arise in the targeted VR system. In some systems, such as Unity 3D, reducing the number of different material properties across the model will also improve performance.

In addition to reducing the rendering time, a simplified model will also lower computational demands on the Virtual World Generator (VWG). For a static world, the VWG does not need to perform any updates after initialization. The user simply views the fixed world, as in the case of a panoramic photo. For dynamic worlds, the VWG maintains a simulation of the virtual world that moves all geometric bodies while satisfying physical laws that mimic the real world. It must handle the motions of any avatars, falling objects, moving vehicles, swaying trees, and so on. Differential equations that model motion laws may be integrated to place bodies correctly over time. These issues will be explained in Chapter 8 but for now it is sufficient to understand that the VWG must maintain a coherent snapshot of the virtual world each time a rendering request is made. Thus, the VWG has a frame rate in the same way as a display or visual rendering system. Each VWG frame corresponds to the placement of all geometric bodies for a common time instant. How many times per second can the VWG be updated? Can a high, constant rate of VWG frames be maintained? What happens when a rendering request is made while the VWG is in the middle of updating the world? If the rendering module does not wait for the VWG update to be completed, then some objects could be incorrectly placed because some are updated while others are not. Thus, the system should ideally wait until a complete VWG frame is finished before rendering. This suggests that the VWG update should be at least as fast as the rendering process, and the two should be carefully synchronized so that a complete, fresh VWG frame is always ready for rendering.

Improving rendering performance Any techniques that improve rendering performance in the broad field of computer graphics apply here; however, one must avoid cases in which side effects that were imperceptible on a computer display become noticeable in VR. It was already mentioned in Section 7.2 that texture and normal mapping methods are less effective in VR for this reason; many more discrepancies are likely to be revealed in coming years. Regarding improvements that are unique to VR, it was mentioned in Sections 7.2 and 7.3 that the stencil buffer and multiresolution shading can be used to improve rendering performance by exploiting the shape and distortion due to the lens in a VR headset. A further improvement is to perform rasterization for the left and right eyes in parallel in the GPU, using one processor for each. The two processes are completely independent. This represents an important first step, among many that are likely to come, in design of GPUs that are targeted specifically for VR.

From rendered image to switching pixels The problem of waiting for coherent VWG frames also arises in the process of rendering frames to the display: When it is time to scan out the rendered image to the display, it might not be finished yet. Recall from Section 5.4 that most displays have a rolling scanout in which the beam moves from the lower right back to the upper left of the screen to start the next frame. During this time, the beam is turned off to avoid drawing a diagonal streak across the frame, hence, the name “blanking.” Short blanking intervals also occurred as each horizontal line to bring the beam back from the right to the left.

In the era of digital displays, the scanning process in unnecessary, but nevertheless persists and causes some trouble. Suppose that a display runs at 100 FPS. In this case, a request to draw a new rendered image is made every 10ms. Suppose that vblank occurs for 2ms and the remaining 8ms is spent drawing lines on the display. If the new rasterized image is written to the video memory during the
7.4. IMPROVING LATENCY AND FRAME RATES

Figure 7.16: Buffering is commonly used in visual rendering pipelines to avoid tearing and lost frames; however, it introduces more latency, which is detrimental to VR. (Figure by Wikipedia user Cmglee.)

2ms of vblank, then it will be correctly drawn in the remaining 8ms. It is also possible to earn extra time through beam racing, which was mentioned in Section 5.4. However, if a new image is being written and passes where the beam is scanning it out, then a problem called tearing occurs because it appears as if the screen is torn into pieces. If the VWG and rendering system produce frames at 300 FPS, then parts or 3 or 4 images could appear on the display because the image changes several times while the lines are being scanned out. One solution to this problem is a method called vsync (pronounced “vee sink”), which is a flag that prevents the video memory from being written outside of the vblank interval.

Another strategy to avoid tearing is buffering, which is shown in Figure 7.16. The approach is simple for programmers because it allows the frames to be written in memory that is not being scanned for output to the display. The unfortunate side effect is that it increases the latency. For double buffering, a new frame is first drawn into the buffer and then transferred to the video memory during vblank. It is often difficult to control the rate at which frames are produced because the operating system may temporarily interrupt the process or alter its priority. In this case, triple buffering is an improvement that allows more time to render each frame. For avoiding tearing and providing smooth video game performance, buffering has been useful; however, it is detrimental to VR because of the increased latency.

