Computer Vision:
Algorithms and Applications
2nd Edition

Richard Szeliski

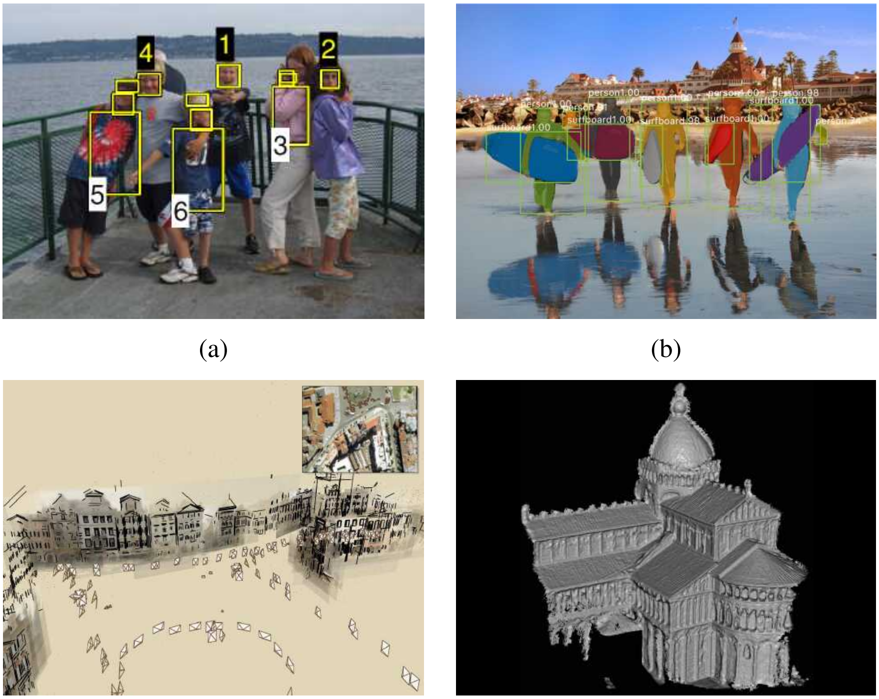
Final draft, September 30, 2021

Please refer interested readers to the book's Web site at
https://szeliski.org/Book, where you can also provide feedback.

Chapter 1

Introduction

Figure 1.1 *The human visual system has no problem interpreting the subtle variations in translucency and shading in this photograph and correctly segmenting the object from its background.*



(a) (b)

(c) (d)

Figure 1.2 *Some examples of computer vision algorithms and applications. (a)* Face detection *algorithms, coupled with color-based clothing and hair detection algorithms, can locate and recognize the individuals in this image (Sivic, Zitnick, and Szeliski 2006) © 2006 Springer. (b)* Object instance segmentation *can delineate each person and object in a complex scene (He, Gkioxari* et al. *2017) © 2017 IEEE. (c)* Structure from motion *algorithms can reconstruct a sparse 3D point model of a large complex scene from hundreds of partially overlapping photographs (Snavely, Seitz, and Szeliski 2006) © 2006 ACM. (d)* Stereo matching *algorithms can build a detailed 3D model of a building façade from hundreds of differently exposed photographs taken from the internet (Goesele, Snavely* et al. *2007) © 2007 IEEE.*

1.1 What is computer vision?

As humans, we perceive the three-dimensional structure of the world around us with apparent ease. Think of how vivid the three-dimensional percept is when you look at a vase of flowers sitting on the table next to you. You can tell the shape and translucency of each petal through the subtle patterns of light and shading that play across its surface and effortlessly segment each flower from the background of the scene (Figure 1.1). Looking at a framed group portrait, you can easily count and name all of the people in the picture and even guess at their emotions from their facial expressions (Figure 1.2a). Perceptual psychologists have spent decades trying to understand how the visual system works and, even though they can devise optical illusions[1] to tease apart some of its principles (Figure 1.3), a complete solution to this puzzle remains elusive (Marr 1982; Wandell 1995; Palmer 1999; Livingstone 2008; Frisby and Stone 2010).

Researchers in computer vision have been developing, in parallel, mathematical techniques for recovering the three-dimensional shape and appearance of objects in imagery. Here, the progress in the last two decades has been rapid. We now have reliable techniques for accurately computing a 3D model of an environment from thousands of partially overlapping photographs (Figure 1.2c). Given a large enough set of views of a particular object or façade, we can create accurate dense 3D surface models using stereo matching (Figure 1.2d). We can even, with moderate success, delineate most of the people and objects in a photograph (Figure 1.2a). However, despite all of these advances, the dream of having a computer explain an image at the same level of detail and causality as a two-year old remains elusive.