Ideally, the displays should have a global scanout, in which all pixels are switched at the same time. This allows a much longer interval to write to the video memory and avoids tearing. It would also reduce the latency in the time it takes to scan the first pixel to the last pixel. In our example, this was an 8ms interval. Finally, displays should reduce the pixel switching time as much as possible. In an smartphone LCD screen, it could take up to 20ms to switch pixels; however, OLED pixels can be switched in under 0.1ms.

The power of prediction For the rest of this section, we consider how to live with whatever latency remains. As a thought experiment, imagine that a fortune teller is able to accurately predict the future. With such a device, it should be possible to eliminate all latency problems. We would want to ask the fortune teller the following:

1. At what future time will the pixels be switching?
2. What will be the positions and orientations of all virtual world models at that time?
3. Where will the user be looking at that time?

Let \( t_s \) be answer to the first question. We need to ask the VWG to produce a frame for time \( t_s \) and then perform visual rendering for the user’s viewpoint at time \( t_s \). When the pixels are switched at time \( t_s \), then the stimulus will be presented to the user at the exact time and place it is expected. In this case, there is zero effective latency.

Now consider what happens in practice. First note that using information from all three questions above implies significant time synchronization across the VR system: All operations must have access to a common clock. For the first question above, determining \( t_s \) should be feasible if the computer is powerful enough and the VR system has enough control from the operating system to ensure that VWG frames will be consistently produced and rendered at the frame rate. The second question is easy for the case of a static virtual world. In the case of a dynamic world, it might be straightforward for all bodies that move according to predictable physical laws. However, it is difficult to predict what humans will do in the virtual world. This complicates the answers to both the second and third questions.

Fortunately, the latency is so small that momentum and inertia play a significant role; see Chapter 8. Bodies in the matched zone are following physical laws of motion from the real world. These motions are sensed and tracked according to methods covered in Chapter 8. Although it might be hard to predict where you
7.4. IMPROVING LATENCY AND FRAME RATES

<table>
<thead>
<tr>
<th>Perturbation</th>
<th>Image effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta \alpha$ (yaw)</td>
<td>Horizontal shift</td>
</tr>
<tr>
<td>$\Delta \beta$ (pitch)</td>
<td>Vertical shift</td>
</tr>
<tr>
<td>$\Delta \gamma$ (roll)</td>
<td>Rotation about image center</td>
</tr>
<tr>
<td>$x$</td>
<td>Horizontal shift</td>
</tr>
<tr>
<td>$y$</td>
<td>Vertical shift</td>
</tr>
<tr>
<td>$z$</td>
<td>Contraction or expansion</td>
</tr>
</tbody>
</table>

Figure 7.17: Six cases of post-rendering image warp based on the degrees of freedom for a change in viewpoint. The first three correspond to an orientation change. The remaining three correspond to a position change. These operations can be visualized by turning on a digital camera and observing how the image changes under each of these perturbations.

will be looking in 5 seconds, it is possible to predict with very high accuracy where your head will be positioned and oriented in 20ms. You have no free will on the scale of 20ms! Instead, momentum dominates and the head motion can be accurately predicted. Some body parts, especially fingers, have much less inertia, and therefore become more difficult to predict; however, these are not as important as predicting head motion. The viewpoint depends only on the head motion, and latency reduction is most critical in this case to avoid perceptual problems that lead to fatigue and VR sickness.

Post-rendering image warp Due to both latency and imperfections in the prediction process, a last-moment adjustment might be needed before the frame is scanned out to the display. This is called post-rendering image warp (it has also been rediscovered and called time warp in the recent VR industry []). At this stage, there is no time to perform complicated shading operations; therefore, a simple transformation is made to the image.

Suppose that an image has been rasterized for a particular viewpoint, expressed by position $(x, y, z)$ and orientation given by yaw, pitch, and roll $(\alpha, \beta, \gamma)$. What would be different about the image if it were rasterized for a nearby viewpoint? Based on the degrees of freedom for viewpoints, there are six types of adjustment; see Figure 7.17. Each one of these has a direction that is not specified in the figure. For example, if $\Delta \alpha$ is positive, which corresponds to a small, counterclockwise yaw of the viewpoint, then the image is shifted horizontally to the right.

Figure 7.17 shows some examples of the image warp. Most cases require the rendered image to be larger than the targeted display; otherwise, there will be no data to shift into the warped image; see Figure 7.17. In this case, it is perhaps best to repeat pixels from the edge [], rather than turning them black.