Why is vision so difficult? In part, it is because it is an *inverse problem*, in which we seek to recover some unknowns given insufficient information to fully specify the solution. We must therefore resort to physics-based and probabilistic *models*, or machine learning from large sets of examples, to disambiguate between potential solutions. However, modeling the visual world in all of its rich complexity is far more difficult than, say, modeling the vocal tract that produces spoken sounds.

The *forward* models that we use in computer vision are usually developed in physics (radiometry, optics, and sensor design) and in computer graphics. Both of these fields model how objects move and animate, how light reflects off their surfaces, is scattered by the atmosphere, refracted through camera lenses (or human eyes), and finally projected onto a flat (or curved) image plane. While computer graphics are not yet perfect, in many domains, such as rendering a still scene composed of everyday objects or animating extinct creatures such

[1] Some fun pages with striking illusions include https://michaelbach.de/ot, https://www.illusionsindex.org, and http://www.ritsumei.ac.jp/~akitaoka/index-e.html.

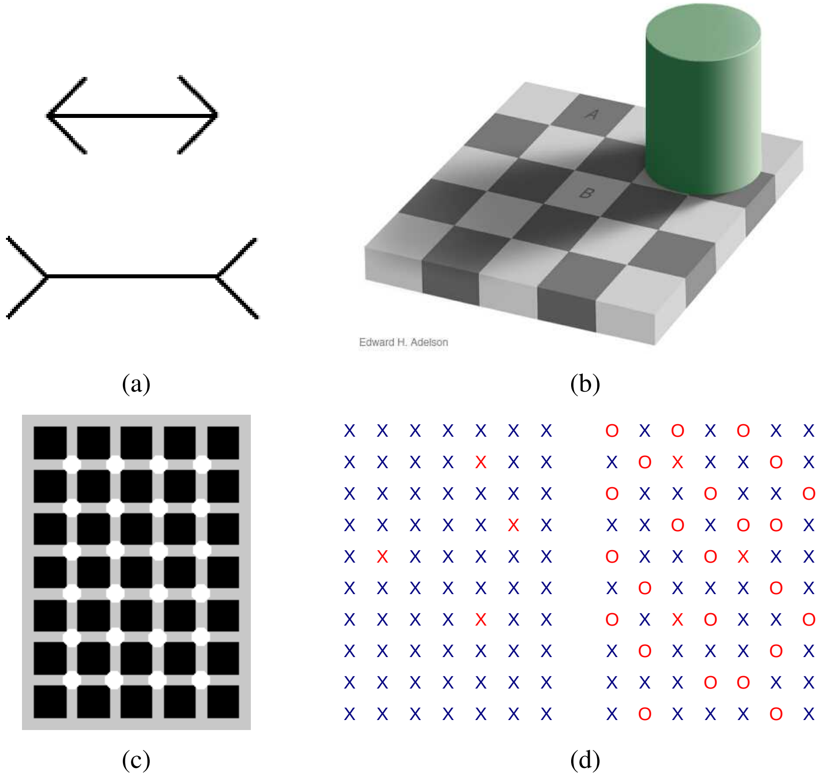


Figure 1.3 *Some common optical illusions and what they might tell us about the visual system: (a) The classic Müller-Lyer illusion, where the lengths of the two horizontal lines appear different, probably due to the imagined perspective effects. (b) The "white" square B in the shadow and the "black" square A in the light actually have the same absolute intensity value. The percept is due to* brightness constancy, *the visual system's attempt to discount illumination when interpreting colors. Image courtesy of Ted Adelson, http://persci.mit.edu/gallery/checkershadow. (c) A variation of the Hermann grid illusion, courtesy of Hany Farid. As you move your eyes over the figure, gray spots appear at the intersections. (d) Count the red* Xs *in the left half of the figure. Now count them in the right half. Is it significantly harder? The explanation has to do with a* pop-out *effect (Treisman 1985), which tells us about the operations of parallel perception and integration pathways in the brain.*

as dinosaurs, the illusion of reality is essentially there.