Flaws in the warped image Imagine warping due to orientation changes produces a correct image in the sense that it should be exactly what would have been rendered from scratch for that orientation (without taking aliasing issues into account). However, positional changes are incorrect! Perturbations in $x$ and $y$ do not account for motion parallax (recall from Section [], which would require knowing the depths of the objects. Changes in $z$ produce similarly incorrect images because nearby objects should expand or contract by a larger amount than further ones. To make matters worse, changes in viewpoint position might lead to visibility events, in which part of an object may become visible only in the new viewpoint; see Figure 7.17. As latencies become shorter and prediction becomes better, the amount of perturbation is reduced. Careful perceptual studies are needed to evaluate conditions under which image warping errors are perceptible or cause discomfort. An alternative to image warping is to use parallel processing to sample several future viewpoints and render images for all of them. The most correct image can then be selected, to greatly reduce the image warping artifacts.

Aspect graph / visibility complex problem! (Put this into figure caption)

Increasing the frame rate Post-rendering image warp can also be used to artificially increase the frame rate. For example, suppose that only one rasterized image is produced every 100 milliseconds by a weak computer or GPU. This would result in poor performance at 10 FPS. Suppose we would like to increase this to 100 FPS. In this case, a single rasterized image can be warped to produce frames every 10ms until the next rasterized image is computed; see Figure 7.17. In this case, 10 warped frames are used for every rasterized image that is computed.

7.5 Immersive Photos and Videos

Up until now, this chapter has focused on rendering a virtual world that was constructed synthetically from geometric models. The methods developed over the decades in computer graphics research have targeted this case. The trend has recently changed, though, toward capturing real-world images and video, which are then easily embedded into VR experiences. This change is mostly due to smartphone industry, which has led to hundreds of millions of people carrying high resolution cameras with them everywhere. Furthermore, 3D camera technology continues to advance, which provides distance information in addition to depth. All of this technology is quickly converging to the case of panoramas, which contained captured image data from all possible viewing directions. A current challenge is to also capture data within a region of all possible viewing positions and orientations.

Texture mapping onto a virtual screen Putting a photo or video into a virtual world is an application of texture mapping. Figure 7.18 shows a commercial use in which Netflix offers online movie streaming through the Samsung Gear VR headset. The virtual screen is a single rectangle, which may be viewed as a simple mesh consisting of two triangles. A photo can be mapped across any triangular
photosphere from a single camera in a single instant of time. Two obvious choices exist:

1. Take multiple images with one camera by pointing it in different directions each time, until the entire sphere of all viewing directions is covered.

2. Use multiple cameras, pointing in various viewing directions, so that all directions are covered by taking synchronized pictures.

The first case leads to a well-studied problem in computer vision and computational photography called image stitching. A hard version of the problem can be made by stitching together an arbitrary collection of images, from various cameras and times. This might be appropriate, for example, to build a photosphere of a popular tourist site from online photo collections. More commonly, a smartphone user may capture a photosphere by using the outward-facing camera and pointing the camera in enough directions. In this case, a software app builds the photosphere dynamically while images are taken in rapid succession. For the hard version, a difficult optimization problem arises in which features need to be identified and matched across overlapping parts of multiple images while unknown, intrinsic camera parameters are taken into account. Differences in perspective, optical aberrations, lighting conditions, exposure time, and changes in the scene over different times must be taken into account. In the case of using a smartphone app, the same camera is being used and the relative time between images is short; therefore, the task is much easier. Furthermore, by taking rapid images in succession and using internal smartphone sensors, it is much easier to match the overlapping image parts. Most flaws in such hand-generated photospheres are due to the user inadvertently changing the position of the camera while pointing it in various directions.

For the second case, a rig of identical cameras can be carefully designed so that all viewing directions are covered; see Figure 7.19. Once it is calibrated so that the relative positions and orientations of the cameras are precisely known, stitching the images together becomes straightforward. Corrections may nevertheless be applied to account for variations in lighting or calibration; otherwise, the seams in the stitching may become perceptible. A tradeoff exists in terms of the number of cameras. By using many cameras, very high resolution captures can be made with relatively little optical distortion because each camera contributes a narrow field-of-view image to the photosphere. At the other extreme, a few as two cameras are sufficient, as in the case of the Ricoh Theta. In this case, the cameras are pointed 180 degrees apart and a fish-eyed lens is able to capture a view that is larger than 180 degrees. This design dramatically reduces costs, but requires significant unwarping of the two captured images.