In computer vision, we are trying to do the inverse, i.e., to describe the world that we see in one or more images and to reconstruct its properties, such as shape, illumination, and color distributions. It is amazing that humans and animals do this so effortlessly, while computer vision algorithms are so error prone. People who have not worked in the field often underestimate the difficulty of the problem. This misperception that vision should be easy dates back to the early days of artificial intelligence (see Section 1.2), when it was initially believed that the *cognitive* (logic proving and planning) parts of intelligence were intrinsically more difficult than the *perceptual* components (Boden 2006).

The good news is that computer vision *is* being used today in a wide variety of real-world applications, which include:

- **Optical character recognition (OCR):** reading handwritten postal codes on letters (Figure 1.4a) and automatic number plate recognition (ANPR);

- **Machine inspection:** rapid parts inspection for quality assurance using stereo vision with specialized illumination to measure tolerances on aircraft wings or auto body parts (Figure 1.4b) or looking for defects in steel castings using X-ray vision;

- **Retail:** object recognition for automated checkout lanes and fully automated stores (Wingfield 2019);

- **Warehouse logistics:** autonomous package delivery and pallet-carrying "drives" (Guizzo 2008; O'Brian 2019) and parts picking by robotic manipulators (Figure 1.4c; Ackerman 2020);

- **Medical imaging:** registering pre-operative and intra-operative imagery (Figure 1.4d) or performing long-term studies of people's brain morphology as they age;

- **Self-driving vehicles:** capable of driving point-to-point between cities (Figure 1.4e; Montemerlo, Becker *et al.* 2008; Urmson, Anhalt *et al.* 2008; Janai, Güney *et al.* 2020) as well as autonomous flight (Kaufmann, Gehrig *et al.* 2019);

- **3D model building (photogrammetry):** fully automated construction of 3D models from aerial and drone photographs (Figure 1.4f);

- **Match move:** merging computer-generated imagery (CGI) with live action footage by tracking feature points in the source video to estimate the 3D camera motion and shape of the environment. Such techniques are widely used in Hollywood, e.g., in movies such as Jurassic Park (Roble 1999; Roble and Zafar 2009); they also require the use of

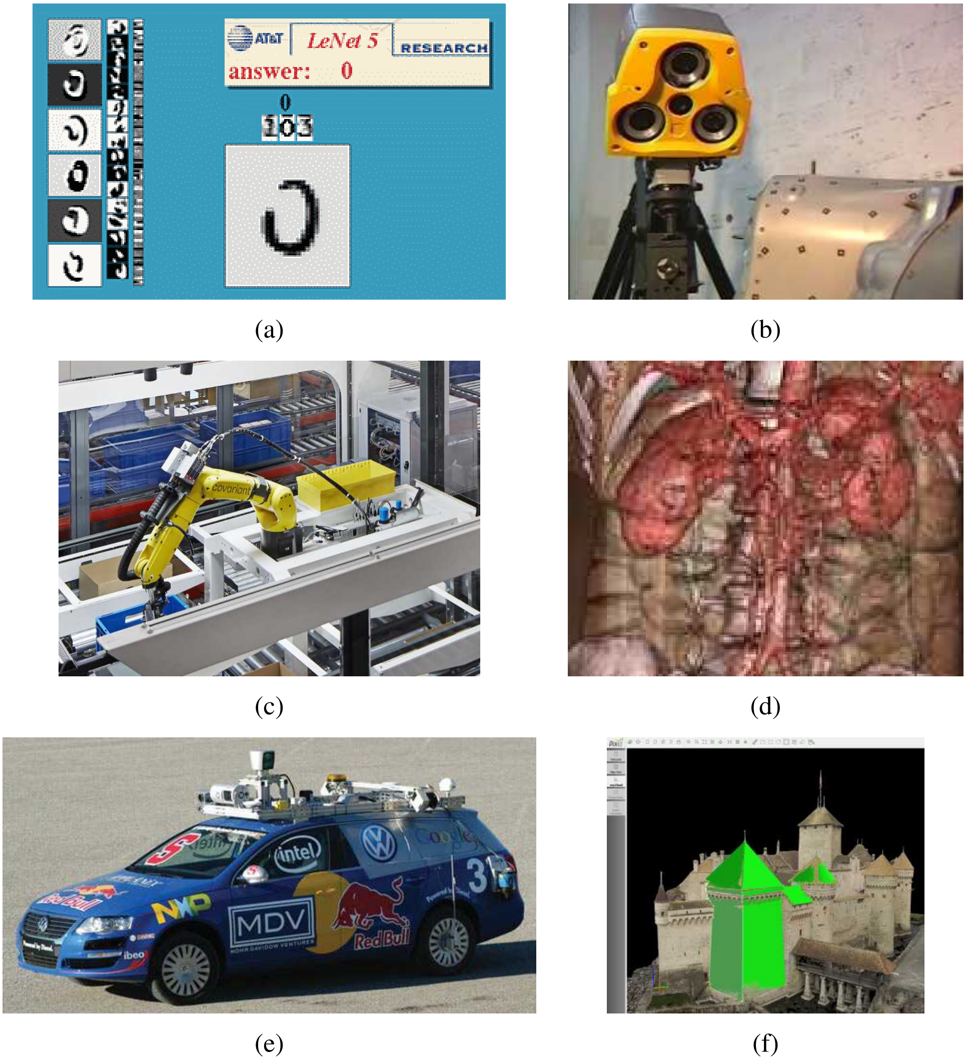


Figure 1.4 *Some industrial applications of computer vision: (a) optical character recognition (OCR), http://yann.lecun.com/exdb/lenet; (b) mechanical inspection, http://www.cognitens.com; (c) warehouse picking, https://covariant.ai; (d) medical imaging, http://www.clarontech.com; (e) self-driving cars, (Montemerlo, Becker* et al. *2008) © 2008 Wiley; (f) drone-based photogrammetry, https://www.pix4d.com/blog/mapping-chillon-castle-with-drone.*

precise *matting* to insert new elements between foreground and background elements (Chuang, Agarwala *et al.* 2002).

- **Motion capture (mocap):** using retro-reflective markers viewed from multiple cameras or other vision-based techniques to capture actors for computer animation;

- **Surveillance:** monitoring for intruders, analyzing highway traffic and monitoring pools for drowning victims (e.g., https://swimeye.com);

- **Fingerprint recognition and biometrics:** for automatic access authentication as well as forensic applications.

David Lowe's website of industrial vision applications (http://www.cs.ubc.ca/spider/lowe/vision.html) lists many other interesting industrial applications of computer vision. While the above applications are all extremely important, they mostly pertain to fairly specialized kinds of imagery and narrow domains.

In addition to all of these industrial applications, there exist myriad *consumer-level* applications, such as things you can do with your own personal photographs and video. These include:

- **Stitching:** turning overlapping photos into a single seamlessly stitched panorama (Figure 1.5a), as described in Section 8.2;

- **Exposure bracketing:** merging multiple exposures taken under challenging lighting conditions (strong sunlight and shadows) into a single perfectly exposed image (Figure 1.5b), as described in Section 10.2;

- **Morphing:** turning a picture of one of your friends into another, using a seamless *morph* transition (Figure 1.5c);

- **3D modeling:** converting one or more snapshots into a 3D model of the object or person you are photographing (Figure 1.5d), as described in Section 13.6;

- **Video match move and stabilization:** inserting 2D pictures or 3D models into your videos by automatically tracking nearby reference points (see Section 11.4.4)[2] or using motion estimates to remove shake from your videos (see Section 9.2.1);

- **Photo-based walkthroughs:** navigating a large collection of photographs, such as the interior of your house, by flying between different photos in 3D (see Sections 14.1.2 and 14.5.5);

[2] For a fun student project on this topic, see the "PhotoBook" project at http://www.cc.gatech.edu/dvfx/videos/dvfx2005.html.

- **Face detection:** for improved camera focusing as well as more relevant image searching (see Section 6.3.1);

- **Visual authentication:** automatically logging family members onto your home computer as they sit down in front of the webcam (see Section 6.2.4).

The great thing about these applications is that they are already familiar to most students; they are, at least, technologies that students can immediately appreciate and use with their own personal media. Since computer vision is a challenging topic, given the wide range of mathematics being covered[3] and the intrinsically difficult nature of the problems being solved, having fun and relevant problems to work on can be highly motivating and inspiring.

The other major reason why this book has a strong focus on applications is that they can be used to *formulate* and *constrain* the potentially open-ended problems endemic in vision. Thus, it is better to think back from the problem at hand to suitable techniques, rather than to grab the first technique that you may have heard of. This kind of working back from problems to solutions is typical of an **engineering** approach to the study of vision and reflects my own background in the field.