Capturing a wider field of view  Mapping onto a rectangle makes it easy to bring pictures or movies that were captured with ordinary cameras into VR; however, the VR medium itself allows great opportunities to expand the experience. Unlike life in the real world, the size of the virtual screen can be expanded without any significant cost. To fill the field of view of the user, it makes sense to curve the virtual screen and put the user at the center. Such curving already exists in the real world; examples are the 1950s Cinerama experience, which was shown in Figure 1.25(d), and modern curved displays. In the limiting case, we obtain a panoramic photo, sometimes called a photosphere. Displaying many photospheres per second leads to a panoramic movie, which we may call a moviesphere.

Recalling the way cameras work from Section 4.5, it is impossible to capture a panoramic photo, sometimes called a photosphere. Displaying many photospheres per second leads to a panoramic movie, which we may call a moviesphere.

Figure 7.18: (a) As of 2015, Netflix offers online movie streaming onto a large virtual TV screen while the user appears to sit in a living room. (b) The movies are texture-mapped onto the TV screen, frame by frame. Furthermore, the gaze pointer allows the user to look in a particular direction to select content.
7.5. IMMERSIVE PHOTOS AND VIDEOS

Figure 7.19: (a) The 360Heros Pro10 HD is a rig that mounts ten GoPro cameras in opposing directions to capture panoramic images. (b) The Ricoh Theta S captures panoramic photos and videos using only two cameras, each with a lens that provides a field of view larger than 180 degrees.

Figure 7.20: (a) The photosphere is texture mapped onto the interior of a sphere that is modeled as a triangular mesh. (b) A photosphere stored as a cube of six images can be quickly mapped to the sphere with relatively small loss of resolution; a cross section is shown here.

Perceptual issues Does the virtual world appear to be “3D” when viewing a photosphere or moviesphere? Recall from Section 6.1 that there are many more monocular depth cues than stereo cues. Due to the high field-of-view of modern VR headsets and monocular depth cues, a surprisingly immersive experience is obtained. Thus, it may feel more “3D” than people expect, even if the same part of the panoramic image is presented to both eyes. Many interesting questions remain for future research regarding the perception of panoramas. If different viewpoints are presented to the left and right eyes, then what should the radius of the virtual sphere be for comfortable and realistic viewing? Continuing further, suppose positional head tracking is used. This might improve viewing comfort, but the virtual world will appear more flat because parallax is not functioning. For example, closer objects will not move more quickly as the head moves from side to side. Can simple transformations be performed to the images so that depth perception is enhanced? Can limited depth data, which could even be extracted automatically from the images, greatly improve parallax and depth perception? Another issue is designing interfaces inside of photospheres. Suppose we would like to have a shared experience with other users inside of the sphere. In this case, how do we perceive virtual objects inserted into the sphere, such as menus or avatars? How well would a virtual laser pointer work to select objects?

Panoramic light fields Panoramic images are simple to construct, but clearly flawed because of their inability to take into account how the surround world
7.5 IMMERSIVE PHOTOS AND VIDEOS

Figure 7.21: The Pantoptican prototype from Figure Digital uses dozens of cameras to improve the ability to approximate more viewpoints so that stereo viewing and parallax from position changes can be simulated.

would appear from any viewpoint that obtained by user movement. To accurately determine this, the ideal situation would be to capture the entire light field of energy inside of whatever viewing volume that user is allowed to move. A light field provides the both the spectral power and direction of light propagation at every point in space. If the user is able to walk around in the physical world while wearing a VR headset, then this seems to be an impossible task. How can a rig of cameras capture the light energy in all possible locations at the same instant in an entire room? If the user is constrained to a small area, then the light field can be approximately captured by a rig of cameras arranged on a sphere; a prototype is shown in Figure 7.21. In this case, dozens of cameras may be necessary, and image warping techniques are used to approximate viewpoints between the cameras or from the interior the spherical rig. To further improve the experience, light-field cameras (also called plenoptic cameras) offer the ability to capture both the intensity of light rays and the direction that they are traveling through space. This offers many advantages, such as refocusing images to different depths, after the light field has already been captured.

Further Reading

Close connections exist between VR and computer graphics because both are required to push visual information onto a display. For more, consult basic textbooks [45]. For high performance rendering, see [3].

Data structures for ray tracing:

Further Reading

Close connections exist between VR and computer graphics because both are required to push visual information onto a display. For more, consult basic textbooks [45]. For high performance rendering, see [3].