First, I come up with a detailed problem definition and decide on the constraints and specifications for the problem. Then, I try to find out which techniques are known to work, implement a few of these, evaluate their performance, and finally make a selection. In order for this process to work, it is important to have realistic **test data**, both synthetic, which can be used to verify correctness and analyze noise sensitivity, and real-world data typical of the way the system will finally be used. If machine learning is being used, it is even more important to have representative unbiased **training data** in sufficient quantity to obtain good results on real-world inputs.

However, this book is not just an engineering text (a source of recipes). It also takes a **scientific** approach to basic vision problems. Here, I try to come up with the best possible models of the physics of the system at hand: how the scene is created, how light interacts with the scene and atmospheric effects, and how the sensors work, including sources of noise and uncertainty. The task is then to try to invert the acquisition process to come up with the best possible description of the scene.

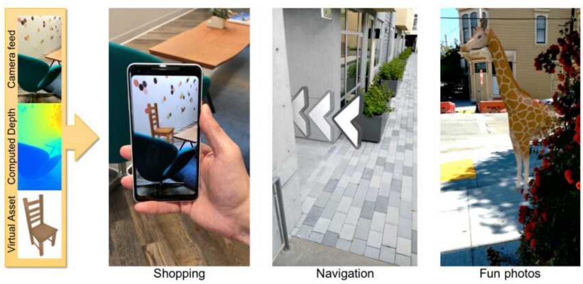
The book often uses a **statistical** approach to formulating and solving computer vision problems. Where appropriate, probability distributions are used to model the scene and the noisy image acquisition process. The association of prior distributions with unknowns is often called *Bayesian modeling* (Appendix B). It is possible to associate a risk or loss function with

[3]These techniques include physics, Euclidean and projective geometry, statistics, and optimization. They make computer vision a fascinating field to study and a great way to learn techniques widely applicable in other fields.

(a)

(b)

(c)



(d)

Figure 1.5 *Some consumer applications of computer vision: (a) image stitching: merging different views (Szeliski and Shum 1997) © 1997 ACM; (b) exposure bracketing: merging different exposures; (c) morphing: blending between two photographs (Gomes, Darsa* et al. *1999) © 1999 Morgan Kaufmann; (d) smartphone augmented reality showing real-time depth occlusion effects (Valentin, Kowdle* et al. *2018) © 2018 ACM.*

misestimating the answer (Section B.2) and to set up your inference algorithm to minimize the expected risk. (Consider a robot trying to estimate the distance to an obstacle: it is usually safer to underestimate than to overestimate.) With statistical techniques, it often helps to gather lots of training data from which to learn probabilistic models. Finally, statistical approaches enable you to use proven inference techniques to estimate the best answer (or distribution of answers) and to quantify the uncertainty in the resulting estimates.

Because so much of computer vision involves the solution of inverse problems or the estimation of unknown quantities, my book also has a heavy emphasis on **algorithms**, especially those that are known to work well in practice. For many vision problems, it is all too easy to come up with a mathematical description of the problem that either does not match realistic real-world conditions or does not lend itself to the stable estimation of the unknowns. What we need are algorithms that are both **robust** to noise and deviation from our models and reasonably **efficient** in terms of run-time resources and space. In this book, I go into these issues in detail, using Bayesian techniques, where applicable, to ensure robustness, and efficient search, minimization, and linear system solving algorithms to ensure efficiency.[4] Most of the algorithms described in this book are at a high level, being mostly a list of steps that have to be filled in by students or by reading more detailed descriptions elsewhere. In fact, many of the algorithms are sketched out in the exercises.

Now that I've described the goals of this book and the frameworks that I use, I devote the rest of this chapter to two additional topics. Section 1.2 is a brief synopsis of the history of computer vision. It can easily be skipped by those who want to get to "the meat" of the new material in this book and do not care as much about who invented what when.

The second is an overview of the book's contents, Section 1.3, which is useful reading for everyone who intends to make a study of this topic (or to jump in partway, since it describes chapter interdependencies). This outline is also useful for instructors looking to structure one or more courses around this topic, as it provides sample curricula based on the book's contents.

1.2 A brief history

In this section, I provide a brief personal synopsis of the main developments in computer vision over the last fifty years (Figure 1.6) with a focus on advances I find personally interesting and that have stood the test of time. Readers not interested in the provenance of various ideas and the evolution of this field should skip ahead to the book overview in Section 1.3.

[4] In some cases, deep neural networks have also been shown to be an effective way to speed up algorithms that previously relied on iteration (Chen, Xu, and Koltun 2017).