Data structures for ray tracing:

Further Reading

Close connections exist between VR and computer graphics because both are required to push visual information onto a display. For more, consult basic textbooks [45]. For high performance rendering, see [3].

Data structures for ray tracing:
Chapter 8

Motion

We need to set up some math so that models can be modeled in both the real and virtual worlds. In the virtual case, it is used to design the VWG. In the real case, it is used to model how the user might move in the matched zone. This is needed to develop tracking systems.

Furthermore, we need these to describe the mismatch that occurs in the matched zone. These cause sickness.

This should explain more details of the VWG: It has to handle kinematics, dynamics, and collision detection.

8.1 Dynamical Systems

Kinematics, velocity, acceleration

Perhaps a simple control example? Moving a car?
What about multibody kinematics?

8.2 Physics in the Virtual World

Simulation

Collision detection

8.3 The Vestibular System

Needs to cover the full sense of balance, from physiology to perception.

8.4 Avatar Motion and Vection

Self motion here. What about moving the whole body through kinematic chains?
Chapter 9

Tracking

<table>
<thead>
<tr>
<th>Chapter Status</th>
<th>Taken from Virtual Reality, S. M. LaValle</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This online chapter is not the final version! Check <a href="http://msl.cs.uiuc.edu/vr/">http://msl.cs.uiuc.edu/vr/</a> for information on the latest draft version.</td>
</tr>
<tr>
<td></td>
<td>This draft was compiled on March 2, 2016.</td>
</tr>
</tbody>
</table>

Tracking the sense organs.
- Ideally, eyes. Currently: Head only.
- Ears go along with head. Can’t rotate independently.
- Tracking the rest of the body.
- Keyboard, mouse, Xbox controller (mechanical switch, potentiometers) - Hands, body, faces
- Tracking the rest of the environment (local or remote in the telepresence case).
- Moving objects, walls, outdoor scenes, other people

CALIBRATION IMPORTANT FOR SENSORS!

9.1 Estimating Orientation

9.2 Estimating Position and Orientation

Camera parameters Calibration parameters: intrinsic and extrinsic (Section 3.2).

There are five intrinsic parameters:

\[
\begin{bmatrix}
\alpha_x & \gamma & u_0 \\
0 & \alpha_y & v_0 \\
0 & 0 & 1
\end{bmatrix}
\]  

(9.1)

These account for focal length, image sensor format, and principal point.

Due to lens aberrations, there could also be nonlinear transforms in the model.
- P3P algebra.
- Incremental PnP
- Noise and ambiguity:
  Visibility approaches: Could be camera or lighthouse – same math!

9.3 Tracking Kinematic Structures

9.4 Localization and Mapping

Further Reading

Fusion of IMU and Vision for Absolute Scale Estimation: Nutzi, Scaramuzza, Weiss, Siegwart.
- Oculus VR blogs: 38, 36, 57
- Oculus Rift tracking: 59
Chapter 10

Interfaces

10.1 Locomotion
Remind about vection. Ways to overcome.

10.2 Manipulation

10.3 System Control

10.4 Social Interaction

10.5 Specialized Interfaces
Chapter 11

Evaluating VR Systems

<table>
<thead>
<tr>
<th>Chapter Status</th>
<th>Taken from Virtual Reality, S. M. LaValle</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Warning]</td>
<td>This online chapter is not the final version! Check <a href="http://msl.cs.uiuc.edu/vr/">http://msl.cs.uiuc.edu/vr/</a> for information on the latest draft version.</td>
</tr>
<tr>
<td></td>
<td>This draft was compiled on March 2, 2016.</td>
</tr>
</tbody>
</table>

11.1 Perceptual Training

Examples: Blind people can hear better, and deaf people can see better. Doctors and medical technicians train to read nonintuitive medical scans. Detectives train to look for clues at a crime scene. Question: What should VR engineers and developers train for?

11.2 Comfort and VR Sickness

Survey: [20]
Strobing (4Hz):
See work by: Tom Stoffregen, Visually induced motion sickness predicted by postural instability
Vection!

11.3 Design of Experiments

11.4 Best Practices

11.5 The Development Cycle
Chapter 12

Audio

12.1 Physics of Sound

12.2 The Physiology of Human Hearing

12.3 Aural Perception

12.4 Aural Rendering
Chapter 13

Frontiers

<table>
<thead>
<tr>
<th>Chapter Status</th>
<th>Taken from Virtual Reality, S. M. LaValle</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This online chapter is not the final version! Check <a href="http://msl.cs.uiuc.edu/vr/">http://msl.cs.uiuc.edu/vr/</a> for information on the latest draft version.</td>
</tr>
<tr>
<td></td>
<td>This draft was compiled on March 2, 2016.</td>
</tr>
</tbody>
</table>

13.1 Haptics
13.2 Taste and Smell
13.3 Robotic Interfaces
13.4 Brain-Machine Interfaces
Bibliography


---

**Index**

3:2 pull down, 201
3D glasses, 158
accelerometer, 40
accommodation, 108, 147
adaptation, 54, 139
additive mixtures, 169
aliasing, 169, 170
amacrine cells, 120
ambient shading, 181
angular velocity, 40
animation, 40
aperture, 119
aperture problem, 154
aqueous humour, 113
AR, 1
aspect ratio, 150
aspheric lens, 117
astigmatism, 113, 119
atmospheric cue, 113, 119
augmented reality, 1
avatar, 20
axis-angle, 159
backface culling, 189
ball and shadow illusion, 136
barrel distortion, 104, 191
barycentric coordinates, 183
Bayes’ rule, 168, 171
beam racing, 116, 197
beta movement, 188, 169
bidirectional reflectance distribution function, 158
bidirectional reflectance distribution function, 158
cue, 114
collision detection algorithm, 20
coherent light, 123
color constancy, 166
color space, 103, 104
coma, 169
cones, 119
configuration space, 49
convergence, 181
corneal, 181
cue, 114
depth, 114
depth, 114
monocular, 114
stereo, 114
...
quaternion, 76
rabbit duck illusion, 172
radiance, 161
rapid eye movements, 152
raster scan, 140
rasterization, 181
ratio principle, 157
ray casting, 176, 177
reading glasses, 113
receptive field, 120
receptor, 52, 59
refractive index, 46
Reichardt detector, 155
retina, 109, 119
retina display, 137
retinal image slip, 123
rigid body transform, 72
rods, 119
roll rotation, 70
rolling scanout, 130, 136
rolling shutter, 113, 130

saccadic masking, 152
saturation, 154
scanout, 130
sclera, 118
scotopic vision, 123
screen-door effect, 149, 157
sense organ, 52
sensor, 52
sensory cue, 143
sensory system selectivity, 158
shading, 176
shading model, 178
shadows
  depth cue, 148
  shutter, 113
shutter speed, 113
simulator sickness, 6
single-unit recordings, 49
size constancy scaling, 145, 147
size perception, 143
smooth pursuit, 132
spatial aliasing, 158, 158
spatial opponency, 126
spectral color, 53
spectral distribution function, 163
spectral power distribution, 163, 175
spectral reflection function, 171
specular reflection, 91
spherical aberration, 132
spinning dancer illusion, 172
spiral aftereffect, 150
sRGB, 178
static world, 195
stencil buffer, 138, 139
stereo depth cue, 143
stereo displays, 153
stereo rig, 143
stereopsis, 139
stereoscope, 23
Steven’s power law, 150
stroboscopic apparent motion, 150, 157
subjective constancy, 134
supersampling, 158, 158
synapse, 53
tearing, 140, 196, 197
teleoperation, 11
telepresence, 11
texture gradient, 144
texture mapping, 153, 200
thalamus, 127
time warp, 159
topographic mapping, 52
transducer, 52
trichromatic theory, 160, 178
triple buffering, 159
ultraviolet, 53
uncanny valley, 4
v, 115
value (HSV), 164
vblank, 196
vection, 155, 172, 193
vector field, 150
vergence, 154, 158
vergence-accommodation mismatch, 142
vertical blanking interval, 196
vestibulo-ocular reflex, 132
view volume culling, 182
viewing frustum, 84
viewing ray, 176
viewport transform, 85
virtual environments, 143
virtual prototyping, 143
virtual reality (definition), 5
Virtual World Generator (VWG), 25
visibility computation, 176
visibility constraints, 10
visibility events, 204
visual acuity, 118, 138
visual cortex, 127
vitreous humor, 118
VOR, 132
  gain adaptation, 139
VOR gain, 139
VR sickness, 6
vsync, 140, 197
VWG, 195
wagon-wheel effect, 155
waterfall illusion, 156
Weber’s law, 57
white light, 166
yaw rotation, 71
Young-Helmholtz theory, 166
z-buffer, 182, 188
zero effective latency, 198
zoetrope, 157, 158